Technische Universität München

Fakultät für Mathematik

Lehrstuhl für Operations Research

Cuts, Paths, and Processes in Graphs

Marinus Christopher Johannes Gottschau

Vollständiger Abdruck der von der Fakultät für Mathematik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzende:		Prof. Dr. Nina Gantert
Prüfer der Dissertation:	1.	Prof. Dr. Andreas S. Schulz
	2.	Prof. Dr. Jannik Matuschke (Katholieke Universiteit Leuven)

Die Dissertation wurde am 01.07.2020 bei der Technischen Universität München eingereicht und durch die Fakultät für Mathematik am 02.09.2020 angenommen.

Abstract

This work studies several optimization problems that are defined in graphs. These include graph cuts with novel objectives, disjoint shortest paths, and three different processes on graphs. We give several polynomial time algorithms for restricted instances and analyze boundaries of polynomial time solvability of the aforementioned problems. Additionally, we address questions related to graph processes such as long-term behavior. We also provide new bounds, e.g. for minimal percolating sets in bootstrap percolation.

Zusammenfassung

In dieser Arbeit untersuchen wir verschiedene auf Graphen definierte Optimierungsprobleme. Dabei betrachten wir Partitionsprobleme mit neuen Zielfunktionen, disjunkte kürzeste Wege sowie drei verschiedene Prozesse auf Graphen. Wir entwickeln polynomielle Algorithmen für eingerschränkte Instanzen und analysieren jeweils die Komplexität der allgemeinen Probleme. Außerdem beantworten wir extremale Fragen zum Prozessverhalten. Zudem beweisen wir neue Schranken, beispielsweise für die Größe perkolierender Mengen bei Bootstrap Perkolation.

Acknowledgments

I would like to thank everybody that in some way or another makes me look back on wonderful years. But especially, I would like to thank the following people:

Exceptionally, cheers to Andreas S. Schulz, for his supervision, helpful comments, support, and the freedom I was given choosing my directions of research. Additionally, I am more than grateful for all travel opportunities, to insightful conferences and workshops, that brought many interesting problems to my attention. Overall, thank you for providing me such an outstanding working environment! Also, I would like to thank Jannik Matuschke for examining the thesis and Nina Gantert for presiding the oral examination.

Thank you as well to all my co-authors of published and unpublished projects: Clara Waldmann [46, 49], Cristina Toninelli [48], Kilian Matzke [47, 48], Jannik Matuschke [51], Marilena Leichter [50], Marcus Kaiser [46, 49], Markus Heydenreich [48], Hermann Haverkort [47], Felix Happach [46], and Susanne Albers [1].

Thank you to everyone I shared offices with in Karlstrasse or Garching: Sebastian, Clara, and Viviana, also, of course, the frequent office-couch visitors Ulf, Marcus and Marilena. To Katie also in particular for making me start to run. To Yiannis for his personal advice during my PhD and for his many exiting quotes of wisdom.

I am very much happy for the great working atmosphere at the group of Operations Research as well. I would like to give special gratitude to all colleagues and friends that made discussions and meetings during lunch and coffee breaks very enjoyable — friendly ears for any kind of problems, also beyond work. Thank you folks!

Also, gratitude to all my friends for giving me a likable past few years outside university — with distraction whenever needed, numerous fantastic travels and hikes, plenty of game nights, and many other activities. Finally, thank you to Krissi and my entire family, for all their love, pieces of advice in tough times, and support during all stages of my PhD project.

Contents

Chapter 1

Introduction

This dissertation deals with problems from two major fields of research. In Chapter 2, we study optimization problems related to cuts and paths in graphs. The problems studied are motivated by applications such as conflict free package routing in networks or clustering in data science. At first, we generalize acyclic mixed graphs and give an algorithm that solves the k disjoint paths problem in polynomial time on this restricted set of instances for constant k. Using this result, we then give a polynomial time algorithm for undirected two disjoint shortest paths on graphs with non-negative edge weights. Furthermore, we analyze novel objectives for graph bipartition problems that are degree dependent, including maximization and minimization of minimum and maximum cut-degree, respectively. We also generalize the standard notion of graph degeneracy and characterize boundaries of polynomial time solvability for all introduced objectives. We also derive approximation algorithms, some with provably best possible approximation guarantees unless $P{=}NP$.

The chapter finishes with another set of optimization problems related to graph (multi)cuts. Again, motivated by applications, e.g. airport controls, we pose a model of network security where a set of invaders has to be deterred from reaching their respective destination by installing security checks on the graph. Novel objectives, like total path length minimization for journeyers with updated edge costs for cut edges — edges with security checks — are introduced. We study both, directed and undirected graphs and prove the hardness of the general problems but give polynomial time algorithms for several special cases.

In Chapter 3, the second part of the thesis, we consider three different but related (infection) processes on graphs. All processes can be modeled as discrete time processes with vertex states with update rules that are only governed by the vertex's and its neighbors' states. First, we study bootstrap percolation, a process that corresponds to the Ising Model and is used to describe ferromagnetism in crystals at zero temperature. The graphs analyzed here are degenerate graphs and grid graphs. For degenerate graphs, we derive a tight bound on the size of the infected set at the end of the process. This bound also gives rise to a bound on the size of sets that infect the entire graph — what is called percolating sets. For grids, we work towards an answer to an extremal question on the maximum size of minimal percolating sets. Note, so-called

minimal percolating sets are sets infecting the entire graph, but no proper subset has this property.

The second process is an infection process with three states (healthy, passive and infected) with a process parameter governing infection rates for the different non-infected states. The process is motivated by applications in statistical physics and has relations to the Fredrickson-Andersen one spin facilitated model that is used to model glassy dynamics. We prove a phase transition in that parameter of the process on the infinite line, i.e. $\mathbb{Z}$. This phase transition means, if the parameter is small, infection dies out, if the parameter is large, the infection survives.

Finally, the third process studied resembles an opinion formation process on two options in which, according to some order, a set of individuals with influence relations resembled by a graph takes a vote on either of the options. Direct applications of the studied process include many public elections such as the US presidential election. Initial preferences of individuals are random and only revealed once the individual takes a vote. Finding a fixed (non-adaptive) order that maximizes the number of votes for one of the options is proved to be *NP*-hard. But, we give algorithms that find non-adaptive orders as well as adaptive orders with tight performance guarantees for certain graphs. If initial preferences were publicly known, the process agrees with bootstrap percolation.

In the remainder of this introductory chapter, we introduce basic notation that is used throughout all sections. Whenever notations or definitions are only needed in very particular parts of the thesis, for convenience, we give the definitions in the respective sections. Definitions given in the introduction are non-exhaustive but are rather needed to fix notations and conventions for which different definitions exist in literature. Still, a profound knowledge of the reader on graph theory (e.g. [17]), algorithms and complexity theory (e.g. [73]), and probability theory (e.g. [82]) is assumed.

1.1 Sets and Functions

We use standard notation for numbers, i.e. let $\mathbb{N} := \{1, 2, \dots\}$ (and $\mathbb{N}_0 := \{0\} \cup \mathbb{N}$) denote the *natural numbers* (including zero). The set of *integers* is denoted by $\mathbb{Z}$, and finally let $\mathbb{R}$ be the set of *real numbers*. We write $\mathbb{R}_{\geq 0} := \{x \in \mathbb{R} : x \geq 0\}$ for the set of *positive reals*. Also, we abbreviate $[n] := \{1, \dots, n\}$ and $[n]_0 := \{0, 1, \dots, n\}$

For a set S we let $\mathcal{P}(S) := \{S' \subseteq S\}$ denote the *powerset* of S. Furthermore, let $\binom{S}{2} := \{\{v_1, v_2\} : v_1, v_2 \in S, v_1 \neq v_2\}$ denote all subsets of S of size 2. For two sets A, B let $A \triangle B := A \cup B \setminus (A \cap B)$ denote the *symmetric difference* of A and B.

Given a function $f : A \to B$ with a discrete domain A. Let $a \in A$, then, we interchangeably use the standard and a vector representation of f, i.e. let $f_a := f(a)$. Additionally, for a subset $A' \subseteq A$ we define $f(A') := \sum_{a \in A'} f_a$.

For our purposes, a function $\pi : V \to [n]$ is called *weak order* of V of width k if

$|\{v \in V : \pi(v) = i\}| \leq k$ for all $i \in [n]$. If π is a bijection, we call it *order* of V. Note, π itself is not a relation, but we make use of the order that is induced naturally by the respective values, i.e. let $x \preceq y$ if $f(x) < f(y)$ or $x = y$. Therefore, the (partial) order $\preceq$ that is induced by π is reflexive, transitive, and also antisymmetric. Also, the notion of width of the weak order π agrees with the notion of width of the relation $\preceq$.

1.2 Graphs

Directed and Undirected Graphs:

Let $G := (V_G, E_G)$ be an *undirected graph* with vertex set V_G and edge set $E_G \subseteq \binom{V_G}{2}$. We say two vertices $u, v \in V_G$ are *adjacent* if there exists an edge $e = \{u, v\} \in E_G$ and write $u \sim v$. We also say the edge e is *incident* to u and v and vice versa. By $N(u) := \{v \in V_G : u \sim v\}$ we denote the *neighbors* of u. Two edges $e, e' \in E_G$ are incident if $e \cap e' \neq \emptyset$. For every vertex $v \in V_G$ we let $\delta_G(v) := \{e \in E : v \in e\}$ denote the set of edges incident to v. Furthermore, let $\deg_G(v) := |\delta_G(v)|$ be the *degree* of v. Last but not least, let $\Delta(G) := \max_{v \in V} \deg_G(v)$ be the *maximum degree* of G.

If the graph $G = (V_G, A_G)$ is *directed*, we denote the edge/arc set with A_G. In that case $A_G \subseteq V_G \times V_G$. For directed graphs we say a vertex $v \in V$ is incident to $a \in A$ if $a \in \{(u, v), (v, u)\}$ for some $u \in V$. Two arcs $a, a' \in A_G$ are incident if $a = (u, v)$ and $a' = (v, w)$ for $u, v, w \in V_G$. We let $\delta^-_G(v) := \{(u, v) \in A\}$ and $\delta^+_G(v) := \{(v, u) \in A\}$ be the sets of *ingoing* and *outgoing arcs*, respectively. Furthermore, $\deg^-_G(v) := |\delta^-_G(v)|$ and $\deg^+_G(v) := |\delta^+_G(v)|$ are called *in-degree* and *out-degree* of $v \in V_G$, respectively. To simplify notation, we omit G as subscript if it is clear which graph we are referring to.

Given $G = (V, E)$ and $H = (V', E')$ with $V' \subseteq V$ and $E' \subseteq E \cap \binom{V'}{2}$. We call H a *subgraph* of G. Furthermore, we say H is an *induced subgraph* of G on V', if $E' = \{\{u, v\} \in E : u, v \in V'\}$. The induced subgraph $H = (V', E')$ is also denoted as $G[V']$. In some parts, subgraphs arise from a subset of edges. Instead of defining an auxiliary subgraph, we describe the respective subgraph by its edge set in subscripts, e.g. we write $\delta_{E'}(v)$ instead of defining $G[E'] := (V, E')$ and using $\delta_{G[E']}(v)$.

Paths, Trails, and Walks:

Given a graph $G = (V, E)$. A *path* P is a sequence of consecutively incident edges without vertex repetitions, e.g. $e_1 = \{v_1, v_2\}, e_2 = \{v_2, v_3\}, \ldots, e_{k-1} = \{v_{k-1}, v_k\} \in E$ with $v_i \neq v_j$ for $i \neq j$. We slightly abuse notation and let $P := (e_1, \ldots, e_k)$ but write $v \in P$ and $e \in P$ if there exists $i \in [k]$ such that $v = v_i$ and $e = e_i$, respectively.

The same notation is used for directed paths, i.e. let $a_1 = (v_1, v_2), a_2 = (v_2, v_3), \ldots, a_{k-1} = (v_{k-1}, v_k) \in A$ with $v_i \neq v_j$ for $i \neq j$. Define $P := (a_1, \ldots, a_{k-1})$ but write $v \in P$ and $a \in P$ if there exists $i \in [k]$ such that $v = v_i$ and $a = a_i$, respectively. In both cases, directed and undirected, we refer to P as v_1-v_k-path of *length* $k - 1$.

A *trail* T is a sequence of consecutively incident edges/arcs without edge/arc repetitions. A *walk* W is a sequence of consecutively incident edges/arcs allowing for both, vertex and edge/arc repetitions. We use the same notations for containment of vertices and edges/arcs for trails and walks as for paths. Also see Figure 1.1.

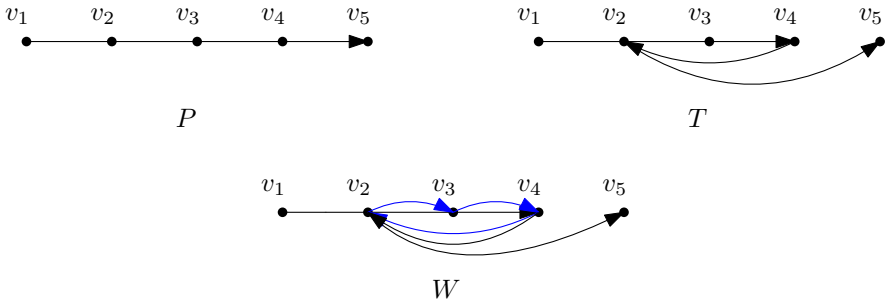


Figure 1.1: A path P of length 4, a trail T of length 5, and a walk W of length 8.

Often we consider edge-weighted[1] graphs represented by a triple $G = (V, E, w)$ with $w : E \to \mathbb{R}$.[2] For the weighted case, we may define a notion of *distance*, to be more precise, define a function dist : $V \times V \to \mathbb{R}$. We then let $\text{dist}(u, v) := \min_{P \text{ is } u\text{-}v\text{-path}} \sum_{e \in P} w_e$. Also, for graphs without edge weights we let $w \equiv 1$ to stay in line with our definition of distance and the definition of the length of a path. We call a sequence of consecutively incident edges, where additionally the first and last edge are incident, i.e. $C_k := (\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{k-1}, v_k\}, \{v_k, v_1\})$ with $v_i \neq v_j$ for $i \neq j$, a *cycle* of length k.

Notions of Connectivity:

We say a graph $G = (V, E)$ is *connected* if there exists an u-v-path for all $u, v \in V$. It is said to be k-edge-connected if there are k edge-disjoint u-v-paths for any pair $u, v \in V$. If there are k vertex-disjoint u-v-paths for any pair $u, v \in V$, the graph is called k-vertex-connected.

For directed graphs, however, there exist three different notions of connectivity that we shall use throughout the thesis. Let $G = (V, A)$ be a directed graph. We say G is *weakly connected* if the undirected graph that is obtained by omitting directions or all arcs is connected. The graph G is said to be *connected* if for every pair $s, t \in V$ there exists a directed s-t-path or a directed t-s-path. Last but not least, a graph is *strongly connected* if for all pairs $s, t \in V$ there exists a directed s-t-path.

[1] sometimes we also refer to edge weights as edge lengths. In particular, if weights are positive, we mainly use the term length instead.

[2] analogous, $w : A \to \mathbb{R}$, for directed graphs

Special Graphs:

Given a graph $G = (V, E)$. We denote by $G^c = (V, E^c)$ with $E^c = \binom{V}{2} \setminus E$ the *complement* of G.

If a graph $G = (V, E)$ does not contain cycles, it is called a *forest*. If G is a forest and is connected, we call G a *tree*. If $G = (V, A)$ is directed and does not contain a directed cycle, G is said to be *acyclic*.

A graph $G = (V, E)$ is *bipartite* if there exists a partition $A \cup B = V$ such that $E \subseteq \{\{u, v\} : u \in A, v \in B\}$.

The graph with $V = [m] \times [n]$ and $E := \{\{(i, j), (i + 1, j)\} : i \in [m - 1], j \in [n]\} \cup \{\{(i, j), (i, j + 1)\} : i \in [m - 1]\}$ is referred to as an m times n *grid graph*. For an illustration see Figure 1.2.

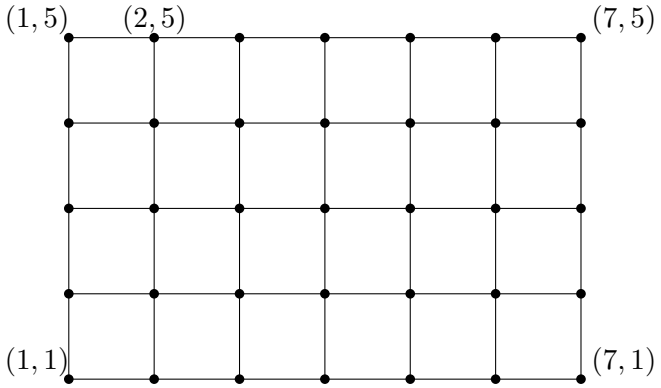


Figure 1.2: A $[7] \times [5]$ grid graph is depicted in the figure.

Complete graphs $G = (V, E)$ (also called *cliques*) are graphs with $E = \binom{V}{2}$. We write K_n for a clique on n vertices. A graph G, respectively its vertex set, is called *independent set* if its complement graph G^c is a clique, i.e. $E_G = \emptyset$ and $E_{G^c} = \binom{V}{2}$. The *independence number* $\alpha(G)$ of a graph G is maximum cardinality among all induced independent sets in G. Similarly, the *clique number* $\kappa(G)$ of a graph G is the size of the largest subgraph of G that is a clique.

A graph is called *planar* if there exists an embedding into the plane without any edge crossings. Last but not least, a graph $G = (V, E)$ is *d-degenerate*, if every subgraph contains a vertex of degree at most d. For degenerate graphs there exists an ordering of the vertices on a line such that every vertex is adjacent to at most d neighbors on its left (vertices with smaller label)[3]. Such a sequence is called an *Erdős-Hajnal sequence*.

[3]The proof is an easy exercise.

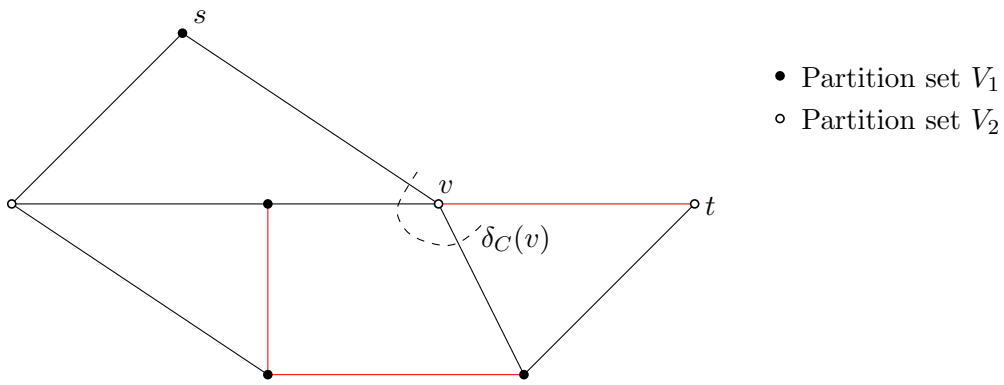


Figure 1.3: A cut on a graph with black cut edges forming the cut set C. Since, s and t are contained in different sets, the cut is also an s-t-cut.

Cuts:

Given a graph $G = (V, E)$. A bipartition of the vertex set V, i.e. $V_1 \dot\cup V_2 = V$, with $V_1, V_2 \neq \emptyset$, is called *cut*. We shall use the terms cut and bipartition interchangeably. All edges in the cut set $C := \{\{u, v\} \in E : u \in V_1, v \in V_2\}$ are said to cross the cut or called cut edges. Given two vertices $s, t \in V$ we call a cut with $s \in V_s$ and $t \in V_t$ to be an *s-t-cut*. The *cut-degree* of a vertex v on a cut with cut set C is defined as $\deg_C(V) = |\delta_C(v)|$.

This notion of cuts can also be extended to directed graphs by replacing edges by arcs in the above definition.

Colorability:

A graph $G = (V, E)$ is said to be *k-colorable*, if there exists a map $\pi : V \to [k]$ such that $\pi(v) \neq \pi(u)$ for all $\{u, v\} \in E$. The smallest value of k, such that G is k-colorable is called *chromatic number* of G.

The graph is said to be *(d, k)-colorable* if $|\{u : \{u, v\} \in E \wedge \pi(v) = \pi(u)\}| \leq d$ for all $v \in V$. The smallest value $k \in \mathbb{N}$ for which G is (d, k)-colorable, is also called *d-defective chromatic number* of G. Note, for $d = 1$ this agrees with the chromatic number.

Matchings:

Given a graph $G = (V, E)$ and a subset of edges $E' \subseteq E$. The edge set E' is called a *matching* if $\deg_{E'}(v) \leq 1$ for all $v \in V$. A matching is a *perfect matching* if every vertex is adjacent to exactly on edge in the matching.

Flows:

Given a directed graph $G = (V, A, c)$ with $c : A \to \mathbb{R}_{\geq 0}$. For $a \in A$ we call c_a *capacity* of arc a. Let $s, t \in V$, we call $f : A \to \mathbb{R}_{\geq 0}$ an *s-t-flow* if

1. $f_a \leq c_a$ for all $a \in A$ (Capacity Constraints),

2. $\sum_{a \in \delta^-(v)} f_a = \sum_{a \in \delta^+(v)} f_a$ for all $v \in V \setminus \{s, t\}$ (Flow Conservation).

We define value$(f) := \sum_{a \in \delta^-(s)} f_a - \sum_{a \in \delta^+(s)} f_a$ to be the *value* of the flow.
Given an additional cost function on the arcs $u : A \to \mathbb{R}$, the *cost* of an s-t-flow f is given by $u(f) := \sum_{a \in A} u_a f_a$.

Graph Processes:

Given a graph and a finite set of vertex states. A *process* on graph is a sequence of changes in vertex states over time according to some predefined process rules. In particular, all processes analyzed in this thesis share a common feature. Update rules for vertex states only depend on the vertex's current state and the states of its neighbors in the graph.

An example of such a process is *r-neighbor bootstrap percolation*. This is a two state process a graph with vertex states uninfected and infected. From an initially infected set of vertices vertex states update in discrete time steps. If a vertex is adjacent to at least r infected vertices, it becomes infected itself and remains infected for all future time steps.

1.3 Complexity Theory

By P we denote the class of decision problems solvable in polynomial time. Let NP be the class of decision problems solvable in non-deterministic polynomial time. We say an optimization problem is NP-hard, if the corresponding decision problem is NP-hard. To be more precise, given, e.g , a minimization problem with objective function f and feasible solutions X. The corresponding decision problem is defined as follows: Given some $l \in \mathbb{Q}$, does there exist a feasible solution $x \in X$ with $f(x) \leq l$. The problem for maximization problems is defined analogously reversing the inequality sign. A decision problem is said to be NP-complete if it is NP-hard and in NP. The following NP-complete decision problems are used in several parts of the thesis. We also would like to point the reader to Garey and Johnson's long list of NP-complete problems that turned out very useful in many places [42].

1.3.1 Short List of NP-complete Problems:

Boolean Satisfiability Problem:

A *truth assignment* is a function $w : X \to \{0,1\}^n$ that assigns truth values to a set of *Boolean variables* $X = \{x_1, \ldots, x_n\}$. We say x_i is assigned false if $w(x_i) = 0$, and assigned true else. A *Boolean expression* is a (syntactically correct) formula making use of *Boolean operations* on Boolean variables, i.e. the three following elementary operations:

1. *Conjunction* of two Boolean variables/expressions $X_1 \wedge X_2$:

$$X_1 \wedge X_2 \equiv \begin{cases} 1 & \text{if } X_1 = X_2 = 1, \\ 0 & \text{else} \end{cases}$$

2. *Disjunction* of two Boolean variables/expressions $X_1 \vee X_2$:

$$X_1 \vee X_2 \equiv \begin{cases} 0 & \text{if } X_1 = X_2 = 0, \\ 1 & \text{else} \end{cases}$$

3. *Negation* of a Boolean variable/expression $\neg$:

$$\neg X_1 \equiv \begin{cases} 0 & \text{if } X_1 = 1, \\ 1 & \text{if } X_1 = 0. \end{cases}$$

We call $l \in \{x, \neg x\}$ *literal* and a Boolean expression of the form $C = (y_1 \vee y_2 \vee \ldots \vee y_l)$ a *clause* of length l. Additionally, we say a Boolean expression is *satisfiable* if there exists a truth assignment such that the expression evaluates to true (resp. 1). For literals that are negated variables, i.e. $\neg x$, we also write $\bar{x}$.

The following decision problem is called SAT: Given a Boolean formula, does there exist an assignment of true/false to $x_1, \ldots, x_n$ such that the formula evaluates to true.

A Boolean expression $C_1 \wedge C_2 \wedge \ldots \wedge C_m$ with clauses $C_1, \ldots, C_m$ over a variable set X is said to be in *conjunctive normal form*.

Given a Boolean formula in conjunctive normal form, where every clause is of length at most 3. The decision problem, does there exist an assignment of true/false to $x_1, \ldots, x_n$ such that this expression evaluates to true, is called 3-SAT.

If every clause contains exactly 3 literals, the decision problem is called exact-3-SAT. Slightly more restrictive, not-all-equal 3-SAT, requires at least one true and one false literal per clause. A yet even more restrictive variant of truth assignments of exact-3-SAT, requires exactly one literal per clause to evaluate to true. This problem is referred to as 1-in-3-SAT. All the above variations of SAT are *NP*-complete [42].

Disjoint Paths:

Given a graph $G = (V, E)$ and k pairs of source and sink vertices (s_i, t_i), $i \in [k]$. The decision problem, do there exist vertex disjoint paths P_i such that P_i is an s_i-t_i-path for $i \in [k]$, is called k *disjoint paths problem* (k-DPP). We refer to the directed version as directed k disjoint paths problem (k-dDPP).

Noteworthy is the seminal result by Robertson and Seymour [79] that k-DPP is solvable in polynomial time if k is constant. However, if k is part of the input, k-DPP is *NP*-complete [58]. For directed graphs, the picture is slightly different, 2-dDPP is already *NP*-complete [39].

1.3.2 Polynomial Time Solvable Problems:

In the following, we give a list of (optimization/decision) problems occurring in this thesis, for which we make use of the existence of polynomial time algorithms solving them.

Shortest Paths:

Given a (possibly directed) graph $G = (V, E)$ with edge weights $w : E \to \mathbb{R}_{\geq 0}$ and $s, t \in V$. A shortest s-t-path, can be computed in polynomial time, if such a path exists. This can be generalized to conservative edge weights, i.e. weights for which there exists no (directed) negative cycle [11].

Undirected Disjoint Paths:

As mentioned before, Robertson and Seymour [79] gave a polynomial time algorithm for k-DPP for constant k.

Matchings:

Given a graph $G = (V, E)$ with edge weights w.

1. Find a maximum weight matching.
2. Find a minimum weight perfect[4] matching.

Both variants can be solved in polynomial time using appropriate adaptions of Edmond's blossom algorithm [32].

[4] Of course, the algorithm finds a perfect matching only if one exists and returns non-existence otherwise.

Degeneracy:

Let $G = (V, E)$ be a graph. The *degeneracy* of the graph, i.e. the minimum $d \in \mathbb{N}$ such that G is d-degenerate, can be computed in polynomial time. Also, an Erdős-Hajnal sequence can be found in polynomial time [68].

Maximum Flow:

Given a directed graph $G = (V, A)$ with edge capacities $c : E \to \mathbb{R}_{\geq 0}$ and $s, t \in V$. There exists an algorithm that computes a maximum s-t-flow in polynomial time, e.g. the Edmonds–Karp algorithm [33].

Minimum-cost Flow:

Given a graph $G = (V, A)$ with edge capacities $c : E \to \mathbb{R}_{\geq 0}$ and edge cost $u : E \to \mathbb{R}_{\geq 0}$. Furthermore, let $s, t \in V$ and $b \geq 0$. A minimum cost s-t-flow f with $\text{value}(f) = b$ can be computed in polynomial time, if it exists [44].

1.4 Outline of the Thesis

In the following paragraphs, we give an outline of the thesis.

In Chapter 2, we algorithmically study optimization problems related to shortest paths and vertex partitions (cuts).
In Section 2.1, we give a polynomial time algorithm for non-negative two disjoint shortest paths. The section is split into two parts, first in Section 2.1.4, an algorithm for k-DPP on weakly acyclic mixed graphs is presented. That algorithm is then used in Section 2.1.5 to solve 2-DSPP. This section is a joint work with Marcus Kaiser and Clara Waldmann [49].
Section 2.2 introduces novel objectives for graph cut problems. We give a full characterization of complexity for the Min-Max-, Min-Min-, Max-Min-, Max-Max-Cut-Degree Problems in Section 2.2.4. Furthermore, in Section 2.2.5, we introduce a new notion of degeneracy and analyze its computational complexity. For this new graph degeneracy, we derive hardness results and an approximation algorithm.
Finally, Section 2.3 presents a work in progress with Jannik Matuschke [51]. It considers yet another novel set of objectives for vertex bipartition problems, that include path length minimization in Section 2.3.2 and flow maximization in Section 2.3.3. Besides new objectives, the main feature of the studied models are increased cost of cut edges. We present several polynomial time algorithms and also characterize boundaries of polynomial time solvability of the problems.

Chapter 3 of the thesis studies three processes on graphs all making use of up to three vertex states.
Section 3.1 studies an infection process (bootstrap percolation) on graph classes, such as degenerate graphs as well as grid graphs, c.f. Section 3.1.2 and Section 3.1.3, respectively. It revisits and strengthens bounds for degenerate graphs published in [45]. Also, we disprove a conjecture by Morris [71] for the maximum size of inclusion minimal sets spreading the infection to all vertices on grids.
Section 3.2 proves a phase transition in a process parameter for a three-state infection process motivated by statistical physics (c.f. Section 3.2.3). This phase transition means, depending on the process parameter, that there are regimes for survival (see Section 3.2.4) or extinction (see Section 3.2.5) of the infection. Section 3.2 is joint work with Markus Heydenreich, Kilian Matzke, and Cristina Toninelli and has been published in [48].
Last but not least, the process introduced in Section 3.3 resembles an election between two options where influences among individuals are present. Interested in the order of votes, we study the hardness of finding an one-option-maximizing order for two variants of the problem. We characterize graphs that admit a non-adaptive order making use of the new graph degeneracy introduced in Section 2.2.5 in Section 3.3.4. We also give an adaptive algorithm with a tight (expected) performance guarantee for general graphs. The algorithm for orders with one vote at a time then is generalized to weak orders, allowing for more that one vertex to take a vote at a time in Section 3.3.5. Also, we present some computational experiments on random graphs of all algorithms in Section 3.3.6. This section is a work in progress with Susanne Albers [1].

In Chapter 4, we conclude the thesis and point to interesting open problems.

Chapter 2

Cuts and Paths

In this chapter, we algorithmically study problems related to cuts and paths on graphs. First, we address the disjoint shortest path problem. We give a polynomial time algorithm for the k disjoint paths problem on weakly acyclic graphs. This algorithm is then used to solve the undirected two disjoint shortest paths problem with non-negative weights in polynomial time. In the second part, we study optimization problems related to cuts with degree-depended objective functions, such as max-min-degree cuts, and also introduce new notion of degeneracy. For these problems, we derive several hardness and (in)approximability results. Finally, we analyze cut constrained problems related to network security on weighted graphs. We study new objective functions, such as total path length minimization and flow maximization, where cut edges have an increased weight and reduced capacities, respectively. Polynomial time algorithms are derived for several restricted instances.

2.1 Disjoint Shortest Paths

This section is a reprint with adaptions of a joint work with Marcus Kaiser and Clara Waldmann that has been published in [49].

2.1.1 Overview

Due to many practical applications, e.g., in communication networks, the k disjoint paths problem (k-DPP) is a well studied problem in the literature. The input of the problem is an undirected graph $G = (V, E)$ as well as k pairs of vertices $(s_i, t_i) \in V^2$ for $i \in [k]$ and the task is to decide whether there exist k paths $P_1, \dots, P_k$ such that P_i is an s_i-t_i-path and all paths are pairwise disjoint. Here, disjoint can either mean vertex-disjoint or edge-disjoint.

The k disjoint shortest path problem (k-DSPP) is a generalization of the k disjoint paths problem. The input of the problem is an undirected graph $G = (V, E)$ with edge lengths $\ell : E \to \mathbb{R}$ and k pairs of vertices $(s_i, t_i) \in V^2$ for $i \in [k]$. But here, all paths P_i for $i \in [k]$ are additionally required to be shortest s_i-t_i-paths. Note, if $\ell \equiv 0$, this agrees with k-DPP.
We shall refer to the versions of the problems in directed graphs by k-dDPP and k-dDSPP.

2.1.2 Related Work

Probably most famously, Menger's theorem [70] deals with disjoint paths which gave rise to one of the most fundamental results for network flows: the Max-Flow-Min-Cut theorem [35, 38]. Using these results, an application of any flow algorithm solves the k-dDPP if $s_i = s_j$ for all $i, j \in [k]$ or $t_i = t_j$ for all $i, j \in [k]$. Without restrictions on the input instances, all variants of the discussed problems are NP-complete if k is considered part of the input [36, 58].

Due to this, a lot of research focuses on the setting where k is considered fixed. Robertson and Seymour [79] came up with an $\mathcal{O}\big(|V|^3\big)$ algorithm for k-DPP.

In contrast to that, Fortune et al. [39] prove that k-dDPP is still *NP*-hard, even if $k = 2$. They give an algorithm that solves k-dDPP for any fixed k on directed acyclic graphs in polynomial time. Zhang and Nagamochi [84] then extended the work of Fortune et al. [39] to solve the problem on acyclic mixed graphs, which are graphs that contain arcs and edges where directing any set of edges does not close a directed cycle.

Since k-dDSPP and k-dDPP agree for $\ell \equiv 0$, all hardness results carry over. However, if all edge lengths are strictly positive Bèrczi and Kobayashi [13] give a polynomial time algorithm for 2-dDSPP. Also, for 2-DSPP with strictly positive edge lengths there exists a polynomial time algorithm that is due to Eilam-Tzoreff [34]. However,

	$\ell \equiv 0$		ℓ non-negative	
k	k-DPP	k-dDPP	k-DSPP	k-dDSPP
arb.	*NP*-hard [36, 58]	*NP*-hard [36]	*NP*-hard [34]	*NP*-hard [34]
fixed	*P*[79]	*NP*-hard [39]	open ($\ell > 0$)	open ($\ell > 0$)
			open ($\ell \geq 0$)	*NP*-hard ($\ell \geq 0$) [39]
2	*P*[79]	*NP*-hard [39]	*P* ($\ell > 0$) [34]	*P*($\ell > 0$) [13]
			P($\ell \geq 0$) *	*NP*-hard ($\ell \geq 0$) [39]

Table 2.1: Complexity of the disjoint paths problem and its variants.
* A polynomial time algorithm for the 2-DSPP on undirected graphs with non-negative edge lengths is the main result of this section.

the complexity of k-DSPP on undirected graphs with non-negative edge lengths and constant $k \geq 2$ is unknown. We settle the case $k = 2$ in this section.

Other than restricting the paths to be shortest s_i-t_i-paths, e.g., Suurballe [83] gave a polynomial time algorithm minimizing the total length, if all arc lengths are non-negative and $s_i = s_j$, $t_i = t_j$ for all $i, j \in [k]$. Björklund and Husfeldt [15] came up with a polynomial time algebraic Monte Carlo algorithm for solving 2-DPP with unit lengths where the total length of the paths is minimized.

2.1.3 Our Results

We give a polynomial time algorithm for 2-DSPP on undirected graphs with non-negative edge lengths. Combining techniques from [39] and [13] enables us to deal with edges of length zero. We consider the following problem.

Problem 1 (Undirected Two Edge-Disjoint Shortest Paths Problem)**.**
Input: *An undirected graph $G = (V, E)$ with non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$, a tuple of sources $s \in V^2$, and a tuple of sinks $t \in V^2$*
Task: *Decide whether there exist two edge-disjoint paths P_1 and P_2 in G such that P_1 is a shortest s_1-t_1-path and P_2 is a shortest s_2-t_2-path w.r.t. the edge lengths ℓ.*

The remaining section is organized as follows. In Section 2.1.4, based on the ideas of [39], we give a dynamic algorithm that solves the k-DPP in polynomial time on weakly acyclic mixed graphs, which are a generalization of directed acyclic graphs. These results are then used in Section 2.1.5 together with a similar approach as in [13] to solve the undirected 2-DSPP with non-negative edge lengths in polynomial time.

The results of this section have been obtained independently by Kobayashi and Sako [61].

2.1.4 Disjoint Paths Problem in Weakly Acyclic Mixed Graphs

In this subsection, we give an algorithm that solves k-DPP in a generalization of directed acyclic graphs. We first define mixed graphs, introduce some notations, and state the problem.

A graph $G = (V, A \cupdot E)$ is a *mixed graph* on the vertex set V with arc set $A \subseteq V^2$ and edge set $E \subseteq \binom{V}{2}$. We define $Æ(G) := A \cupdot E$. The set of ingoing (outgoing) arcs of a set of vertices $W \subseteq V$ is denoted by $\delta_A^-(W)$ $\left(\delta_A^+(W)\right)$.

For pairwise disjoint vertex sets $W_1, \dots, W_h$, we denote by $G/\{W_1, \dots, W_h\}$ the graph that results from G by contracting $W_1, \dots, W_h$ into h vertices.

A (directed) *u-w-path* P in G is a sequence of h arcs and edges $(æ_1, \dots, æ_h) \in Æ^h$ such that there exists a sequence of vertices $(u = v_1, \dots, v_{h+1} = w) \in V^{h+1}$ satisfying either $æ_i = (v_i, v_{i+1})$ or $æ_i = \{v_i, v_{i+1}\}$ for all $i \in [h]$. Two paths are *arc/edge-disjoint* (*vertex-disjoint*) if they do not have a common arc or edge (vertex).

Note that a directed acyclic graph induces natural orderings of its vertices. A linear ordering of the vertices is called a *topological ordering* if, for every arc (v, w), the tail v precedes the head w in the ordering. An ordering is called a *reverse topological ordering* if its reverse ordering is a topological ordering.

On a ground set U, a *binary relation* R is a subset of U^2. For $(u, v) \in R$, we write $u\,R\,v$. A relation R is called *reflexive*, if $u\,R\,u$ holds for all $u \in U$. For two binary relations $R, S \subseteq U^2$, the *composition* $S \circ R$ is defined by

$$\{(u, w) \in U^2 \mid \exists v \in U : u\,R\,v \wedge v\,S\,w\}.$$

Note that $\circ$ is an associative operator.

We consider the following problem for fixed k.

Problem 2 (Mixed k Arc/Edge-Disjoint Paths Problem)**.**
Input: *A mixed graph $G = (V, Æ)$, a k-tuple of sources $s \in V^k$, and a k-tuple of sinks $t \in V^k$*
Task: *Decide whether there exist k pairwise arc/edge-disjoint paths $P_1, \dots, P_k$ in G such that P_i is an s_i-t_i-path, for all $i \in [k]$.*

We give an algorithm that solves this problem on a class of mixed graphs, that generalize directed acyclic graphs:

Definition 1 (Weakly Acyclic Mixed Graphs)**.** *We call a mixed graph $G = (V, A \cupdot E)$* weakly acyclic *if the contraction of all edges E yields a directed acyclic graph without loops.*

Note that a weakly acyclic mixed graph can contain (undirected) cycles in its edge set. For a mixed graph $G = (V, Æ)$, we use the following notation in order to discuss the existence of disjoint paths.

Definition 2 (Arc/Edge-Disjoint Paths Relation)**.** *For $k \in \mathbb{N}$, we define the binary relation $\rightrightarrows_{Æ}$ on the set V^k as follows. For $v, w \in V^k$, we have $v \rightrightarrows_{Æ} w$ if there exist pairwise arc/edge-disjoint v_i-w_i-paths for all $i \in [k]$ in $Æ$. We shall also write $\rightrightarrows_G$ short for $\rightrightarrows_{Æ(G)}$.*

Since paths of length zero are allowed, the relation $\rightrightarrows_{Æ}$ is reflexive. In general, it is not transitive. When considering two relations based on two disjoint sets of arcs and edges, however, these two act in a transitive manner. In that case, the respective underlying arc/edge-disjoint paths from both relations can be concatenated. The resulting arc/edge-disjoint paths correspond to an element in the composition of the two relations.

Observation 3 (Partial Transitivity)**.** *For disjoint arc/edge sets $Æ_1, Æ_2 \subseteq Æ$ and vectors of vertices $u, v, w \in V^k$, it holds*

$$u \rightrightarrows_{Æ_1} v \wedge v \rightrightarrows_{Æ_2} w \implies u \rightrightarrows_{Æ_1 \cup Æ_2} w.$$

Algorithm 1: Dynamic Program for k-DPP in Weakly Acyclic Mixed Graphs

Input: weakly acyclic mixed graph $G = (V, A \cup E)$
Output: $\rightrightarrows_G$ on V^k

```
Find connected components V_1, ..., V_h of the subgraph (V, E) sorted
  according to a topological ordering of G/{V_1, ..., V_h};
for j = 1, ..., h do
    Compute ⇉_{G[V_j]} using an algorithm for k-DPP
end
Initialize ⇉ to the relation {(v, v) : v ∈ V^k};
for j = 1, ..., h do
    Update ⇉ to ⇉_{G[V_j]} ∘ ⇉_{δ_A^-(V_j)} ∘ ⇉
end
return ⇉
```

This observation is exploited in Algorithm 1 in order to solve Problem 2 for fixed k for weakly acyclic mixed graphs. It computes the relation $\rightrightarrows_G$ in polynomial time by dealing with the edges and arcs in G separately.

For the undirected components, i.e., the connected components of the subgraph (V, E), it uses an algorithm for edge-disjoint paths in undirected graphs (e.g., [79]) to find the relation $\rightrightarrows$ on each component.

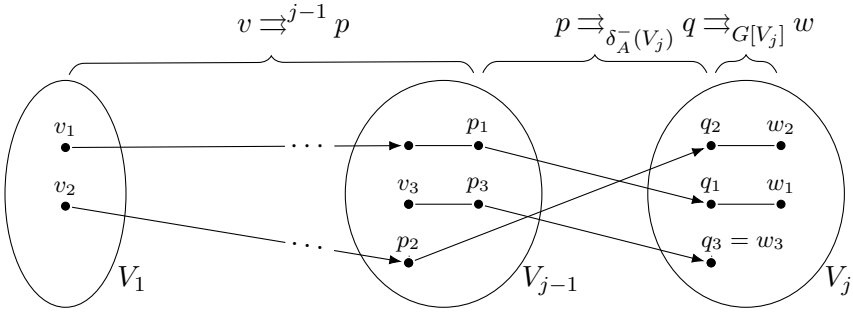


Figure 2.1: In iteration j of Algorithm 1, relation $\rightrightarrows^j$ is built by concatenating previously computed paths $\left(\rightrightarrows^{j-1}\right)$, pairwise different arcs to the next component $\left(\rightrightarrows_{\delta_A^-(V_j)}\right)$, and undirected edge-disjoint paths in the next component $\left(\rightrightarrows_{G[V_j]}\right)$.

Afterward, dynamic programming is used to compute $\rightrightarrows$ on successively larger parts of the mixed graph. As G is weakly acyclic, contracting all undirected components results in an acyclic graph. The algorithm iterates over the components in a topological ordering. Based on Observation 3, previously found arc/edge-disjoint paths are extended alternately by arcs between components and edge-disjoint paths within one component. This approach is a generalization of the methods presented in [39].

Theorem 4 (Algorithm 1: Correctness and Running Time)**.** *Let $k \in \mathbb{N}$ be fixed. Given a weakly acyclic mixed graph $G = (V, A \cup E)$, Algorithm 1 computes the relation $\rightrightarrows_G$ on V^k in polynomial time.*

Proof. Let $V = \bigcup\!\!\!\cdot_{j=1}^{h} V_j$ be the partition of V into the vertex sets of the h connected components of (V, E) as computed by the algorithm.

For all $j \in \{0, \dots, h\}$, let $Æ_j$ be the arc and edge set of $G\left[\bigcup_{l=1}^{j} V_l\right]$. In particular, $Æ_0 = \emptyset$ holds true. For each $j \in \{0, \dots, h\}$, let $\rightrightarrows^j$ be the relation $\rightrightarrows$ as computed by Algorithm 1 after the j-th iteration of Line 4. In particular, $\rightrightarrows^0$ is the relation after Line 3. In the following, we prove by induction on j that $\rightrightarrows^j$ is equal to $\rightrightarrows_{Æ_j}$.

After the initialization, this is true for $j = 0$, as $Æ_0$ contains no arcs or edges. Consider an iteration $j \in [h]$ and assume that the claim was true after the previous iteration.

"$\subseteq$": Let $v, w \in V^k$ with $v \rightrightarrows^j w$. There exist $p, q \in V^k$ such that $v \rightrightarrows^{j-1} p \rightrightarrows_{\delta_A^-(V_j)} q \rightrightarrows_{G[V_j]} w$. Using the induction hypothesis, we know $v \rightrightarrows_{Æ_{j-1}} p$. Since the arc and edge sets in the three relations are pairwise disjoint, Observation 3 yields $v \rightrightarrows_{Æ_j} w$.

"$\supseteq$": Let $v, w \in V^k$ with $v \rightrightarrows_{Æ_j} w$, and $P_i, i \in [k]$ be arc/edge-disjoint v_i-w_i-paths in $Æ_j$. Let $q_i \in V$ be the first vertex on P_i in V_i and p_i be its predecessor if they exist,

otherwise set them to w_i and q_i, respectively. As G is weakly acyclic, we have $v \rightrightarrows_{Æ_{j-1}} p$, $p \rightrightarrows_{\delta_A^-(V_j)} q$, as well as $q \rightrightarrows_{G[V_j]} w$. It follows from the induction hypothesis that $v \rightrightarrows^j w$.

The connected components of (G, E) and their topological ordering in $G/\{V_1, \dots, V_h\}$ can be computed in polynomial time. Finding edge-disjoint paths in the undirected components can also be done efficiently (e.g., [79]). A binary relation on V^k contains at most $|V|^{2k}$ elements and composing two of them can be done in time polynomial in their sizes. Hence, Algorithm 1 runs in time polynomial in the size of the input if k is fixed. □

In many settings, the problem of finding arc/edge-disjoint paths can be reduced to finding vertex-disjoint paths. Observe that arc/edge-disjoint paths in a graph correspond to the vertex-disjoint paths in its line graph and an appropriate notion of a line graph can be defined for mixed graphs as well.
For directed graphs, there is a reduction from vertex-disjoint to arc-disjoint instances based on splitting vertices. This generic reduction, however, cannot be applied to undirected or mixed graphs. Yet, Algorithm 1 can be modified slightly as follows to compute vertex-disjoint paths. An algorithm for the undirected vertex-disjoint path problem is used in Line 2. Only vectors with pairwise different elements are included in the initial relation in Line 3. Finally in Line 4, tuples $v, w \in V^k$ are related only if their sets of endpoints $\{v_i, w_i\}, i \in [k]$ are pairwise disjoint.

2.1.5 Undirected Disjoint Shortest Paths

In this subsection, we study Problem 1 on undirected graphs with non-negative edge lengths. We first transform the undirected graph G into a mixed graph and then use the results of the previous subsection to solve the transformed instance.

From Shortest to Directed Paths

Let an instance of Problem 1 be given by an undirected graph $G = (V, E)$, non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$, and $s, t \in V^2$. We are going to transform the graph G into a mixed graph such that the shortest source-sink-paths in G correspond to directed source-sink-paths in the resulting mixed graph.

Since we are interested in shortest s_1–t_1- and s_2-t_2-paths, we consider the shortest path networks rooted at s_1 and s_2. For $i \in [2]$, we define the distance function $d_i : V \to \mathbb{R}_{\geq 0}$ induced by ℓ w.r.t. s_i by $d_i(v) := \min_{s_i\text{-}v\text{-path } P} \sum_{e \in P} \ell(e)$. The *shortest path network* rooted at s_i is given by the set

$$E_i := \{\{v, w\} \in E : \ell(\{v, w\}) = |d_i(v) - d_i(w)|\}.$$

See Figure 2.3a for an example of the sets E_i.
The distances d_i induce an orientation for all edges in E_i which have a strictly positive

Figure 2.2: Gadget for resolving conflicts during the orientation of an edge $\{v, w\}$ induced by d_1 and d_2.

length. We would like to replace an edge $\{v, w\} \in E$ with $d_i(v) < d_i(w)$ by the arc (v, w) (with the same length). The orientations induced by d_1 and d_2, however, do not have to agree on the set $E_1 \cap E_2$. Introducing both arcs would neglect the fact that only one of them can be included in any set of arc/edge-disjoint paths. We will overcome this by replacing such edges by a standard gadget of directed arcs as depicted in Figure 2.2.
Consider the gadget for an edge $\{v, w\} \in E$. It contains exactly one v-w-path and one w-v-path corresponding to the two possible orientations of $\{v, w\}$. Since both share an arc, only one of two arc/edge-disjoint paths in the transformed graph can use the gadget. As further both paths consist of three arcs, setting the length of all the arcs in the gadget to $\frac{1}{3}\ell(\{v, w\})$ preserves the distances in the graph. That way, the distance functions d_i can be extended to the new vertices introduced with gadgets.

For $i \in [2]$, A_i denotes the set of arcs that result from orienting E_i w.r.t. d_i. More precisely, for $\{v, w\} \in E_i$ with $d_i(v) < d_i(w)$ the arc (v, w) is included into the set A_i if $\{v, w\} \in E_1 \triangle E_2$ or the orientation induced by d_1 and d_2 agree. Otherwise, the arcs of the v-w-path in the gadget replacing $\{v, w\}$ are added to A_i.

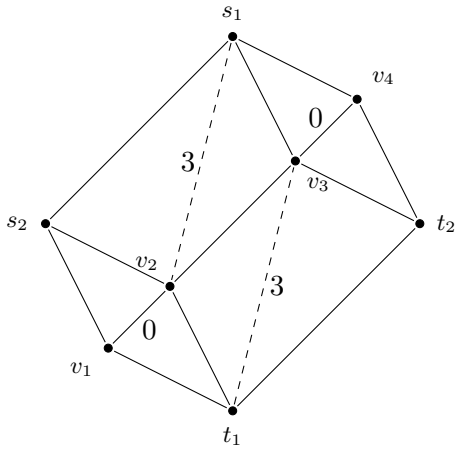
The induced orientation is only well-defined for edges with strictly positive lengths. Therefore, the set of edges with length zero $E_0 := \{e \in E : \ell(e) = 0\}$ are left undirected and have to be treated in a different manner.

Definition 5 (Partially Oriented Expansion)**.**

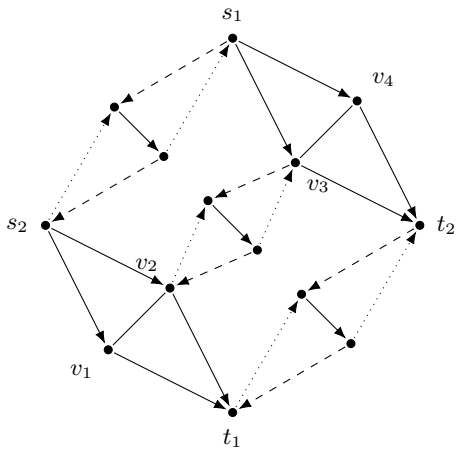
Let $G = (V, E)$ be an undirected graph with non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$ and $s \in V^2$.

The partially oriented expansion of G w.r.t. ℓ and s is the graph $\vec{G} := (W, E_0 \cup A_1 \cup A_2)$ where W is the set of vertices V augmented with additional vertices introduced with gadgets, and E_0, A_1, and A_2 are as defined above.

The partially oriented expansion of the example from Figure 2.3a is depicted in Figure 2.3b.



(a) Example: Edges without label have length 1, solid edges are in $E_1 \cup E_2$.



(b) Partially oriented expansion of (a): solid arcs are in $A_1 \cap A_2$, dashed arcs are in $A_1 \setminus A_2$, dotted arcs are in $A_2 \setminus A_1$.

Figure 2.3: Exemplary construction of partially oriented expansion.

As we are going to discuss the existence of shortest edge-disjoint paths in G and the existence of arc/edge-disjoint paths restricted to different arc and edge sets in $\overrightarrow{G}$, the following notation will be useful.

Definition 6 (Two Disjoint Paths Relations)**.**

1. *Let $G = (V, E)$ be an undirected graph with non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$.*
   For $v, w \in V^2$, we write $v \overset{\ell}{\rightrightarrows}_E w$ if there exist edge-disjoint shortest v_i-w_i-paths w.r.t. ℓ for $i \in [2]$ in E.

2. *Let $G = (V, A \cup E)$ be a mixed graph and let $Æ_1, Æ_2$ be two subsets of arcs and edges of $A \cup E$.*
   For $v, w \in V^2$, we write $v \rightleftarrows^{Æ_1}_{Æ_2} w$ if there exist a v_1-w_1-path in $Æ_1$ and a w_2-v_2-path in $Æ_2$ which are arc/edge-disjoint.

As described above, the distance functions of the original graph G extend to the vertices of $\overrightarrow{G}$. For $i \in [2]$ and $v \in W$, $d_i(v)$ is the length of a shortest s_i-v-path in $\overrightarrow{G}$.

Lemma 7 (Paths in the Partially Oriented Expansion)**.** *Let $G = (V, E)$ be an undirected graph with non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$ and $s \in V^2$. Furthermore, let $\overrightarrow{G} = (W, E_0 \cup A_1 \cup A_2)$ be the partially oriented expansion of G w.r.t. ℓ and s.*
Then for every $t \in V^2$, we have $s \overset{\ell}{\rightrightarrows}_E t$ in G if and only if $\binom{s_1}{t_2} \rightleftarrows^{E_0 \cup A_1}_{E_0 \cup A_2} \binom{t_1}{s_2}$ in $\overrightarrow{G}$.

Proof. "⇒": Assume there exist two edge-disjoint shortest s_i-t_i-paths P_i in E_i for $i \in [2]$. Replace each edge with non-zero length in P_i by the respective oriented arc or path in the respective gadget to obtain $\overrightarrow{P}_i$ in $E_0 \cup A_i$. $\overrightarrow{P}_1$ and $\overrightarrow{P}_2$ are arc/edge-disjoint as different edges are replaced by disjoint (sets of) arcs.
"⇐": Assume there are arc/edge-disjoint s_i-t_i-paths $\overrightarrow{P}_i$ in $E_0 \cup A_i$ for $i \in [2]$. Replace the subpath of P_i within one gadget with the corresponding edge in E_i. The remaining arcs are translated directly to the respective edges in E_i. Due to the mentioned equality of distances in G and $\overrightarrow{G}$ and the fact that d_i is non-decreasing along arcs in $\overrightarrow{G}$, P_i is a shortest path in G. Any path that uses a gadget in $\overrightarrow{G}$, uses its inner arc. Therefore, P_1 and P_2 inherit being edge-disjoint from $\overrightarrow{P}_1$ and $\overrightarrow{P}_2$. □

Disjoint Paths in the Partially Oriented Expansion

Lemma 7 shows that $\overrightarrow{G}$ captures the shortest paths in G by using orientation. We will use the distances, however, to prove the main structural result. This concerns the subgraph of $\overrightarrow{G}$ that is potentially used by both paths and its weakly connected components, which are its connected components when ignoring the arcs' directions.

Lemma 8 (Structure of Partially Oriented Expansion)**.** *Let $G = (V, E)$ be an undirected graph with non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$ and $s \in V^2$. Furthermore, let $\overrightarrow{G} = (W, E_0 \cup A_1 \cup A_2)$ be the partially oriented expansion of G w.r.t. ℓ and s. Let $W = \biguplus_{j=1}^{h} W_j$ be the partition of W into the vertex sets of the h weakly connected components of the subgraph $(W, E_0 \cup (A_1 \cap A_2))$.*

Then

1. *$\overrightarrow{G}[W_j]$ is weakly acyclic for all $j \in [h]$,*
2. *sorting the components $W_j, j \in [h]$ in non-decreasing order w.r.t. the function $d_1 - d_2$ is a topological ordering of $(W, A_1)/\{W_1, \ldots, W_h\}$ and a reverse topological ordering of $(W, A_2)/\{W_1, \ldots, W_h\}$, and*
3. *$\overrightarrow{G}[W_j]$ contains arcs only from $A_1 \cap A_2$ and edges only from E_0 for all $j \in [h]$.*

Proof. In the following, we prove the three statements of the Lemma.

1.) By definition of A_1, we know that d_1 increases strictly along arcs in the set $A_1 \cap A_2$. Further, d_1 is constant on edges in E_0. Assume there is $j \in [h]$ and a (directed) cycle C in $\vec{G}[W_j]$ such that there exists $a \in C \cap A_1 \cap A_2$. Along a the distance d_1 strictly increases. However, d_1 cannot decrease along C, which yields a contradiction.
2.) Consider the function on the vertex set of $\vec{G}$. Based on the common underlying lengths in G and the definitions of A_1 and A_2, it is strictly increasing along arcs in $A_1 \setminus A_2$ and strictly decreasing along arcs in $A_2 \setminus A_1$. Opposed to that, it is constant on edges in E_0 as well as along arcs in $A_1 \cap A_2$.
3.) The function $d_1 - d_2$ is constant along all arcs $A_1 \cap A_2$ and edges in E_0. Hence, it is constant on each weakly connected component w.r.t. those arcs and edges. At the same time, the function is not constant along arcs in $A_1 \triangle A_2$. □

Algorithm 2: Dynamic Program for 2-DSPP with Non-negative Edge Lengths

Input: undirected graph $G = (V, E)$, non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$, $s \in V^2$

Output: set of pairs in V^2 that succeed s w.r.t. $\overrightarrow{\rightleftarrows}_E^{\ell}$

Construct $\vec{G} = (W, E_0 \cup A_1 \cup A_2)$ for G w.r.t. ℓ and s;
Find weakly connected components $W_1, \dots, W_h$ of the subgraph $(W, E_0 \cup (A_1 \cap A_2))$ sorted non-decreasingly w.r.t. $d_1 - d_2$;
for $j = 1, \dots, h$ **do**
    Compute $\rightleftarrows^{\vec{G}[W_j]}_{\vec{G}[W_j]}$ using Algorithm 1
end
Initialize $\rightleftarrows$ to the relation $\{(v, v) : v \in W^2\}$;
for $j = 1, \dots, h$ **do**
    Update $\rightleftarrows$ to $\rightleftarrows^{\vec{G}[W_j]}_{\vec{G}[W_j]} \circ \rightleftarrows^{\delta^-_{A_1}(W_j)}_{\delta^+_{A_2}(W_j)} \circ \rightleftarrows$
end
return $\{t \in V^2 : \binom{s_1}{t_2} \rightleftarrows \binom{t_1}{s_2}\}$

The structural result of Lemma 8 allows to use dynamic programming described in Algorithm 2 for solving Problem 2 on the partially oriented expansion. Similar to Section 2.1.4, the problem is split into two parts. First, the two arc/edge-disjoint paths problem on the weakly connected components $W_1, \dots, W_h$ of the sub-

graph $(W, E_0 \cup (A_1 \cap A_2))$ is solved by Algorithm 1. Afterward, a dynamic program is used to incorporate the results into arc-disjoint paths in $\overrightarrow{G}/\{W_1, \dots, W_h\}$ to get arc/edge-disjoint paths in $\overrightarrow{G}$.

We know that the two arc/edge-disjoint paths that we are looking for, if they exist, pass through $\overrightarrow{G}/\{W_1, \dots, W_h\}$ in opposite directions. In order to accomplish simultaneous construction of both, one of the paths is created backward. Apart from that, Algorithm 2 resembles Algorithm 1.

Theorem 9 (Algorithm 2: Correctness and Running Time)**.** *Given an undirected graph $G = (V, E)$ with non-negative edge lengths $\ell : E \to \mathbb{R}_{\geq 0}$ and $s \in V^2$, Algorithm 2 computes all successors of s w.r.t. $\overset{\ell}{\rightrightarrows}_E$ in polynomial time.*

Proof. Let $W = \biguplus_{j=1}^{h} W_j$ be the partition of W into the vertex sets of the h weakly connected components of the subgraph $(W, E_0 \cup (A_1 \cap A_2))$ as computed by the algorithm. Lemma 8.2 shows that the W_j's are sorted in a topological ordering of $(W, A_1)/\{W_1, \dots, W_h\}$ and in a reverse topological ordering of $(W, A_2)/\{W_1, \dots, W_h\}$.

For $i \in [2]$ and $j \in \{0, \dots, h\}$, set $Æ_i^j$ to be the arcs of A_i and edges of E_0 in the induced subgraph $\overrightarrow{G}\big[\bigcup_{l=1}^{j} W_l\big]$. In particular, we have $Æ_i^0 = \emptyset$. For $j \in [h]$, let $\rightleftarrows^j$ denote the relation $\rightleftarrows$ computed by Algorithm 2 after the j-th iteration. In particular, $\rightleftarrows^0$ is as defined in Line 4. We will prove by induction on $j = 0, \dots, h$ that $\rightleftarrows^j$ is equal to $\rightleftarrows^{Æ_1^j}_{Æ_2^j}$. The correctness of the algorithm then follows from Lemma 7.

The claim holds for $j = 0$, since $Æ_1^0 = Æ_2^0 = \emptyset$ by definition. Consider iteration $j \in [h]$ and assume that the claim holds for the preceding iteration.

"$\subseteq$": Let $v, w \in W^2$ such that $v \rightleftarrows^j w$. Considering Line 5 and using induction hypothesis, there exist $p, q \in W^2$ with

$$v \rightleftarrows^{Æ_1^{j-1}}_{Æ_2^{j-1}} p \rightleftarrows^{\delta^-_{A_1}(W_j)}_{\delta^+_{A_2}(W_j)} q \rightleftarrows^{\overrightarrow{G}[W_j]}_{\overrightarrow{G}[W_j]} w.$$

Lemma 8.3 guarantees that the arc and edge sets of the three relations are pairwise disjoint. As a result, $v \rightleftarrows^{Æ_1^j}_{Æ_2^j} w$ follows from Observation 3.

"$\supseteq$": Let $v, w \in W^2$ such that $v \rightleftarrows^{Æ_1^j}_{Æ_2^j} w$. Thus, there have to be a simple v_1-w_1-path P_1 in $Æ_1^j$ and a simple w_2-v_2-path P_2 in $Æ_2^j$ that are arc/edge-disjoint. Define $q_1 \in W$ to be the first vertex on P_1 in W_j, if it exists, or w_1. Let p_1 be the predecessor of q_1 on P_1 or q_1 if it is the first vertex of P_1. Similarly, let $q_2 \in W$ be the last vertex on P_2 in W_j or w_2 if it does not exist, and let p_2 be the successor of q_2 or q_2 if q_2 does not have a successor. The topological ordering of the W_j's implies that the subpaths of P_1 and P_2 prove

$$v \rightleftarrows^{Æ_1^{j-1}}_{Æ_2^{j-1}} p \rightleftarrows^{\delta^-_{A_1}(W_j)}_{\delta^+_{A_2}(W_j)} q \rightleftarrows^{\overrightarrow{G}[W_j]}_{\overrightarrow{G}[W_j]} w.$$

Finally, $v \rightleftarrows^j w$ follows by induction hypothesis.

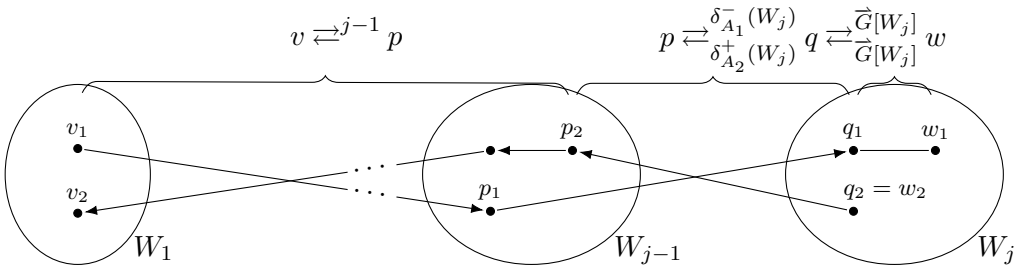


Figure 2.4: Iteration j of Algorithm 2: Relation $\rightleftarrows^j$ is built by concatenating already computed paths, pairwise different arcs to the next component, and arc/edge-disjoint paths in the next mixed component

As for the running time, finding the weakly connected components and sorting them in a topological ordering can be done in polynomial time. Computing the relations $\rightleftarrows^{\vec{G}[W_j]}_{\vec{G}[W_j]}$ also can be done efficiently by virtue of Algorithm 1. Finally, relations on V^2 have at most $|V|^4$ elements and can be composed efficiently. Therefore, the total running time of the algorithm is polynomial in the input size. □

Similar to Section 2.1.4, Algorithm 2 can be adapted to check for the existence of two vertex-disjoint shortest paths. In that case, the gadget from Figure 2.2 is not needed anymore but can be replaced by two opposite arcs.

2.1.6 Open Problems

In this section, we gave a polynomial time algorithm for 2-DSPP. The hardness of k-DSPP follows from hardness of k-DPP for variable k. However, k-DPP is solvable in polynomial time if k is constant. Therefore, the following natural questions remain open and deserve further studying.

Open Problems.

1. *Does there exist a polynomial time algorithm for 3-DSPP if $l > 0$ ($l \geq 0$)? Moreover, does there exist a polynomial time algorithm for k-DSPP for constant k if $l > 0$ ($l \geq 0$)?*

2. *What is the complexity of k-DSPP and k-dDSPP for constant $k \geq 3$ and $l > 0$?*

2.2 Cut-Degree Problems

In this section, we turn our attention to several graph bipartition problems with novel, degree-related objectives.

2.2.1 Overview

Graphs can be used in data analysis to find clusters with similarities on a set of data points. Each data point is given as a vertex and is adjacent to another data point if they share (a minimum number of) similar attributes. We are now interested in a partition of the vertex set such that vertices with similarities are assigned to the same cluster [37]. Standard measures of quality of clusters include the total number of edges crossing the induced cut, the number of edges in the respective sets or connectivities within the cluster.

However, in this section, we study graph cut problems with novel objective functions. Particularly, objectives that are depending on the degree of vertices within the cut set are investigated. Instead of minimizing the total number of edges in the cut set, we constrain our bipartitions to minimize and maximize the minimum and maximum degree of all vertices within the cut set, respectively. Additionally, we introduce a new notion of degeneracy, which only constrains one set of the partition to follow a size and degree bound. These kind of objectives are motivated from game theory such as anti-coordination games but also from data analysis resembling a new measure of quality of partitions.

2.2.2 Related Work

Probably the most famous result for maximum flow computations is the seminal Max-Flow-Min-Cut result [38, 35]. It states that the value of a maximum s-t-flow in a directed graph G equals the weight of a minimum s-t-cut. Here, the weight of a cut is determined by the sum of all capacities of arcs from the set containing s to the set containing t. These minimum cuts can be computed in polynomial time unlike maximum weight cuts [58]. Maximum cut is APX-hard and proven to be inapproximable within a factor of $\frac{16}{17}$ if $P \neq NP$, as shown by Khot et al. [60]. The same paper proves that the best known current approximation factor 0.878 by Krishnamurti and Gaur [63] is best possible under the Unique Games Conjecture. Yet another objective of interest in many applications are so-called sparsest cuts, where the objective measures the ratio of edges crossing the cut divided by the number of vertices in the smallest set. Finding the sparsest cut is also NP-hard, but there exists an $\mathcal{O}(\sqrt{\log |V|})$-approximation algorithm by Arora et al., c.f. [4].

Charikar et al. [24] study the Minimum Maximum Disagreements Problem and mention Min-Max s-t cuts as a special case of this problem. The objective for these

cuts is to minimize the maximum cut-degree. Their $\mathcal{O}(\sqrt{|V|})$ approximation algorithm for the more general Minimum Disagreements Problem is also applicable to the Min-Max-s-t Cut Problem, however, they leave the complexity of the cut problem open.

Another related problem is the Satisfactory Partition Problem. Here, one partitions the vertex set into two non-empty sets such that every vertex on the respective subgraph fulfills certain degree bounds. To be more precise, given a graph G and function f, one needs to find a partition $A \cup B$ such that $\deg_{G[A]}(v) \geq f(v)$ and $\deg_{G[B]}(v) \geq f(v)$ for $v \in A$ or $v \in B$, respectively. Gerber and Kobler [43] study the existence of such partitions for $f(v) = \lceil \frac{\deg(v)}{2} \rceil$. The variant introduced by Gerber and Kobler then is generalized by Bazgan et al. [8]. It is shown that the problem is *NP*-complete if $\lceil \frac{\deg(v)}{2} \rceil + 1 \leq f(v) \leq \deg(v) - 1$. Bang-Jensen and Bessy study yet another partitioning problem but with asymmetric degree constraints for the partition sets A and B [8]. Their partition must satisfy $\deg_{G[A]}(v) \geq k_1$ and $\deg_{G[B]}(v) \geq k_2$ for $v \in A$ or $v \in B$, respectively. They give a complete characterization of computational complexity for all values of k_1, k_2.

Furthermore, in game theory there is an interest in anti-coordination games [64]. In these games, vertices correspond to players with two options, A or B, to choose from. Their payoff is determined by the number of neighbors that have chosen a different option/set. The natural question studied is the existence of pure Nash Equilibria, corresponding to a partition such that $\deg_{G[A]}(v) \leq \frac{\deg(v)}{2}$ and $\deg_{G[B]}(v) \leq \frac{\deg(v)}{2}$ if $v \in A$ or $v \in B$, respectively. The problem of finding such a partition is also referred to as Co-Satisfactory Partition Problem and can be solved by a simple greedy algorithm. The complexity of the balanced partition size[1] version has been studied by Bazgan et al. in [9]. Besides providing hardness results, they give a 3-approximation algorithm for the number of satisfied vertices in a balanced partition.

2.2.3 Our Results

By *cut-degree* of some vertex, we denote the number of edges in the cut set incident to that specific vertex. We study computational complexity of all possible Minimum/Maximum-Minimum/Maximum-Cut-Degree variants for graph bipartition. That means, we prove the hardness of Max-Min-Cut-Degree and Min-Max-Cut-Degree and prove that Min-Min-Cut-Degree and Max-Max-Cut-Degree can be computed in polynomial time.

Furthermore, we introduce a new notion of degeneracy, i.e. (k, d)-degeneracy, and prove hardness of the decision problem in both parameters. We also give a first simple $\frac{|V|-d}{a}$-approximation algorithm for k when d is fixed and a may be any constant.

[1] A balanced partition requires equal sized sets, i.e. $|A| = |B|$.

2.2.4 Symmetric Cut-Degree Problems

First, we study objectives that are symmetric w.r.t. both sets in the partition. The objective in all four problems is determined by the maximum/minimum cut-degree among all vertices. We denote the bipartition of the vertex set by $A \cup B = V$, $A \neq \emptyset, B \neq \emptyset$, and the cut set by $C = \{\{u, v\} \in E : u \in A, v \in B\}$. The problems are defined as follows.

Definition 10 (Max-Max-Cut-Degree Problem)**.** *Find a partition $A \cup B$ with cut set C, that maximizes the maximum degree over that cut, i.e. maximizes* $\max_{v \in V} \deg_C(v)$.

Theorem 11. *Given a graph $G = (V, E)$. A Max-Max-Cut-Degree can be computed in polynomial time.*

Proof. Find a vertex $u \in V$ with $\deg(u) = \Delta(G)$. Choosing the bipartition $\{u\}, V \setminus \{u\}$ with cut set $C = \delta(u)$, then gives $\max_{v \in V} \deg_C(v) = \deg_C(u) = \Delta(G)$. □

Just like Max-Max-Cut-Degree, minimizing the minimum degree is also solvable in polynomial time.

Definition 12 (Min-Min-Cut-Degree Problem)**.** *Find a partition $A \cup B$ with cut set C that minimizes the minimum degree over that cut, i.e. minimizes* $\min_{v \in V} \deg_C(v)$.

Theorem 13. *Given a graph $G = (V, E)$. Min-Min-Cut-Degree can be computed in polynomial time.*

Proof. Observe, if the graph is not complete, i.e., if for example $\{u, v\} \notin E$, then $A = \{u\}$ and $B = V \setminus \{u\}$ defines a cut with minimum cut-degree 0.

On the other hand, if the graph is complete, $A = \{u\}$ and $B = V \setminus \{u\}$ for any $u \in V$ defines a cut with minimum cut-degree 1, which is best possible. □

More interesting and with applicable relevance are the following two problems.

Definition 14 (Max-Min-Cut-Degree Problem)**.** *Find a cut with cut set C that maximizes the maximum degree over that cut, i.e. maximizes* $\min_{v \in V} \deg_C(v)$.

In particular in anti-coordination games, this corresponds to finding a partition guaranteeing a maximum minimum payoff to all players.

Definition 15 (Min-Max-Cut-Degree Problem)**.** *Find a partition $A \cup B$ and cut set C that minimizes the maximum degree over that cut, i.e. minimizes* $\max_{v \in V} \deg_C(v)$.

This second variant is motivated from data analysis, where the maximum cut-degree of a vertex can be a measure of stability of the partition.

Unfortunately, both decision versions of the problems are *NP*-complete.

Theorem 16. *Max-Min-Cut-Degree is NP-hard. Furthermore, there does not exist a polynomial time α-approximation algorithm with $\alpha > \frac{1}{2}$ unless $P = NP$.*

Proof. We make use of a reduction from not-all-equal-3-SAT. The graph that we construct will have a maximum minimum cut-degree of 2, if and only if the instance is satisfiable. Every clause and every variable will be represented by four vertices each. The way the graph is constructed, it forces certain vertices to be in opposing sets of the bipartition. Then, by construction, we ensure that clause representing vertices need to be adjacent to at least one variable representing vertex in the other set. By that, if in the end every clause representing vertex has at least two neighbors in the other set, we satisfy the not-all-equal condition of the SAT instance.

Let us start with the formal construction. Given an instance of not-all-equal-3-SAT with variables x_i, $i \in [n]$ and clauses C_j, $j \in [m]$. Introduce four vertices a_C, a'_C, b_C, b'_C for every clause C and let $\{a_C, b'_C\}, \{a'_C, b_C\} \in E$. Furthermore, for every variable x introduce four vertices $x, x', \bar{x}, \bar{x}'$ with $\{x, \bar{x}\}, \{x', \bar{x}\}, \{x, \bar{x}'\}, \{x', \bar{x}'\} \in E$. Finally, add another four dummy vertices a, a', b, b' with $\{a, b\}, \{a', b\}, \{a', b\}, \{a', b'\} \in E$.
Additionally to the already added edges, for every clause let $\{a'_C, b'\}, \{b'_C, a\} \in E$ and for every literal $l \in C$ (e.g. $l \in \{x, \bar{x}\}$) also let $\{l, a_C\}, \{l, b_C\} \in E$. Observe that a'_C and b'_C have degree 2 and therefore any cut has minimum cut-degree at most 2.

The construction for the SAT instance $(x \vee \bar{y} \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee \bar{z})$ is depicted in Figure 2.5.

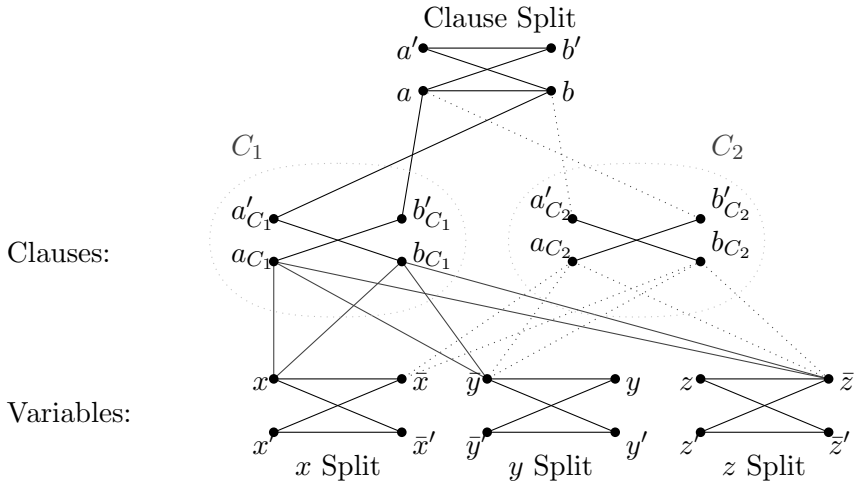


Figure 2.5: The construction of the graph for the SAT-formula $C_1 \wedge C_2$, with $C_1 = (x \vee \bar{y} \vee \bar{z})$ and $C_2 = (\bar{x} \vee \bar{y} \vee \bar{z})$.

On the bottom of the figure, a module for each variable x, y, z consisting of four vertices. Circled, the cliques representing vertices a'_{C_i}, a_{C_i} and b'_{C_i}, b_{C_i}. Finally, a, a'

and b, b' that make variable representing vertices be assigned to different sets of the partition. At the top of the figure, the module consisting of four vertices a, b, a', b' that ensures that the clause vertices a_C and b_C for each clause cannot be contained in the same set of the partition.

We claim that the graph admits a cut with minimum cut-degree exactly 2, if and only if the instance of not-all-equal-3-SAT has a valid truth assignment.

One direction is rather obvious, namely, if there exists such an assignment, w.l.o.g. setting all variables to true, the following partition has the desired minimum cut-degree:

$$A = \{a, a', a_{C_1}, \ldots, a_{C_m}, a'_{C_1}, \ldots, a'_{C_m}, x_1, \ldots, x_n, x'_1, \ldots, x'_n\},$$
$$B = \{b, b', b_{C_1}, \ldots, b_{C_m}, b'_{C_1}, \ldots, b'_{C_m}, \bar{x}_1, \ldots, \bar{x}_n, \bar{x}'_1, \ldots, \bar{x}'_n\}.$$

Note, all vertices but clause representing vertices have a cut-degree of at least two. By the choice of our partition, also all vertices a'_C and b'_C have cut-degree 2. Finally, a_C is adjacent to b'_C and an additional vertex $\bar{x}$ (representing the respective literal) that the clause must contain and is set to false, which exists by assumption that the assignment was satisfying the not-all-equal condition.

The other direction needs a little more effort. Assume, we found a partition $A \cup B$ with minimum cut-degree 2. Then, by construction of the graph, a, a' and b, b' may not be contained in the same set, i.e. we may assume w.l.o.g. that $a, a' \in A$ and $b, b' \in B$. But then, $a'_C \in A$ and $b'_C \in B$ for all clauses C. Immediately, we derive $a_C \in A$ and $b_C \in B$. At the same time, again by construction, x' and $\bar{x}'$ cannot be contained in the same set and therefore x and $\bar{x}$ cannot be contained in the same set. Finally, since every vertex must have two neighbors in the other set, every clause representing vertex a_C is adjacent to at least one variable in B different from b'_C. Similarly, every clause representing vertex b_C is adjacent to at least one vertex in A different from a'_C. This implies, assigning true to all variables x such that $x \in A$, results in every clause containing one True and one False literal. Hence, we found a valid assignment for the instance. This concludes the hardness proof.

As mentioned in the beginning, the graph admits a maximum minimum cut-degree of 2, if and only if the instance admits a valid truth assignment, and has maximum minimum cut-degree at most 1 otherwise. Thus, any α-approximation algorithm with guarantee $\alpha > \frac{1}{2}$ can be used to find a solution to the not-all-equal-3-SAT instance if it exists. □

Remark 17. One can find a partition with minimum cut-degree $\lceil \frac{\min_{v \in V} \deg(v)}{2} \rceil$ in polynomial time and every graph admits such a partition. An easy best response algorithm[2] can be used for that matter. According to Theorem 16 this algorithm

[2] Switch sets for one vertex that has less than half of its neighbors in the other set until no more such vertex exists. In each step the number of cut edges increases, implying polynomial running time.

already yields the best possible performance guarantee as $\min_{v \in V} \deg(v)$ is a natural upper bound on the cut-degree of any vertex.

Next, we study the last remaining variant, the Min-Max-Cut-Degree Problem.

Theorem 18. *Min-Max-Cut-Degree is NP-hard. Unless $P = NP$, there exists no polynomial time α-approximation algorithm for Min-Max-Cut-Degree with $\alpha < \frac{3}{2}$.*

Proof. We use a reduction from Exact-3-SAT. Given an instance of Exact-3-SAT, we construct a graph G that admits a cut $A \dot\cup B$ with cut set C such that $\max_{v \in V} \deg_C(v) = 2$, if and only if the instance of Exact-3-SAT is satisfiable.

For our construction, each variable x is represented by a module consisting of 6 vertices $d^T_{x,1}, d^T_{x,2}, d^F_{x,1}, d^F_{x,2}, x$, and $\bar{x}$ with edge set

$$\begin{aligned}&\{\{d^T_{x,1}, x\}, \{d^T_{x,1}, \bar{x}\}, \{d^T_{x,2}, x\}, \{d^T_{x,2}, \bar{x}\}, \{d^T_{x,1}, d^T_{x,2}\},\\ &\{d^T_{x,1}, d^F_{x,1}\}, \{d^T_{x,2}, d^F_{x,2}\},\\ &\{d^F_{x,1}, x\}, \{d^F_{x,1}, \bar{x}\}, \{d^F_{x,2}, x\}, \{d^F_{x,2}, \bar{x}\}, \{d^F_{x,1}, d^F_{x,2}\}\}.\end{aligned}$$

The module is designed in such a way that both vertices x and $\bar{x}$ have to be separated by any cut with $\max_{v \in V} \deg_C(v) = 2$.

For each variable x and clause C_i, introduce a vertex x_{C_i} and $\bar{x}_{C_i}$.

Let these vertices just introduced (for each variable and its respective negation), i.e. $\{x_{C_i}\}_i$, x, and $\{\bar{x}_{C_i}\}_i$, $\bar{x}$, form cliques, respectively [3].

Also, introduce a vertex C_i for each clause C_i and let $\{C_i, x_{C_i}\} \in E$ or $\{C_i, \bar{x}_{C_i}\} \in E$, if $x \in C_i$ or $\bar{x} \in C_i$.

Finally, let D^T and D^F be two cliques of size 5. We let $\{d^T_{x,i}, d^T\} \in E$ for all $d^T \in D^T$ variables x and $i \in \{1, 2\}$. Furthermore, $\{d^T, C_i\} \in E$ for all $d^T \in D^T$ and clauses C_i.

Similarly, $\{d^F_{x,i}, d^F\} \in E$ for all $d^F \in D^F$ variables x and $i \in \{1, 2\}$.

The graph is depicted in Figure 2.6. Keep in mind that any cut of a clique of size 5, i.e. K_5, has a maximum degree on that cut of at least 3. Thus, for any cut $A \dot\cup B$ with cut-degree at most two, either the entire clique is contained in A or in B.

It is easy to see that the graph admits a cut with maximum cut-degree of 2 if the instance is satisfiable. For that, consider the set B that contains D^F, $d^F_{x,1}, d^F_{x,2}$. It also contains x and x_{C_i} for all clauses C_i the literal x is contained in, if x is set to false. Otherwise, if x is set to true, let $\bar{x} \in B$ and $\bar{x}_{C_i} \in B$ for all such clauses C_i. The remainder of the vertex set forms the partition set A. Also see Figure 2.6, where x is set to true and y, z are set to false. The dashed line describes the cut, in particular, it intersects all cut edges.

[3] Introducing a literal for every clause directly implies that this clique is of size at least 5

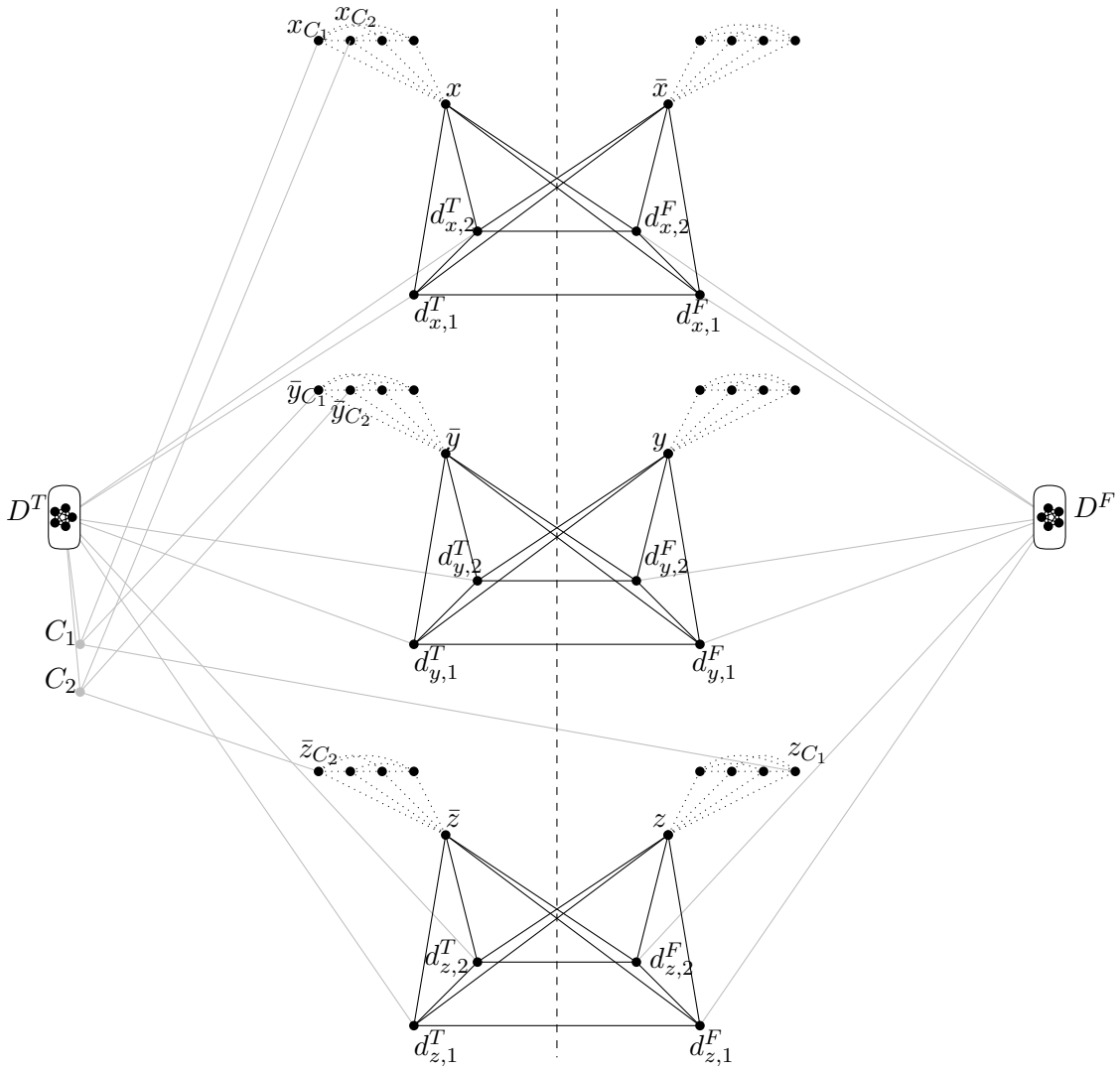


Figure 2.6: In bold black, a module for each variable x, y, z. Dotted, the cliques representing possible literals for clauses. Finally, D^T and D^F dummy cliques, to ensure a correct truth assignment. Dashed, a cut with maximum degree 2 if the instance is satisfiable. The instance described here is $C_1 \wedge C_2$ with $C_1 = (x \vee \bar{y} \vee z)$, $C_2 = (x \vee \bar{y} \vee \bar{z})$ and variables x, y, z.

We claim that such a partition with cut-degree at most 2 exists, if and only if the instance is satisfiable. Therefore, assume the instance is not satisfiable but the graph admits a cut $A \cup B$ with cut-degree at most 2.

First of all, every such cut gives rise to a proper truth assignment to the variables. The set A may only contain one of the vertices x or $\bar{x}$ for each of the variables. If it contains both, it must contain all four $d^T_{x,1}, d^T_{x,2}, d^F_{x,1}, d^F_{x,2}$ in order to fulfill the maximum degree 2 constraint. This is because each of these four vertices together with x and $\bar{x}$ is adjacent to 4 other vertices in the set. Thus, in order to fulfill a maximum cut-degree of two, each vertex must be in the same set as at least two of its neighbors, but that is impossible. However, if the set A it contains all four, $d^T_{x,1}, d^T_{x,2}, d^F_{x,1}, d^F_{x,2}$, it also must contain D^T and D^F and hence, all other vertices C_i representing a clause. Consequently, it also contains all vertices $d^T_{y,1}, d^T_{y,2}, d^F_{y,1}, d^F_{y,2}$ and thus also y and $\bar{y}$, as well as $y_{C_i}, \bar{y}_{C_i}$ for every variable y and clause C_i. But this implies that $A = V$, which contradicts, that $A \cup B$ is a proper partition, i.e. that $B \neq \emptyset$.

As we have seen above, either $D^T \in A$ and $D^F \in B$ or vice versa. So, w.l.o.g., assume $D^T \in A$.

Next, we claim that we find a valid truth assignment by setting all literals to true that are contained in the same set as D^T, i.e. are contained in A. Therefore let $x = 1$ if $x \in A$ and $x = 0$ if $\bar{\in} A$. Since $D^T \in A$, additionally $C_i \in A$ for all clauses C_i. Because the cut induced by $A \cup B$ has maximum cut-degree two, there must exist at least one literal for each clause that is also contained in A, which implies that the corresponding vertex x or $\bar{x}$ is also contained in A. Thus, the truth assignment indeed satisfies all clauses, which finishes the proof.

Observe, any α-approximation algorithm with guarantee $\alpha < \frac{3}{2}$ would find a partition with cut-degree 2. Therefore, unless $P = NP$, no such algorithm can have polynomial running time. □

Remark 19. Note, one cannot use the same approximation algorithm as for Max-Min-Cut-Degree. Even if we adapted the strategy and were able to achieve $\geq \lceil \frac{\deg(v)}{2} \rceil$ neighbors in the same set, this does not yield any approximation guarantee as $\lfloor \frac{\deg(v)}{2} \rfloor$ can be large for some vertices.

The best known approximation algorithm is due to Charikar et al. [24] and has an approximation factor of $\mathcal{O}(\sqrt{|V|})$.

2.2.5 An Asymmetric Cut-Degree Problem — A Generalization of Degeneracy

In this subsection, we study, in the broadest sense, a cut problem with an asymmetric objective function, constraining the maximum degree only for one set of the partition. With this setting in mind, we introduce a generalized notion of graph degeneracy. First, recall the definition of degeneracy.

Definition 20. *A finite graph $G = (V, E)$ is called d-degenerate if every subgraph contains a vertex of degree less than or equal to d.*

This definition has a nice equivalent formulation:
Namely, a graph $G = (V, E)$ is d-degenerate if and only if it has an ordering of the vertices on a line such that each vertex has at most d neighbors to its left, that means there exists an ordering π such that

$$|\{j \in N_G[i] : \pi(j) < \pi(i)\}| \leq d \qquad \forall i \in [n].$$

Such an ordering is called Erdős-Hanjal sequence. Also, this definition can be extended to directed graphs by replacing degree by in-degree.

In the following, we generalize the definition of degeneracy, which shall be used later, e.g. in Chapter 3.

Definition 21. *A graph G is called (k, d)-degenerate if every subgraph $H \subseteq G$ contains a set $S \neq \emptyset$ of at most k vertices such that every vertex within S has at most d neighbors in $H \backslash S$. Furthermore, let*

$$k_d(G) := \min\{k : G \text{ is } (k, d)\text{-degenerate}\}.$$

Again, this notion of degeneracy can be extended to directed graphs if we require that each vertex $i \in S$ has at most d vertices in $H \backslash S$ that have an arc towards i. Also, there is a nice equivalent formulation:

Equivalently, a graph $G = (V, E)$ is called (k, d)-degenerate if there exists a weak order $\pi \colon V \to [n]$ of the vertices such that

$$|\{j \in N_G[i] : \pi(j) < \pi(i)\}| \leq d \qquad \forall i \in [n], \tag{2.1}$$

and the weak order is of width at most k[4], that is, the maximal number of vertices with the same label is at most k. Which is formally

$$\max\{|S_i| : S_i = \{j \in V : \pi(j) = i\}\} \leq k.$$

Definition 22. *A graph $G = (V, E)$ is called exact-(k, d)-degenerate if we can find a weak order $\pi : V \to [n]$ fulfilling (2.1) together with*

$$|\{j \in V : \pi(j) = i\}| \in \{0, k\} \quad \forall i \in [n] \backslash \{l\} \quad \text{and} \quad |\{j \in V : \pi(j) = l\}| \leq k$$

for some $l \in [n]$.

Naturally, every graph that is exact-(k, d)-degenerate is (k, d)-degenerate. However, the reverse implication does not hold true.

[4] Also meaning that the longest antichain in the respective partial order is of length at most k.

Our notion of degeneracy falls into the framework of asymmetric partitions like studied in [8] as mentioned earlier. But our notion of degeneracy is also somewhat related to the notion of (k, d)-partitionable graphs, that says, there exists a partition of the vertex set into k sets, each inducing a d-degenerate graph which was studied for example by Simões-Pereira in [80]. A simple observation is that every (k, d)-degenerate graph is (k, d)-partitionable.

While it is possible to decide in polynomial time whether a given graph is d-degenerate, the following theorem states that deciding if a given graph is (k, d)-degenerate for some variable k is a *NP*-complete problem.

Theorem 23. *Given G and k, it is NP-complete to decide whether G is (k, d)-degenerate for $d \geq 2$.*

Proof. In our proof, we stick to undirected graphs. Given a weak order $\pi\colon V \to [n]$ of width at most k. We can check in polynomial time if condition (2.1) is fulfilled. Hence, the problem is in *NP*.

We shall now prove the theorem by reducing the *NP*-complete problem Exact-l-SAT to our problem. Even though our construction works for general l, we set $l = 3$ (else, we would have to introduce many more dummy vertices and computations would be more complex). In the remainder of the proof, we abbreviate Exact-3-SAT by 3-SAT. So, given an instance of 3-SAT with n variables and m clauses we construct a graph on $m+3n+1$ vertices and $5n+\binom{n+m}{2}+3m$ edges, that is $(m+2n, 2)$-degenerate if and only if the given instance is satisfiable. First of all, we assume that in each clause there are three distinct variables. If there were clauses containing the same literal twice, we split that clause into two clauses using one dummy variable. Also, if there was a clause containing both, a variable and its complement, this clause would always be fulfilled and could be neglected.

Next, we give a formal construction of our graph

1. Each of the n variables is represented by three vertices, namely x_i^t, x_i^f and a dummy vertex x_i. Those three vertices form a triangle, i.e. $x_i^t \sim x_i^f \sim x_i \sim x_i^t$ for all $i \in [n]$.

2. There is one dummy vertex d that is adjacent to all variable-representing vertices x_i^t, x_i^f and x_i for all $i \in [n]$.

3. Each clause j of the m clauses is represented by one vertex, namely c_j. If clause j contains a variable i, the vertex c_j is adjacent to x_i^t, if the clause contains the complement of the variable i, the vertex c_j is adjacent to x_i^f. (Hence, every vertex c_j is, by now, adjacent to exactly 3 variable-representing vertices.)

4. The union of the set of dummy vertices x_i for $i \in [n]$ and the set of clause-representing vertices c_j for $j \in [m]$ forms a clique.

Figure 2.7 depicts the constructed graph for a 3-SAT instance.

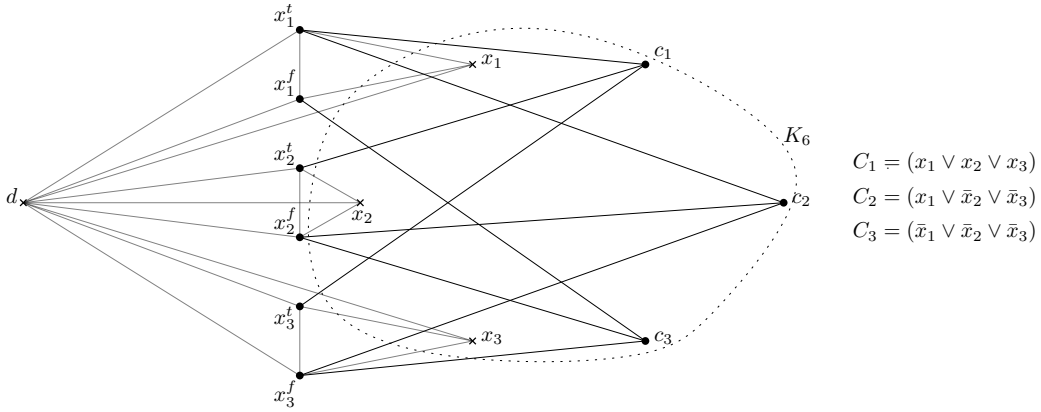


Figure 2.7: Graph corresponding to the instance of 3-SAT: $(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)$ with three variables x_1, x_2, x_3 and three clauses $C_1 = (x_1 \vee x_2 \vee x_3), C_2 = (x_1 \vee \bar{x}_2 \vee \bar{x}_3)$, and $C_3 = (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)$.

An easy computation gives the vertex number and the edge number as mentioned above.

Claim 1. The constructed graph is not $(m + 2n - 1, 2)$-degenerate.

Let us assume the contrary, i.e. there exists a set S of at most $m + 2n - 1$ vertices such that each vertex has at most 2 neighbors in $V \backslash S$.

First of all, observe that the dummy vertex d cannot be contained in S. If it was, all but two variable-representing vertices would have also to be contained in S. Since the vertices in $\bigcup_j c_j \cup \bigcup_i x_i$ form a clique, again we have that all but two vertices from this set must be contained in S. We get that the size of the set is at least $1 + (3n - 2) + m - 2 = 3n + m - 3 > m + 2n - 1$, since $n \geq 3$.

Next, we claim that clause-representing vertices cannot be part of S.
If there was one clause-representing vertex contained in S, this, just like before, implies that all but at most 2 vertices in $\bigcup_j c_j \cup \bigcup_i x_i$ must also be contained in the set S.

Assume next, all but two vertices in $\bigcup_j c_j \cup \bigcup_i x_i$ were contained in S. Then, at least one of these vertices would have to be a clause-representing vertex. This is due to the fact that otherwise the vertices $x_i \in S$ would have three neighbors outside S (the two vertices and d). On the other hand if there were two clause-representing vertices outside the set together with vertex d, again each vertex $x_i \in S$ would have 3 neighbors outside S. So assume $x_j, c_l \notin S$. Then, again since $d \notin S$ all $x_i \in S$ would have three neighbors outside S we derive a contradiction.

Therefore, all but at most 1 vertex in $\bigcup_j c_j \cup \bigcup_i x_i$ must also be contained in the set S.

If all vertices of this set were contained in S, every variable representing vertex can have at most one neighbor not contained in S different from d. We get that $|S| \geq m + n + n = m + 2n > m + 2n - 1$. Again, this yields a contradiction.

Finally, assume that exactly one vertex in $\bigcup_j c_j \cup \bigcup_i x_i$ is not contained in S. If this vertex was some x_j, as d is not part of S, all other variable-representing vertices x_i for $i \in [n] \backslash \{j\}$ would already have two neighbors outside S, and, therefore, $x_i^t, x_i^f \in S$ for all $i \in [n] \backslash \{j\}$. We get that $|S| \geq m + n - 1 + 2(n-1) = m + 3n - 3 > n + 2n - 1$ since $n \geq 3$. So this vertex outside S must be a clause-representing vertex c_j. But then again, as d is not part of S, all variable-representing vertices x_i for $i \in [n]$ would have already two neighbors outside S, and therefore $x_i^t, x_i^f \in S$ for all $i \in [n]$. We get that $|S| \geq m - 1 + n + 2n = m + 3n > n + 2n - 1$. Yet, another contradiction.

We have seen so far, $c_j \in S$ and $d \in S$ contradict the properties of the set S.

Finally, assume a variable-representing vertex x_i, x_i^f or x_i^t for some $i \in [n]$ was contained in S, then at least one of the remaining two vertices representing variable i is also contained in S. This is due to the fact that vertex d cannot be part of S.

As we have seen before, $x_i \in S$ directly implies that all but at most 2 vertices in $\bigcup_j c_j \cup \bigcup_i x_i$ are contained in S, which led to a contradiction. Therefore, x_i cannot be part of S.

All what is left is that at least one vertex x_i^t or x_i^f for some $i \in [n]$ is contained in S. Then, as x_i and d cannot be part of S, we immediately have that $x_i^t, x_i^f \in S$. But then, since there is at least one clause j containing one of the two literals, a vertex c_j, representing this clause, must also be contained in S. Otherwise the corresponding variable-representing vertex would neighbor the vertices d, x_i and c_j outside the set S. However, as we again have at least one clause representing vertex in S, we end up with a contradiction.

So, the set S cannot contain the vertices d nor x_i, x_i^t, x_i^f for any $i \in [n]$ nor c_j for any $j \in [m]$. Therefore, such a set S does not exist, finally contradicting that the graph was $(m + 2n - 1, 2)$-degenerate.

Claim 2. The constructed graph is $(m + 2n, 2)$-degenerate if and only if the instance of 3-SAT is satisfiable.

Let us assume the instance of the given 3-SAT formula is satisfiable and, w.l.o.g., in such a truth assignment each variable is set to true. Then, we can remove the following set from G

$$S = \Big(\bigcup_j c_j\Big) \cup \Big(\bigcup_i (x_i^t \cup x_i)\Big),$$

where each vertex in this set has at most two neighbors outside the set. Every vertex x_i or x_i^t is adjacent to x_i^f and d outside the set only. Also, every clause representing vertex is adjacent to at most two vertices x_i^f because we chose a truth assignment implying that it contains at most two negative literals. Next, observe that $|S| = m + 2n \geq n + 1$.

Thus, we have found a weak order of the vertices, only using two labels. Given a vertex i, we let $\pi(i) = 1$ if $i \in S$ and $\pi(i) = 0$ else. The longest antichain/largest label class is of the size of S, which is $m + 2n$, the graph is thus $(m + 2n, 2)$-degenerate. This concludes one direction.

Assume the graph is $(m+2n, 2)$-degenerate. Then there is some set S of size at most $m + 2n$ such that each vertex has at most two neighbors in the remaining graph. As we have seen in the proof of Claim 1, this is only possible if S contains all vertices in $\bigcup_j c_j \cup \bigcup_i x_i$. But then, d cannot be part of S, so for each variable, there exists at least one additional variable-representing vertex that is contained in S. We now find a truth assignment by setting the variables contained in this set S to true, i.e. the variable i is set to true if $x_i^t \in S$ and to false otherwise. Note, both variable-representing vertices x_i^t and x_i^f cannot be contained in S at the same time. Since every vertex in S has at most two neighbors outside the set, every clause contains at most two literals that are set to false.

Combining Claim 1 and Claim 2, we found a polynomial time reduction from 3-SAT to our decision problem, which proves the theorem.

For our example instance of 3-SAT, with $m = 3$ and $n = 3$, the graph is $(9, 2)$-degenerate and a corresponding decomposition can be seen in Figure 2.8. In the example we have

$$S = \{c_1, c_2, c_3, x_1, x_2, x_3, x_1^f, x_2^f, x_3^t\},$$

which is of size 9 and every vertex within the set has at most 2 neighbors outside the set S. Also, the rest of the graph has less than 9 vertices. This implies that the instance is satisfiable and a truth assignment is given by $x_1 = 0, x_2 = 0, x_3 = 1$.

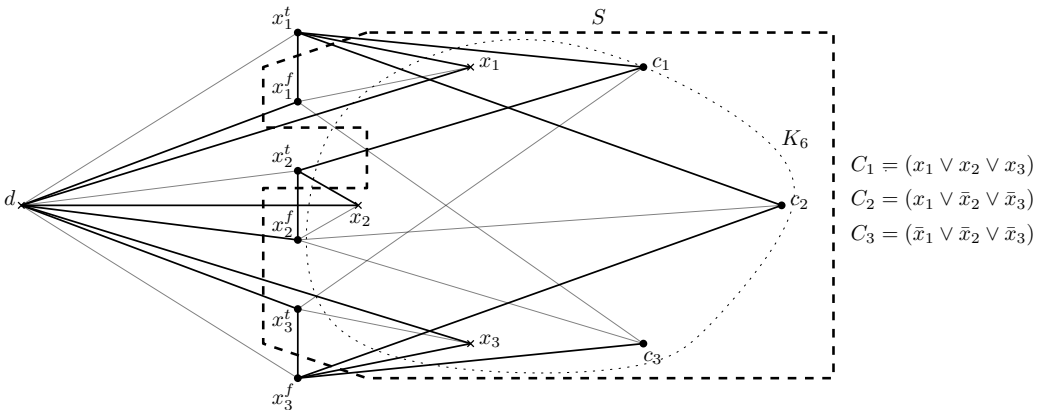


Figure 2.8: A $(9, 2)$-decomposition of the graph. Labeling all vertices outside the marked set S with label 0 and all vertices inside the set with label 1.

□

As we have seen in Theorem 23, the decision problem is *NP*-hard if $d \geq 3$. However, given d, one can use the following algorithm to compute an approximate value of k such that the graph is (d, k)-degenerate.

Theorem 24. *Algorithm 3 computes a bipartition $V_1 \dot\cup V_2$ with $|N(v) \cap V_1| \leq d$ for all $v \in V_2$ and runs in $\mathcal{O}(|V|^{a+2})$ time.*

Proof. Observe, that the partition computed in line 1 fulfills the degree constraint and therefore is a feasible solution. Algorithm 3 therefore always finds a solution. It takes $\binom{|V|}{a}$ many loops, and each loop requires $|V|$ degree checks. Therefore, Algorithm 3 runs in

$$\binom{|V|}{a}|V|^2 \leq |V|^{a+2}$$

time. □

Algorithm 3: Approximation Algorithm for k_d

Input: Graph $G = (V, E)$, parameter d and a

Output: Bipartition of $V = V_1 \dot\cup V_2$ such that all vertices in V_2 have at most d neighbors in V_1

```
1  Let k = |V| − d, V_1 any set of size d and V_2 = V \ V_1;
2  for each subset S ⊂ V of size at most a do
3      if all vertices in S have at most d neighbors in V \ S and |S| ≤ k then
4          Let k = |S|, V_2 = S and V_1 = V \ S
5      end
6      if all vertices in V \ S have at most d neighbors in S and |V \ S| ≤ k then
7          Let k = |V \ S|, V_1 = S and V_2 = V \ S
8      end
9  end
10 return V_1, V_2
```

This algorithm can be used to obtain an approximation algorithm for the general problem, i.e. finding a weak order of width k for given parameter d fulfilling the degree bounds (2.1).

Theorem 25. *Given a graph G and a parameter d. There exists an $\frac{|V|-d}{a}$-approximation algorithm for $k_d(G)$ for every constant a.*

Proof. Iterate Algorithm 3 on the output sets V_1 until no more changes occur.

It is easy to see, that if $k \leq a$, we find an optimal weak order applying Algorithm 3. Thus we may assume $k \geq a$. But then, the bipartition computed by Algorithm 3 is a weak order of width $n - d$, which gives the desired approximation guarantee. □

The rest of this subsection contains some additional properties of the new notion of degeneracy, some of which shall be useful later.
Next, we give some simple facts for arbitrary graphs.

Lemma 26. *Given an arbitrary graph G.*

1. *$k_0(G)$ equals the size of the largest connected component, hence, $k_0(G) = n$ if G is connected.*
2. *G is d-degenerate if and only if $k_d(G) = 1$.*
3. *If G is a (k, d)-degenerate graph, i.e. $k_d(G) \leq k$, then G is also $(k - i, d + i)$-degenerate for all $i \in [k - 1]$. This implies that G is $(d + k - 1)$-degenerate.*
4. *If G is a D-degenerate graph but not $(D - 1)$-degenerate, then $k_d(G) \geq D - d$.*
5. *If G is (k, d)-degenerate, then G is $(k + d)$-colorable.*
6. *If G is (k, d)-degenerate, then G is $(k - 1, d + 1)$-colorable.*

Proof. 1.) It is immediate, that two connected vertices i, j must obtain the same label. But the result also holds true for directed graphs, replacing connectivity by strong connectivity. Given strongly connected i and j that got different labels, either the i-j-path or the j-i-path must contain an arc in the opposite direction, that is an arc from a vertex to another vertex that has a smaller label. On the other hand, partitioning the set of vertices into the strongly connected components gives rise to a weak order fulfilling the degeneracy conditions. This is due to the fact that the graph obtained by contracting all strongly connected components is acyclic.

2.) This fact is clear by definition of both notions of degeneracy.

3.) Consider the weak order, where each label class is of size at most k and every vertex is adjacent to at most d vertices with smaller label. Iteratively removing one vertex at a time from each label class j of size $s > k - i$ and assigning it the new label $j - 1/s$ (shift all labels at the end to obtain integer labels again). That way each vertex which had the same label as the removed vertex now has at most one additional neighbor. Since this is done for each label class at most i times, degrees increase by at most i. The graph is thus $(k - i, d + i)$-degenerate.

4.) This simply follows from 3. as if $k_d(G) \leq D - d$, the graph would be $(d + (D - d) - 1) = (d - 1)$-degenerate.

5.) By 3. we have that G is $(d + k - 1)$-degenerate and hence $(k + d)$-colorable.

6.) Recall that a graph is (l, χ)-colorable if there exists a coloring of the vertices using at most χ colors such that each vertex is adjacent to at most l vertices that received the vertex's color [27]. Our weak order directly implies the existence of such a coloring as every vertex has at most d neighbors to its left, we can iteratively color each vertex with one unused color. Since in each antichain there are at most $k - 1$ vertices which might receive the same color, each vertex has at most $k-1$ vertices in its neighborhood that received the same color. □

Finally, we close this subsection with another remark on computational complexity of the parameter k_d.

Remark 27. If $d = 0$, the least k such that the graph is (k, d)-degenerate would be the size of the largest (strongly) connected component, and hence could be computed in polynomial time. Also, observe that for any fixed constant k we can check in polynomial time whether a given graph is (k, d)-degenerate by simply iteratively checking all subsets of size at most k as those are polynomially many in the number of vertices. The case $d = 1$ remains open as our reduction only works for $d \geq 2$ since 2-SAT can be solved in polynomial time.

2.2.6 Open Problems

For the Max-Min-Cut-Degree problem, there exists an $\frac{1}{2}$-approximation algorithm, c.f. Remark 17, which is best possible if $P \neq NP$. For the Min-Max-Cut-Degree Problem, however, unless $P = NP$, no polynomial time algorithm can have an approximation ratio of $< \frac{3}{2}$. The best known approximation algorithm has a guarantee of $\mathcal{O}(\sqrt{|V|})$ (see Remark 19), leaving a large gap for improvement.

Concerning our new notion of graph degeneracy, we gave a proof that $k_d(G)$ cannot be computed in polynomial time unless $P = NP$ for $d \geq 2$. Also, we have seen that $k_0(G)$ can be computed in polynomial time, however, the case $d = 1$ remains open. Additionally, better approximation algorithms for k_d might exist and would be of practical relevance c.f. Section 3.3. Last but not least we give some open problems deserving further research.

Open Problems.

1. *Derive better approximation algorithms for the Min-Max-Cut-Degree problem.*
2. *Since the instances are more restrictive, can the methods of the $\sqrt{|V|}$-approximation algorithm of Charikar et al. [24] for Minimum Maximum Disagreements be adopted to obtain better approximation algorithms for the Min-Max-Cut-Degree problem?*
3. *Can $k_1(G)$ be computed in polynomial time?*
4. *Does there exist a better approximation algorithm than Algorithm 3 for $k_d(G)$?*

2.3 Mitigating the Impact of Security Inspections on Regular Journeyers

We consider a weighted, possibly directed, graph with a number of invaders and journeyers that both would like to travel through the network. From a security point of view, one would like to prevent the invaders from reaching their destination by installing security checks on some of the edges without impacting the journeyers. We pose two models that are applicable to several real-world problems, where in each model we want to deter the invaders, i.e. install at least one security check on each path from their starting point to their respective destination. Therefore, the set of edges with security checks induces a (multi-terminal) cut.

The first model aims to minimize the time delay for all journeyers caused by these security checks, namely minimize the sum of traveling times for journeyers. In this model, security checks increase the traveling time of the respective edge they are placed on. We prove several hardness results for this problem and present a polynomial time algorithm for one passenger and a constant number of invaders. However, this algorithm is only applicable to instances where traveling times only increase by some constant or traveling times increase proportionally.

The second model aims to maximize the journeyer flow from a given set of sources to a given set of sinks, subject to deterring all invaders. Here, security checks decrease journeyer capacities along edges. Again, we state a hardness and an inapproximability result for this model.

This section is based on joint work with Jannik Matuschke [51].

2.3.1 Overview

We are dealing with possibly directed graphs $G = (V, A)$ with vertex set V and arc set $A \subseteq V^2$. Throughout this section, there will be starting vertices as well as destination vertices for both, invaders and journeyers. We shall use the terms starting vertex and source or destination vertex and sink interchangeably. The main mission in all studied variants of the following problems is to prevent the invaders from reaching their destination without passing a security check. Therefore, we call an invader pair to be deterred if every (directed) path in G, connecting the source and sink of this invader, contains at least one security check.

Of course, one could deter all invaders by placing security checks on all arcs, but this might cause delays to journeyers or decrease the amount of journeyers that can travel through the network.

The impact of security checks is modeled in two different ways. In Section 2.3.2, security checks increase travel times and we aim to minimize the sum of shortest paths of all journeyers. In Section 2.3.3, security checks decrease capacities of arcs and the objective is to maximize passenger flow.

Related Literature

These two kinds of objectives for (multi-terminal) cuts have not yet been studied in the literature. For related literature on cuts with different objectives, we refer to Section 2.2.2. However, one can also interpret the problems introduced as a shortest path/multi commodity flow problem with extra constraints. In particular disjoint paths fall into this framework. Especially, computational complexity results of our introduced problems can be related to disjoint paths. Therefore, we also refer to Section 2.1.2.

But also very related problems like resource[5] constrained shortest paths have been studied in the past [10, 74, 69]. Here, the graph is equipped with distance and weight function(s). The task is to find a shortest path not exceeding the resource budgets. However, this problem is *NP*-hard even for one resource constraint [54].

2.3.2 Paths

In this subsection, we study the problem of placing security checks on a graph in such a way that invaders cannot reach their destination from their respective starting point without traveling along an edge/arc with a security check. However, security checks increase the time needed to travel along an edge/arc for journeyers. Our objective is to minimize the sum of traveling times for a given set of journeyer sources and corresponding destinations. More formally, given a graph $G = (V, E, \tau)$ with edge/arc transit times τ. Each invader and journeyer is represented by a tuple consisting of a source and sink vertex, i.e. let $I \subseteq V^2$ be the set of *invaders* and $J \subseteq V^2$ be the set of *journeyers*. A security check allocation $C \subseteq E$ is said to *deter* an invader $(s_i, t_i) \in I$ if every s_i-t_i-path in G contains at least one edge/arc in C. Furthermore, let transit times on edges/arcs increase when a security check is placed, i.e.

$$\tau_e^c = \begin{cases} \tau_e & \text{for all } e \notin C. \\ \tau_e + \lambda_e & \text{for all } e \in C \end{cases}$$

with $\lambda_e \geq 0$ for all $e \in E$.

Definition 28 (TESC)**.** *The problem of deciding whether there exists a security check allocation $C \subseteq E$ on the edges of G in such a way that all invaders are deterred and the sum of distances w.r.t. τ^c of the respective shortest paths for each pair of journeyers does not exceed T, is called* Time Efficient Security Checks. *We denote it by* (I, J)-Tesc .

We slightly abuse notation and write (l, m)-Tesc for $l, m \in \mathbb{N}$ to denote an instance with l invaders and m journeyers. An example input of $(1, 1)$-Tesc is given in

[5] Sometimes also referred to as weight constrained shortest paths

Figure 2.9.

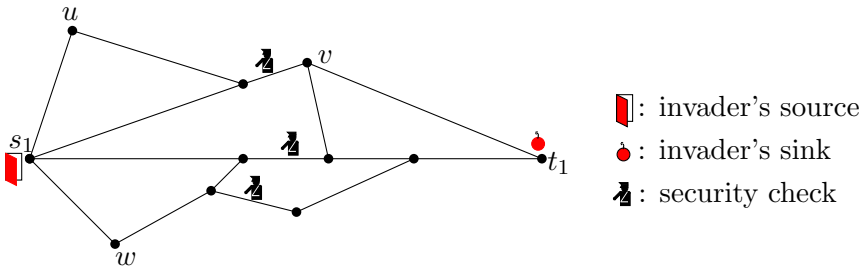


Figure 2.9: Example on an undirected graph. The invader is formed by the vertex pair (s_1, t_1). Journeyers could be e.g. $J_1 = (u, w), J_2 = (v, w)$. The placement is obviously deterring the only invader as removing all edges with security checks disconnects s_1 from t_1. Assuming unit weights on edges ($\tau \equiv 1$) and an increase of 1 for edges with security checks ($\tau^c \equiv 1 + 1$), the cost (sum of shortest path lengths for all journeyers) of the depicted security check allocation is $2 + (3 + 1) = 6$.

The Directed Case

Theorem 29. *Given a directed graph $G = (V, A, \tau)$.*

i) (I, J)-TESC *is NP-complete, even if* $|I| = |J| = 1$*, the invader and the journeyer agree, and* $\lambda_a = \lambda > 0$ *for all but one* $a \in A$*.*

ii) (I, J)-TESC *is NP-complete, even if* $|I| = |J| = 1$*, the invader and the journeyer agree, and* $\tau^c_a = \alpha\tau_a$ *for all* $a \in C$ *and* $\alpha > 1$*.*

iii) $(1, J)$-TESC *is NP-complete, if* $|J| \geq 2$ *and* $\lambda_a = \lambda$ *for all* $a \in A$ *with* $\lambda > 0$*.*

Proof.
i) The problem is obviously in *NP*, since, given a security check allocation one can check all — at most $\binom{n}{2}$ — invaders which must be deterred and compute the path lengths of journeyers efficiently. For hardness, we use a reduction from the *NP*-complete two vertex disjoint directed path problem (2-dDPP) [39]. That is, given two pairs of vertices on a directed graph, find two vertex disjoint paths connecting one pair of vertices respectively. Let $G = (V, A)$, (s_1, u) and (v, t_1) be an instance of the two disjoint path problem. We now prove that we can use $(1, 1)$-TESC to decide on the existence of these two disjoint paths. W.l.o.g assume that there is a directed arc from u to v (none of the paths would ever use this arc), we now let $G' = (V, A, \tau)$, where $\tau = 0$ and $\lambda_a = 1$ for all arcs $a \in A \setminus \{(u, v)\}$ and $\lambda_{(u,v)} = 0$. Finally, let (s_1, t_1) be both, an invader pair as well as a journeyer pair (c.f. Figure 2.10).

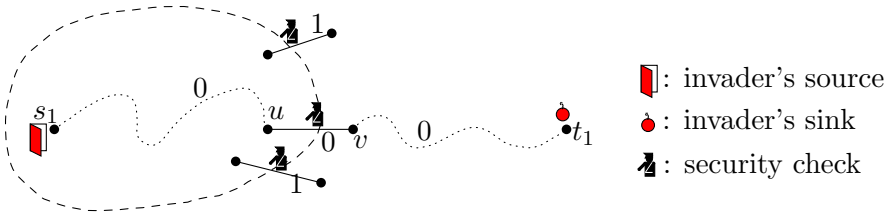


Figure 2.10: Construction of G'.

Now, we claim that there exists a security check allocation deterring the invader such that the shortest path for the journeyer is of length 0 if and only if there exist two disjoint paths connecting the pairs of vertices respectively.

Observe, if there are two disjoint paths connecting s_1, u and v, t_1 one can easily separate s_1 and t_1 by placing security checks on all arcs not contained in either of the paths as well as on the arc (u, v). Placing security checks in this way gives rise to a path of length 0 connecting s_1 and t_1.

Since any path connecting s_1 and t_1 must cross a security check at some point, the path of length 0 must cross a security check on the arc (u, v). Also, the path may not cross any other security check. Therefore, and due to the fact that s_1 and t_1 are separated by security checks, the path is split into two disjoint parts, the part connecting s_1 and u and the part connecting v and t_1, which are two disjoint paths connecting both pairs of vertices.

ii) The proof is analogous to i), where, using the same notation and reduction, we let $\tau_a = 1$ for all arcs $a \in A$, $\alpha_a = n^2$ for all $a \in A \setminus \{(u, v)\}$, and $\alpha_{(u,v)} = 1$.

iii) For the proof, we again use a reduction from 2-dDPP: Given a directed unweighted graph and two pairs of vertices $(s_1, t_1), (s_2, t_2)$, decide whether there exist two vertex disjoint paths connecting both pairs of vertices. We consider the same graph and let 0 be the traveling time on all arcs and let the traveling time increase constantly by $\lambda = 1$ if a security check is installed. We let (s_1, t_2) be one invader and let $(s_1, t_1), (s_2, t_2)$ be two journeyers (c.f. Figure 2.11).

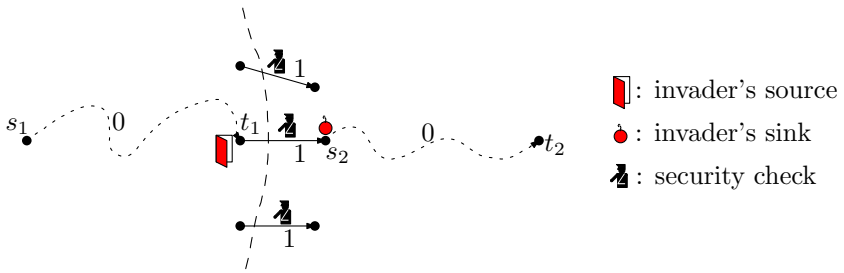


Figure 2.11: Schematic visualization of the constructed instance.

We have that there exists a security check allocation deterring (s_1, t_2) with path lengths 0 each, for a shortest $s_1 - t_1$-path and shortest $s_2 - t_2$-path, if and only if there exist two vertex disjoint paths connecting both, s_1 and t_1 as well as s_2 and t_2.
If these two disjoint paths exist, simply place security checks on all arcs not contained in either of the paths.
If there exists a security check allocation deterring s_1 and t_2 such that both paths, from s_1 to t_1 and s_2 to t_2, are of length 0, none of these paths can ever cross a security check and hence, both paths must be disjoint. □

Remark 30. Note that the construction in the proof of Theorem 29i) actually proves that there cannot exist any polynomial time approximation algorithm unless $P = NP$, as an optimum solution has value 0.

Theorem 31. *Given an acyclic graph $G = (V, A, \tau)$ and $\tau_a^c = \tau_a + \lambda$ for some $\lambda > 0$. Then $(I, 1)$-*Tesc *is NP-complete.*

Proof. We use a reduction from the well known *NP*-complete problem SAT to our security check problem with one journeyer (s, t) and several pairs of invaders $(s_i, t_i) \in I$. Given an instance of SAT with n variables and m clauses we construct an acyclic graph in polynomial time as follows:

The vertex set is going to consist of vertices s and t and a vertex for each literal $l_{k,i}$ in each clause C_k. Now s has arcs towards $l_{1,i}$ for all $l_{1,i} \in C_1$, and each literal $l_{k,i}$ $(0 < k < m)$ has arcs towards all literals in C_{k+1}, i.e. all literals $l_{k+1,i}$ for all i. Finally, all literals $l_{m,i} \in C_m$ have an arc towards t.

Next, we introduce the set of invaders. Each pair of literals, where one represents the complementary of the other, forms a pair $(s_i, t_i) \in I$. Note, there are at most nm^2 of such pairs, assuming that each clause contains the same literal at most once. (Actually one would only need pairs (l_{k,i_1}, l_{l,i_2}) where $k < l$.) Finally, all traveling times on arcs will be 0 and increase by 1 if a security check is installed. For a visualization of the construction for an example see Figure 2.12

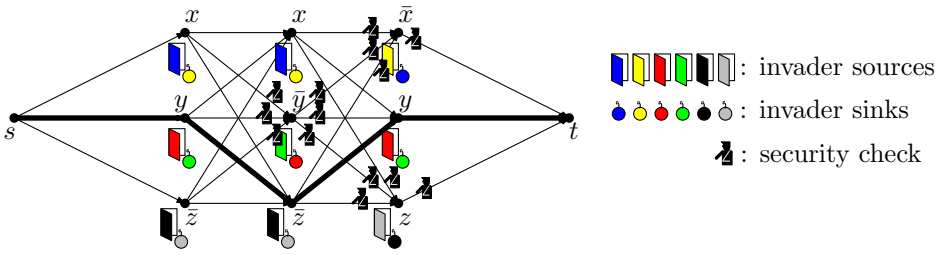


Figure 2.12: Visualization of the constructed graph for the instance $(x \vee y \vee \bar{z}) \wedge (x \vee \bar{y} \vee \bar{z}) \wedge (\bar{x} \vee y \vee z)$ with truth assignment $x = y = \bar{z} = true$. The instance has a total of 12 invaders.

We claim, that there exists an allocation of security checks deterring all invaders and an s-t-path of length 0, if and only if the formula is satisfiable. First, if the formula is satisfiable, there exists an s-t-path of length 0 by traveling through any true literal in each clause and blocking all other arcs, which then deters all invaders as the path may not contain complementary literals.

Second, if there exists an allocation of security checks and a path connecting s and t of length 0, it may not contain any pair of complementary literals (those would form an invader pair and one would have placed a security check in between, contradicting the path length of 0). Therefore, there exists an assignment such that the formula is satisfiable by setting the variables corresponding to the passed literals in such a way that those literals are all true. □

Theorem 32. *Given a directed graph $G = (V, A, \tau)$, and let $\tau_a^c = \tau_a + \lambda$ for all $a \in A$, then $(I, 1)$-*Tesc *is solvable in $\mathcal{O}(n^2)$ time if $|I|$ is constant.*

Before starting to prove the theorem, we state the following observation.

Observation 33. *If there is only one journeyer (s, t), there always exists an allocation of security checks that minimizes the length of a shortest s-t-path, in which the shortest s-t-path traverses security checks only on outgoing arcs of source vertices s_i or incoming arcs of sink vertices t_i for $(s_i, t_i) \in I$.*

Consider an allocation of security checks that deters all invaders and let P be the s-t-path minimizing the distance. W.l.o.g. we can assume that there are security checks placed on all arcs not on P. Let $\mathcal{I} \subseteq \{s_1, \ldots, s_k\} \cup \{t_1, \ldots, t_k\}$ be the vertices the path P traverses. Observe that each security check the path traverses can be moved either to the last visited source vertex s_i or next visited sink vertex t_i without generating any path without a security check between a pair of invaders. This is also true for an optimum allocation.

Proof of Theorem 32. The procedure is described in Algorithm 4. Key for the algorithm is Observation 33. The high level idea is as follows. We iterate over all subsets of vertices of $s_1, \ldots, s_k$, and $t_1, \ldots, t_k$ an optimum path could traverse. For each such subset, we also guess the vertices that have security checks on all outgoing/incoming arcs (c.f. Observation 33) and update the arc weights accordingly. Then, we compute a shortest walk from s to t traversing the vertices of the subset in the guessed order with the updated weights. If all invaders are deterred, we store the length of the computed s-t-walk and return the shortest of those once we have checked all possibilities.

Algorithm 4: Solving $(I, 1)$-TESC

Input: Graph $G = (V, A, \tau)$, invaders $I = \{(s_1, t_1), \ldots, (s_k, t_k)\}$ with $S = \bigcup_i s_i$ and $T = \bigcup_i t_i$ and a single journeyer $J = \{(s, t)\}$

Output: Deterring security check allocation $C \subseteq A$ minimizing the length of a shortest s-t-path

Let $C = V$ and P be a shortest s-t-path in $G = (V, A, \tau + \lambda)$ **for** ***each*** $\mathcal{I} \in \mathcal{P}(S \cup T)$ **do**
  for ***each*** *bijective* $\pi : \mathcal{I} \to [|\mathcal{I}|]$ **do**
    for ***each*** $c : \mathcal{I} \to \{0, 1\}$ **do**
      $C' := \{(u, v) \in A : u \in S \cap \mathcal{I} \text{ and } c(u) = 1\} \cup \{(u, v) \in A : v \in T \cap \mathcal{I} \text{ and } c(v) = 1\}$,
      $C' = C' \cup \{(u, v) \in A : u \in S \setminus \mathcal{I} \text{ or } v \in T \setminus \mathcal{I}\}$
      let $V' := V \setminus ((S \cup T) \setminus \mathcal{I})$, $A' := \{(u, v) \in A : u, v \in V'\}$
      let τ' such that $\tau'(a) := \tau(a)$ for all $a \in A' \setminus C'$ and $\tau'(a) = \tau(a) + \lambda$ for all $a \in C'$
      compute a shortest $(s\text{-}\pi(1)^{-1}\text{-}\ldots\text{-}\pi(|\mathcal{I}|)^{-1}\text{-}t)$-tour P' in $G' := (V', A', \tau')$
      let $C' = C' \cup A' \setminus \{a \in A' : a \in P'\}$
      if *C' is a deterring security check allocation* ***and*** $\tau'(P') < \tau(P)$ **then**
        let $C = C'$ and $P = P'$
      end
    end
  end
end
return C

As mentioned above, first of all, guess which vertices $\mathcal{I} \subseteq \{s_1, \ldots, s_k\} \cup \{t_1, \ldots, t_k\}$ the optimal path traverses. Now, consider all possible orderings of the guessed vertices in $\mathcal{I}$. Once we have fixed the order $\pi : \mathcal{I} \to [|\mathcal{I}|]$, choose for each vertex one of two options. For vertices $s_i \in \mathcal{I}$ either secure all outgoing arcs or none whereas for vertices $t_i \in \mathcal{I}$ either secure all incoming arcs or none. Additionally, place security checks on all arcs incident to vertices s_i or t_i not in $\mathcal{I}$. Proceed for each of these possible placements of security checks as follows: Compute a shortest $(s\text{-}\pi(1)^{-1}\text{-}\ldots\text{-}\pi(l)^{-1}\text{-}t)$-walk by iteratively computing a shortest $(\pi(j)^{-1}\text{-}\pi(j+1)^{-1}$-path, where $\pi(0)^{-1} := s$

and $\pi(|\mathcal{I}|+1)^{-1} := t$. Then, place additional security checks on all arcs not on this walk and check if the allocation is a deterring all invaders. Note that once security checks are fixed on the graph, one can verify for each invader in polynomial time, whether it has been deterred or not, by checking for connectivity after removing all arcs from the network which carry a security check. Finally, return the security check allocation deterring all invaders with the shortest s-t-path[6].

We need to prove that we find an optimal allocation and a corresponding path in some iteration. Let C^* be an optimal security check allocation and let P^* be a shortest s-t-path in the network with security checks according to C^*. Consider all vertices $\mathcal{I}^*$ on this path that are an invaders' source or sink and let π^* be the ordering of these vertices induced by the path P^*. Due to Observation 33 we may assume that all security checks in C^* on P^* are allocated on outgoing arcs of vertices s_i or incoming arcs of vertices t_i. In some iteration, the algorithm must choose $\mathcal{I} = \mathcal{I}^*$, $\pi = \pi^*$, and C in such a way that we exactly match the placements of security checks on the path P^*. By a simple inductive argument, the computed $(s\text{-}\pi^*(1)^{-1}\text{-}\ldots\text{-}\pi^*(|\mathcal{I}^*|)^{-1}\text{-}t)$-tour must have the same length as P^*, since the segments, i.e. the $(\pi^*(j)^{-1}\text{-}\pi^*(j+1)^{-1})$-paths, must be of the same length. The computed tour cannot use any source or sink vertex not in $\mathcal{I}$, and, using any vertex s_i or t_i on P^* with $\pi^*(s_i) > j$ or $\pi^*(t_i) > j$ would imply that one could shortcut P^* by directly continuing from vertex s_i or t_i. Thus, the algorithm finds a tour of length $\tau(P^*)$ and returns an optimal security check allocation.

Last but not least, we need to prove that Algorithm 4 runs in polynomial time. Observe that we have a constant number of sets, a constant number of permutations π for each of these sets, and finally a constant number of functions c for each permutation. Therefore, the number of iterations is constant, whereas in each iteration we need to compute C', V', A' and τ' each in $\mathcal{O}(n^2)$ time. Additionally, we compute a tour connecting the corresponding vertices in $\mathcal{I}$ in given order taking $\mathcal{O}(|\mathcal{I}|n^2)$ time. Finally, check if C' is a deterring security check allocation, that takes $\mathcal{O}(|I|n^2)$ time. Therefore, Algorithm 4 runs in $\mathcal{O}(f(|I|)\cdot n^2)$ time, where

$$f(|I|) = \underbrace{\sum_{l=0}^{2|I|} \binom{2|I|}{l}}_{\text{outer loop}} \cdot \underbrace{l!}_{\text{second loop}} \cdot \underbrace{2^l}_{\text{third loop}} \cdot \underbrace{(l+|I|)}_{\text{operations in third loop}},$$

which is constant if $|I|$ is constant. □

[6] It is possible to shortcut walks and trails, however, the corresponding path can be found for the correct choice of $\mathcal{I}$ as well.

The Undirected Case

Theorem 34. *Given an undirected graph $G = (V, E, \tau)$.*

*i) If $\tau_e^c = \tau_e + \lambda$ for all $e \in E$, (I, J)-*Tesc *is NP-complete with $|I|, |J| \geq 1$.*

*ii) If $\tau_e^c = \alpha_e \tau_e$ for all $e \in E$, (I, J)-*Tesc *is NP-complete with $|I|, |J| \geq 1$.*

Proof. For both cases, we use a reduction from k undirected disjoint paths problem (k-DPP), where k is part of the input.

i) Given an instance of k-DPP on a graph G and k tuples $(s_i, t_i)_{i \in [k]}$. We construct an instance of (I, J)-Tesc as follows: Let $I = \{(s_i, s_j) : i, j \in [k] \wedge i \neq j\}$, $J = \{(s_i, t_i) : i \in [k]\}$, $\tau \equiv 0$, and $\tau^c = 1$.

We claim the instance of (I, J)-Tesc has a solution of cost 0 if and only if, the instance of k-DPP has a solution. Obviously, if there exists a set of disjoint s_i-t_i-paths for every $i \in [k]$, placing security checks on all arcs not on any of the paths deters all invaders. On the other hand, if we find a security check placement of cost 0, no two paths P_i, P_j may intersect. If they do, due to $(s_i, s_j) \in I$, they must contain at least one security check, which implies that one path contains an edge with security check of cost 1. This contradicts a solution of cost zero.

ii) The proof only changes slightly. We use the same construction but with $\tau \equiv 1$ and $\alpha = n^2$. It is easy to see, that there exists a solution to k-DPP if and only if the constructed instance has a solution of cost at most n. □

In contrast to directed instances, one can solve general cost $(1, 1)$-Tesc in polynomial time.

Theorem 35. *Given an undirected graph $G = (V, E, \tau)$ with $\tau_e^c = \tau_e + \lambda_e$ for all $e \in E$. Then $(1, 1)$-*Tesc *is solvable in polynomial time.*

Proof. The main idea underlying the algorithm: Either, the path contains exactly one security check or none at all.

First, if the optimal solution does not need any security check, we find such a solution by computing two shortest s-t-paths when removing s_1 and t_1 from the graph, respectively.

However, if the optimum solution contains one security check, we find that edge in the following way. Guess the edge containing a security check. From the undirected graph then construct a directed graph and solve a min-cost-flow problem on the construction. With this computation, we determine the minimum cost of a solution (if it exists) with a security check on that edge.

Given a specific edge $\{u', v'\}$ of the graph, we can compute two disjoint s-u'-path and v'-t-path or two disjoint s-v'-path and u'-t-path of minimum total cost with the following construction.

Replace each edge but the edge $\{u', v'\}$ by its bidirectional arc correspondences, i.e. replace $\{u, v\} \in E$ by $(u, v), (v, u)$. Then, replace every vertex by two vertices v^- and v^+, as in Figure 2.2. Direct all ingoing arcs to v towards v^- and let all outgoing arcs from v leave from v^+. Additionally, add the arc (v^-, v^+). Let the capacity of all arcs be 1, i.e. $c \equiv 1$. Let the cost of an arc (u, v) with $\{u, v\} \in E$ be $\tau_{(u,v)} = \tau_{\{u,v\}}$ and $\tau_{(v^-,v^+)} = 0$. Finally, add a source vertex s^f and arcs $(s, u'^-), (s, v'^-)$ and a sink t^f vertex with arcs $(s^+, t^f), (t^+, t^f)$. All these arcs have capacity 1 and cost 0. Now, compute a minimum cost s^f-t^f-flow of value 2.

Due to integral capacities, if there exists a solution, there always exists an integral minimum cost flow. This solution consists of two disjoint s^f-t^f-paths, which contain two disjoint paths from s^f to s and t. Either an u'-s-path and an u'-t-path or an u'-t-path and an u'-s-path. Both paths have flow value one and therefore their cost contribution is exactly the cost of the path in the original undirected graph.

Finally, observe that a minimum cost path solution to the original problem using this one particular edge $\{u', v'\}$, directly gives rise to a flow of value 2 with same cost. □

Corollary 36. *Given an undirected graph $G = (V, E, \tau)$ with $\tau_e^c = \tau_e + \lambda$ for all $e \in E$. Then $(I, 1)$-*Tesc *is solvable in polynomial time if $|I| \geq 1$ is constant.*

Proof. We may use Theorem 32 with a standard gadget to transform undirected edges into directed edges to compute a solution. This gadget is depicted in Figure 2.2. To overcome issues arising with security check placements on arcs $(u, z_{uv}^-), (v, z_{uv}^-), (z_{uv}^+, u)$, and (z_{uv}^+, v), we double the number of invaders. We replace each invader s_i, t_i by a pair of invaders $(s_i, t_i), (t_i, s_i)$. Given a solution to the undirected case, we find a solution of same cost by placing security checks on the corresponding directed arc (z_{uv}^-, z_{uv}^+).

Given a solution to the directed instance, placing security checks on all edges but the final path, deters all invaders but invaders, that have both s_i, t_i on the path, without increasing the total cost. W.l.o.g., since both $(s_i, t_i), (t_i, s_i) \in I$, assume that the path traverses all invader pairs in s_i-t_i order. But then, the path has to contain at least as many security checks on the directed arcs as a deterring placement of security checks for that path in the undirected case. Overall, we obtain a solution of same cost for the undirected case. □

2.3.3 Flows

In this subsection, we again address the question, where to install security checks on our graph such that invaders cannot reach their destination from their respective starting point without traveling along an arc with a security check. But this time we

want to maximize the flow we can send from a set of source vertices to a set of sink vertices. Here, installing a security check will decrease the capacity of an arc. We define the following problem:

Definition 37 (Flow efficient security checks)**.** *Given a directed graph $G = (V, A, \gamma)$ with arc capacities γ. Additionally, there is a set of invaders I and a set of journeyers J each represented by a tuple consisting of a source and a sink. We let capacities on arcs decrease when a security check is installed, i.e. $\gamma_a^c \leq \gamma_a$. Let (I, J)-*Fesc *refer to the problem of deciding whether there exists a security check allocation on the arcs on G in such a way that all invaders are deterred and the sum of maximum journeyers flows exceeds F.*

Theorem 38. *Given a directed graph $G = (V, A, \gamma)$.*

*i) If $\gamma_a^c = \alpha_a \gamma_a$ for all $a \in A$, (k, l)-*Fesc *is NP-hard for all $k, l \geq 1$, even if $k = l = 1$ and the invader pair and the traveler pair agree, and $\alpha_a = \alpha$ for all but one $a \in A$.*

*ii) If $\gamma_a^c = \alpha \gamma_a$ for all $a \in A$, $(1, 1)$-*Fesc *is NP-hard, even if $\gamma_a = 1$, $\alpha_a = \alpha$ for all $a \in A$, and source and sink of travelers and invaders agree.*

Proof. i) We can use the same reduction as in the proof of Theorem 29 i). Let all arc capacities be 1 and instead of setting $\lambda_a = 0$ we let $\alpha_a = 0$ for all arcs but (t_1, s_2) and observe that there is a s_1-t_2-flow of value 1 if and only if there are two disjoint paths connecting s_1, t_1 and s_2, t_2.

ii) We once more use a reduction from the disjoint paths problem. Let $G = (V, A)$ and $(s_1, t_1), (s_2, t_2)$ be an instance of the disjoint path problem. We let the underlying graph be the same graph as in the disjoint paths problem. Additionally, we let $s = s_1$ and $t = t_2$, the source and sink for both, the traveler and the invader. Furthermore, let $\alpha = \frac{1}{n^2}$ and finally replace the arc (t_1, s_2) by n^2 directed paths of length 2 from t_1 to s_2. We claim that there exist two disjoint paths from s_1 to t_1 and from s_2 to t_2 if and only if there is an allocation of security checks separating s_1 and t_2 with a maximum flow value greater or equal to 1.

First, if there are two disjoint paths, we can separate s_1 and t_2 by placing security checks everywhere but on the paths. The flow value is at least 1 since we can route a flow along the paths and then send a flow of value 1 from t_1 to s_2 along the n^2 paths, each with capacity $\frac{1}{n^2}$ connecting them.

Second, any placing of a minimal number of security checks induces a cut in the graph. So, if there is a flow of value 1, and since the security checks induce a cut, the value of the cut must be at least 1. This implies, since all arcs on that cut have capacity $\frac{1}{n^2}$, that the cut must contain at least one arc of the newly added paths between t_1 and s_2, directly implying that it must contain an arc of each of these paths. Now there must be a path from s_1 to t_1 and from s_2 to t_2 on two disjoint sets of vertices which means that both paths are disjoint. □

Remark 39. There exists no polynomial time ρ-approximation algorithm for $(1,1)$-FESC for any constant ρ, unless $\mathcal{P} = NP$.

Proof. Given a constant ρ, we prove that there cannot exist a polynomial time algorithm with approximation ratio ρ. The proof uses the same construction as in the proof of Theorem 38 (ii), but instead of $\alpha = \frac{1}{n^2}$, we let $\alpha = \frac{\rho}{n^3}$ and $\frac{n^3}{\rho}$ paths between t_1 and s_2. □

2.3.4 Open Problems

In this section, we gave a polynomial time algorithm for both, the directed and undirected version of $(I,1)$-TESC if $|I|$ is constant. Hardness for the directed version of $(1,2)$-TESC followed from 2-dDPP, making use of zero weight edges. The proof cannot easily be adopted to the case if only strictly postive edge weights were allowed. Thus, we pose the following questions for further research:

Open Problems.

1. *What is the computational complexity of undirected* $(I,2)$-TESC *if* $\tau_e = 1$ *for all* $e \in E$ *and* $\lambda_e = \lambda$, *or* $\alpha_e = \alpha$ *for all* $e \in E$*?*
2. *What is the computational complexity of directed* $(I,2)$-TESC *if* $\tau_a = 1$ *for all* $a \in A$ *and* $\lambda_a = \lambda$, *or* $\alpha_a = \alpha$ *for all* $a \in A$*?*

Additionally, numerous classical polynomial time solvable problems can be studied w.r.t. this new modification of increasing cost on cut edges, e.g. the maximum weight matching problem or the minimum spanning tree problem.

Chapter 3

Graph Processes

In this chapter, we study three processes on graphs. Two of the processes have applications in statistical physics, e.g. can be used to model ferromagnetism or glassy dynamics. First, we derive bounds for special sets in bootstrap percolation on graph classes, such as degenerate graphs and grids. Second, we analyze the limit behavior of a three-state contact process. The three states are infected, two healthy states, active and passive, with different infection probabilities.

The third process can be regarded as an election on two options and partly coincides with the model of US presidential elections. We design algorithms computing an order of votes aiming to maximize the number of votes for one option.

3.1 Bootstrap Percolation

The first process studied in this chapter is bootstrap percolation on degenerate graphs and grids. Note, a slightly weaker version of Theorem 41 was obtained in the author's master's thesis. However, with a finer proof analysis we derived a tight bound, c.f. Lemma 43. Parts of this section are published in [45], particularly, most of Section 3.1.1 and some parts of Section 3.1.2 coincide.

3.1.1 The Model and Related Work

Bootstrap percolation is a discrete time infection process on a graph $G = (V, E)$. Uninfected vertices become infected over time if the number of adjacent infected vertices exceeds some threshold. Let this threshold be $r \in \mathbb{N}$. We also refer to this process as r-neighbor bootstrap percolation.

Initially, at time zero, there is some set of infected vertices. At every next time step, any vertex that is adjacent to at least r infected neighbors becomes infected in the next time step, too. The process was first introduced by Chalupa et al. in 1979 in [23] and is a simple example for a cellular automaton. It is also closely related to the Glauber dynamics, which represent the Ising model at zero-temperature (see [65]). Another application one can think of is rumor spreading in a social network, where individuals start spreading a rumor to all their friends once they have heard that rumor from a number of other friends. Instead of speaking of infection, the literature also uses the term activation, but we shall stick to the term infection.

We now shortly introduce the process formally before recalling some known results. Call the set of initially infected vertices A_0 and the vertices that are infected at the end of the process A_f. More formally, let A_t be the vertices which are infected at time t, where $A_t := A_{t-1} \cup \{v \in V : |N(v) \cap A_{t-1}| \geq r\}$ and $N(v)$ denotes the neighborhood of some vertex $v \in V$ and thus $A_f = \bigcup_{t>0} A_t$.

The dynamics of bootstrap percolation are be depicted on an example graph in Figure 3.1.

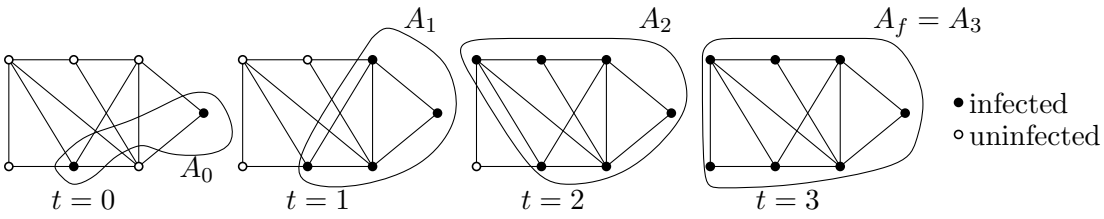


Figure 3.1: Dynamics of the 2-neighbor bootstrap percolation process on an example graph where $|A_0| = 2$ and $A_f = V$.

Sets A_0 that infect the entire graph, i.e. $\bigcup_{t\geq 0} A_t = V$, are called percolating sets. A minimal percolating set is a percolating set where every proper subset does not infect the entire graph. In many applications, the set A_0 is a random subset of vertices, where each vertex is initially infected independently with a given probability p.

For several graph classes, there is already much known about the behavior of this process and there are a quite few things of interest. First of all, one can study the probability of percolation, i.e. the probability that $A_f = V$, depending on p, with which each vertex is initially infected, independently of all other vertices. Several authors surveyed the probability threshold at which percolation is more likely to occur than not. For example, if the underlying graph is the d-dimensional cube graph $[n]^d$, the exact threshold function for $d = r = 2$ was shown by Holroyd in [56] to be $\frac{\pi^2}{18\log n} + o(\frac{1}{\log n})$. Later, Balogh et al. gave the exact threshold function for all $d \geq r \geq 2$ in [6]. Additionally, threshold functions for bootstrap percolation were intensively studied on trees, like periodic trees in [20] and Galton-Watson trees in [18] and [7].

Some papers also study the size of sets of vertices that infect the entire graph. There is also a special class of percolating sets, namely minimal percolating sets, which are sets of vertices where any proper subset does not infect the entire graph. Riedl showed in [78] that for a tree on n vertices with l vertices of degree less than r, a minimal percolating set A_0 is of size $\frac{(r-1)n+1}{r} \leq |A_0| \leq \frac{rn+l}{r+1}$. Riedl also gave an algorithm in his paper that computes the size of a minimum percolating set for $r \geq 2$ as well as the size of a maximum minimal percolating set for $r = 2$ in linear time. In another paper, he extended the studies of minimal percolating sets to hypercubes under 2-neighbor bootstrap percolation (see [77]).

Also, other graphs like the well known Erdős-Rényi random graph have been studied, for example in [57] by Janson et al. There, the authors give a function for the edge probabilities when percolation occurs with high probability, depending on the size of A_0.

Another quite interesting parameter is the running time of such a process, which is the time until no new vertex becomes infected, i.e. the least t such that $A_t = A_{t+1} = A_f$. This parameter has been studied for several graphs like the grid $[n]^2$, where Benevides and Przykucki [12] showed, that for $r = 2$ the time of the process is bounded by $\frac{13}{18}n^2 + \mathcal{O}(n)$. In [75], Przykucki considered bootstrap percolation on the d-dimensional hypercube and proved the time to be at most $\lfloor \frac{d^2}{3} \rfloor$, again for $r = 2$. Bollobás et al. [19] analyzed the time of bootstrap percolation on the discrete torus, while Janson et al. [57] gave a time-bound for bootstrap percolation on the Erdös-Rényi random graph.

3.1.2 Degenerate Graphs

In this subsection, we focus on the size of the infected set A_f at the end of the bootstrap percolation process on degenerate graphs. We give a result for r-neighbor bootstrap percolation when the underlying graph is a degenerate graph and the set A_0 is an

arbitrary subset of the vertices. Furthermore, our theorem implies a bound on the size of percolating sets. Another direct consequence is an upper bound on the running time of the process on degenerate graphs. Finally, it is noteworthy that, due to bounded degeneracy of several known graph classes, like trees or planar graphs, our results also covers more commonly studied graph classes.

Our main result gives a tight bound on the size of the set A_f of vertices that are infected at the end of the process on degenerate graphs. Recap the definition of degeneracy.

Definition 40. *A graph $G = (V, E)$ is called d-degenerate if every subgraph contains a vertex of degree at most d.*

There are many graph classes that have a bounded degeneracy. For example, forests are 1-degenerate graphs, planar graphs are 5-degenerate while outerplanar graphs are 2-degenerate. Also, scale free networks generated by the Barabási-Albert model using a preferential attachment mechanism, have bounded degeneracy.

Let us now state our main theorem.

Theorem 41. *Consider bootstrap percolation on a d-degenerate graph G with $r \geq d + 1$ and a set of initially infected vertices A_0. Additionally assume $|A_0| \geq d$. Then the set A_f of vertices that are infected at the end of the process fulfills*

$$|A_f| \leq \left(1 + \frac{d}{r-d}\right)|A_0| - \frac{d(d+1)}{2(r-d)}.$$

Proof. First of all, since the process is deterministic, once A_0 is fixed, we may assume that $A_f = V$ (otherwise delete vertices that do not become infected from the graph). There exists an ordering (Erdös-Hanjal sequence, c.f. Section 2.2.5) $\pi : V \to V$ of the vertices, where each vertex has at most $d < r$ neighbors to its left, i.e. $|\{w \in \delta(v) : \pi(w) < \pi(v)\}| \leq d$. W.l.o.g. assume that $V = [n]$ and $\pi(i) = i$ for all $v \in V$. Observe next, since $d < r$, every vertex that becomes infected at some point must have at least one already infected vertex in its right neighborhood.

We want to make use of this observation and therefore introduce a potential Φ_t, which, after each time step t of the infection process, bounds the number of vertices that might be infected in the next step from above.

Let $\Phi_t = \sum_{v \in V} \Phi_t^v$, where

$$\Phi_t^v := \begin{cases} 0, & \text{if} \quad v \notin A_t \\ |\{w \in \delta(v) : \pi(w) < \pi(v)\} \setminus A_t|, & \text{else} \end{cases}$$

is the number of uninfected vertices in the left neighborhood of vertex v if it is infected and is 0 otherwise. Note that an uninfected vertex might be counted more than once in

our potential if it has more than one infected vertex in its right neighborhood. Clearly, $\Phi_0 \leq d|A_0|$ since every initially infected vertex has at most d uninfected neighbors to its left.

Lemma 42. *The potential decreases for each infection at time t by at least $r - d$, i.e. we have that $\Phi_{t-1} - \Phi_t \geq (r - d)|A_t \setminus A_{t-1}|$.*

Proof. Consider the set of vertices infected at time t and let vertex v be such a vertex, i.e. $v \in A_t \setminus A_{t-1}$. Due to the degeneracy of the graph, v has at most d uninfected neighbors in its left neighborhood and therefore v can increase the potential by at most d. Observe next that for vertex v to become infected it must have at least r infected neighbors. Now there are two kinds of such infecting vertices, namely those that lie in the right neighborhood of v and those that lie in the left neighborhood of v as depicted in Figure 3.2. Let λ be the number of infected vertices in the left neighborhood and ρ

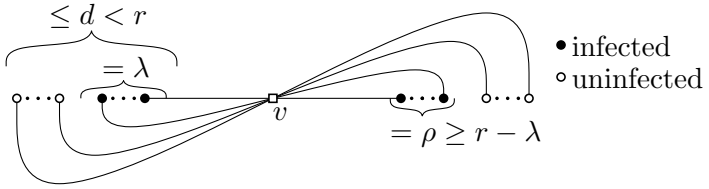


Figure 3.2: Left and right neighborhood of v at the time of its infection.

be the number of infected vertices in the right neighborhood. For each infected vertex w that lies in the right neighborhood, the potential decreases by one, since vertex v was accounted for in Φ^w_{t-1} but is no longer uninfected and therefore does not contribute to Φ^w_t. Each of the λ infected vertices in the left neighborhood of v does not add to the potential either, since it is already infected, so $\Phi^v_t \leq (d - \lambda) + \Phi^v_{t-1}$. If v is the only vertex infected at time i, we have that

$$\Phi_t \leq (d - \lambda) - \rho + \Phi_{t-1} \leq (d - \lambda) - (r - \lambda) + \Phi_{t-1} \leq \Phi_{t-1} - (r - d),$$

which, since $r > d$, means that the potential decreases by $r - d$. Finally, note that additional, simultaneous infections do not increase Φ^v_t and the above argumentation can independently be done for each vertex that is infected at time i, and thus each infection decreases the potential by the aforementioned amount and the claim follows. □

Now consider the left most d that are infected at the end of the process. W.l.o.g. these shall be the the vertices in $[d]$. There are two cases for each of these vertices:

Case 1: Vertex $i \in [d]$ is an initially infected vertex, i.e., $i \in A_0$, then we may bound the contribution of i to Φ_0 by $i - 1$ instead of d.

Case 2: Vertex $i \in [d]$ becomes infected at some point in time t, i.e. $i \in A_f \setminus A_0$. Now, once vertex i becomes infected, it contributes at most $i-1$ to the potential, while decreasing the potential by r due to its infection. Therefore, we may bound the decrease of the potential due to vertex i by $r-(i-1)$ instead of $r-d$.

So in total, we have that the vertices in $[d]$ are either in A_0 and decrease Φ_0 or decrease the potential at the time of their infection. Once the potential is less than 1 the process stops. This is simply due to the fact that there does not exist an uninfected vertex that has an infected vertex in its right neighborhood and hence could be infected next. We can now bound the number additional of infections by

$$\frac{\Phi_0 - \sum_{i \in [d] \cap A_f \setminus A_0} (r-(i-1))}{r-d} + |[d] \cap A_f \setminus A_0|$$

$$\begin{aligned}
&\leq \frac{d|A_0| - \sum_{i \in [d] \cap A_0} (d-(i-1)) - \sum_{i \in [d] \cap A_f \setminus A_0} (r-(i-1))}{r-d} + |[d] \cap A_f \setminus A_0| \\
&= \frac{d|A_0| - \sum_{i \in [d] \cap A_0} d + \sum_{i \in [d]} (i-1) - \sum_{i \in [d] \cap A_f \setminus A_0} r}{r-d} + |[d] \cap A_f \setminus A_0| \\
&= \frac{d|A_0| - d|[d] \cap A_0| + \frac{d(d-1)}{2} - r|[d] \cap A_f \setminus A_0|}{r-d} + |[d] \cap A_f \setminus A_0| \\
&= \frac{d|A_0|}{r-d} + \frac{d(d-1)}{2(r-d)} - \frac{d|[d] \cap A_0| + r|[d] \cap A_f \setminus A_0| - (r-d)|[d] \cap A_f \setminus A_0|}{r-d} \\
&= \frac{d|A_0|}{r-d} + \frac{d(d-1)}{2(r-d)} - \frac{d|[d] \cap A_0| + d|[d] \cap A_f \setminus A_0|}{r-d} \\
&= \frac{d|A_0|}{r-d} + \frac{d(d-1)}{2(r-d)} - \frac{d^2}{r-d} = \frac{d|A_0|}{r-d} - \frac{d(d+1)}{2(r-d)}.
\end{aligned}$$

Hence,

$$|A_f| \leq |A_0| + \frac{d|A_0|}{r-d} - \frac{d(d+1)}{2(r-d)} = \left(1 + \frac{d}{r-d}\right)|A_0| - \frac{d(d+1)}{2(r-d)}.$$

It is obvious that $|A_0| \leq |A_f|$, which finishes the proof of Theorem 41. □

The theorem requires $r \geq d+1$ which lets us derive a slightly simpler bound on A_f, namely that

$$|A_f| \leq (d+1)\left(|A_0| - \frac{d}{2}\right).$$

It is also remarkable that our bound on the size of A_f in Theorem 41 is sharp in the

following sense:

Lemma 43. *For every odd $d \geq 1$ and $r \geq d+1$, such that $d \mid r$ or $d \mid (r - \frac{1}{2}(d-1))$, there exists a d-degenerate graph with an initially infected set A_0, such that the set A_f at the end of the process fulfills $|A_f| = \left(1 + \frac{d}{r-d}\right)|A_0| - \frac{d(d+1)}{2(r-d)}$.*

Proof. In the following, we describe the construction of such a graph. The graph is constructed assuming a total order on the vertices that gives rise to an Erdős-Hajnal sequence. All but the first d vertices have exactly d neighbors of smaller label and all vertices not in A_0 have exactly r neighbors with larger label.

For the construction of such a graph take three disjoint sets of vertices $[d]$, H, and A_0, such that

$$|H| = r - \frac{1}{2}(d-1), \quad \text{and} \quad |A_0| = \frac{r(r - \frac{1}{2}(d-1))}{d}.$$

Now, let the vertices in $[d]$ form a clique of size d. Additionally, let $H_1 \cup H_2 = H$ such that $|H_1| = r - d + 1$ and $H_2 = H \setminus H_1$. Then, let every vertex in H_1 be adjacent to all vertices in $[d]$. Therefore, every vertex in H_1 has d neighbors of smaller label.

Note, if all vertices in H_2 have d neighbors of smaller label these are

$$d(|H| - (r - d + 1)) = \frac{1}{2}d(d-1) = \sum_{i=1}^{d}(i-1)$$

many edges. Furthermore, as mentioned above, every vertex in A_0 shall also have degree d, which gives us additional $d|I| = r(r - \frac{1}{2}(d-1))$ edges that we have to match to vertices in $H_1 \cup H_2 \cup K_d$.

On the other hand, every vertex in H_1 needs r neighbors in $H_2 \cup A_0$ and every vertex $i \in [d]$ needs $i-1$ neighbors in $H_2 \cup A_0$ in order to be adjacent to exactly r vertices with larger label. Therefore, we need in total

$$\sum_{i=1}^{d}(i-1) + r(r-d+1) = \frac{d(d-1)}{2} + r(r-d+1)$$

edges from $H_2 \cup A_0$ to $[d] \cup H_1$.

First, we match all vertices in H_2 to vertices missing edges in H_1. Since $d \mid r$ or $d \mid (r - \frac{1}{2}(d-1))$, we have that $r - d + 1 \geq d > \frac{1}{2}(d-1)$, and we can easily assign all $\frac{1}{2}d(d-1)$ edges from vertices in H_2 to vertices in H_1 in this way without creating multi edges.

Finally, there are still

$$\frac{d(d-1)}{2} + r(r-d+1) - \frac{1}{2}d(d-1) = r(r-d+1)$$

not yet assigned edges from $[d] \cup H_1$ which all can be matched with the remaining edges from A_0.

A schematic picture of the graph can be seen in Figure 3.3.

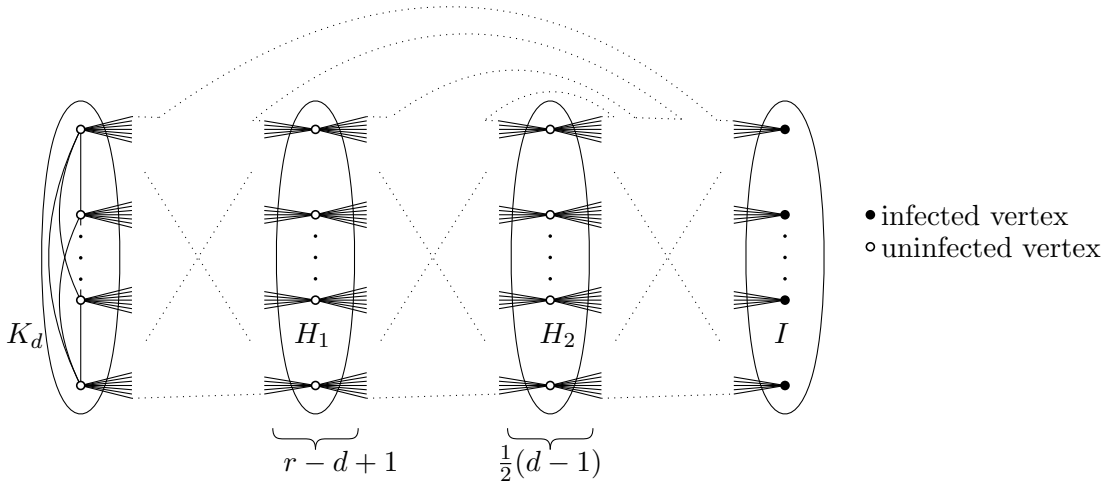


Figure 3.3: Schematic picture of the constructed graph that matches the given upper bound.

The set A_0 is of size $\frac{r}{d}(r - \frac{1}{2}(d-1))$. Also, $A_f = V$, since every vertex will be infected at the end. We obtain

$$|A_0| + |H| + d = |A_0| + (r - \frac{1}{2}(d-1)) + d.$$

Hence, we have that

$$\begin{aligned}
|A_f| &= |A_0| + (r - \frac{1}{2}(d-1)) + d = \\
&= |A_0| + \frac{d}{r}|A_0| + d \\
&= \left(1 + \frac{d}{r-d}\right)|A_0| - \frac{d^2}{r(r-d)}|A_0| + d \\
&= \left(1 + \frac{d}{r-d}\right)|A_0| - \frac{d}{r-d}\left(r - \frac{1}{2}(d-1)\right) + d \\
&= \left(1 + \frac{d}{r-d}\right)|A_0| - \frac{d(d+1)}{2(r-d)}
\end{aligned}$$

which matches the upper bound in Theorem 41. □

Theorem 41 directly implies a bound on percolating sets.

Corollary 44. *Consider bootstrap percolation on a d-degenerate graph with $n \geq d$ vertices and let $r \geq d+1$. Then the size of a percolating set A_0 fulfills*

$$\tfrac{r-d}{r}n + \frac{d(d+1)}{2r} \leq |A_0|.$$

This bound is obtained by setting $|A_f| = n$ in Theorem 41.

Note, Riedl gave a bound for percolating sets in trees in [78]. Recall, every tree is a 1-degenerate graph, since every subgraph of a tree is a forest and thus contains at least one leaf, which is a vertex of degree one. As already mentioned in Section 3.1.1, Riedl considered the size of percolating sets on trees and proved that a percolating set is of size at least $\frac{r-1}{r}n + \frac{1}{r}$. Our bound yields the same bound. Also note, the set A_0 in the proof of Lemma 43 is percolating and therefore proves that this bound is essentially tight. In this case, the constructed graph corresponds to the r-ary tree with r^2 leaves.[1]

Remark 45. The proof of Theorem 41 can easily be modified to obtain an even stronger bound for percolating sets. The idea of the missing edges of the first d infected vertices, which gave a reduced potential increase upon infection can be used for all vertices that do not have d left-neighbors. These missing edges make up for a total potential decrease of $d|V| - |E|$. Making use of the fact that all vertices become infected at some point (either are infected in A_0 or at a later point) and with the same case distinction, we can derive the following bound:

$$|A_f| \leq |A_0| + \frac{d|A_0|}{r-d} - \frac{d|V| - |E|}{(r-d)}.$$

As $|A_f| = n$, we obtain

$$\frac{r-d}{r}n + \frac{d|V| - |E|}{r} \leq |A_0|.$$

The construction in Lemma 43 is a percolating set and also matches the bound given in Remark 45. This is due to the fact that in the construction we get that $d|V| - |E| = \frac{d(d+1)}{2}$.

Finally, we can also prove a similar bound to Corollary 44 for graphs of bounded degree.

[1] The set K_1 corresponds to the root.

Theorem 46. *Consider bootstrap percolation on a graph G with maximum degree Δ and $r > \frac{\Delta}{2}$ and a set of initially infected vertices A_0. Then the set A_f of vertices that are infected at the end of the process fulfills*

$$|A_f| \leq \left(1 + \frac{\Delta}{2r - \Delta}\right) |A_0|.$$

Proof. Similarly to the proof of Theorem 41, we make use of a potential Φ_t which shall serve as an upper bound on yet uninfected vertices in the neighborhood of all infected vertices. We let $\Phi_t = \sum_{v \in V} \Phi_t^v$, with

$$\Phi_t^v := \begin{cases} 0, & \text{if} \quad v \notin A_t \\ |\delta(v) \setminus A_t|, & \text{else} \end{cases}.$$

With this potential, we overestimate the number of uninfected vertices that potentially can be infected in the next time step.

Observe, if at time t a vertex becomes infected, we have $\Phi_t^v = |\delta(v) \setminus A_t|$. Additionally, each of v's neighbors potential is updated accordingly, i.e. $\Phi_t^w \leq \Phi_{t-1}^w - |\delta(w) \cap (A_t \setminus A_{t-1})|$. Since $r \geq \frac{\Delta}{2}$, we have that each new infection at time t of some vertex v the global potential changes by

$$+|\delta(v) \setminus A_t| - |A_{t-1} \cap \delta(v)| \leq (\deg(v) - r) - r \leq \Delta(G) - 2r < 0.$$

Thus, the total number of infections is bounded by

$$|A_0| + \frac{\Delta |A_0|}{2r - \Delta} = \left(1 + \frac{\Delta}{2r - \Delta}\right) |A_0|.$$

□

With Theorem 46 in mind, we work towards an answer to a question raised by Morris regarding minimal percolating sets [71, Problem 1]. These sets are defined as follows.

Definition 47 (Minimal percolating sets)**.** *Given a graph $G = (V, E)$. We call a set $A \subseteq V$ a minimal percolating set if every proper subset of A does not percolate.*

Question ([71, Problem 1])**.** *Does there exist a sequence of n-vertex graphs $(G_n)_{n \in \mathbb{N}}$ of bounded maximum degree Δ, such that $E(G_n, r) = o(n)$. Here,*

$$E(G, r) = \max\{|A_0| : A_0 \subseteq V_G \text{ minimal percolating in } r\text{-neighbor bootstrap percoaltion}\}.$$

The answer is twofold. If $r \geq \Delta + 1$, the answer is obviously no. Even more, by definition every minimal percolating set is a percolating set. Therefore, if the graph

sequence not necessarily has bounded degree but bounded degeneracy $d \leq r-1$, we may apply Corollary 44 and find that percolating sets must be of size $\Theta(n)$. Also, due to Theorem 46, if $r > \frac{\Delta}{2}$, we obtain that percolating sets must be of size at least $(1 - \frac{\Delta}{2r})n = \Theta(n)$.

But, if $r \leq \frac{\Delta}{2}$, the following graph sequence has the desired property. The construction is somewhat akin to the construction in Figure 3.3.

Let G_{lr} consist of a sequence of l independent sets $I_1, \ldots, I_l$ each of size r. Let every vertex in I_i be adjacent to every vertex in I_{i+1} for $i \in [l-1]$. The maximum degree in this graph is $2r$.[2]

Observe, a smallest percolating set is of size r. Additionally, every minimal percolating set may not contain more than r vertices: If one vertex $v \in I_i$ becomes infected in the first time step, all other vertices in I_i become infected as well. This set I_i then suffices to infect the remaining vertices.

3.1.3 Grids

In this subsection, we study the size of percolating sets in bootstrap percolation on an $n \times m$ grid[3]. We are interested in the maximum size of minimal percolating sets on grids (denoted by $E(m,n)$ or $E(n)$ for an $n \times m$ and $n \times n$ grid graph, respectively) and improve known bounds on their size, disproving a conjecture on the size of minimal percolating sets by Morris [71]. Throughout the remainder of the section we assume $r = 2$.

Definition 48 (Special sets on grids)**.** *Let G be an $n \times m$ grid graph. We say a subset $A \subset V$ spans a rectangle if the set of vertices infected with $A_0 = A$ forms a rectangle. We denote this rectangle by $\boxed{A}$* [4]*.*

Additionally, given an n' times m' rectangle $\boxed{R}$ we denote by

$$\boxed{R}^{\urcorner}_{[k]\times[l]} := \{(n'+1-i, m'+1-j) : i \in [k], j \in [l]\},$$
$$\boxed{R}^{\ulcorner}_{[k]\times[l]} := \{(i, m'+1-j) : i \in [k], j \in [l]\},$$
$$\boxed{R}^{\llcorner}_{[k]\times[l]} := \{(i, j) : i \in [k], j \in [l]\},$$
$$\boxed{R}^{\lrcorner}_{[k]\times[l]} := \{(n'+1-i, j) : i \in [k], j \in [l]\},$$

the four k times l sized corners of $\boxed{R}$.
We call a set A one-(k,l)-corner-avoiding if A is percolating and there exists a k times

[2] Filling in $k = n \mod r$ isolated vertices to account for parity, we obtain graphs for all values of n.
[3] We interpret the corresponding graph as a subgraph of $\mathbb{N}^2$ and use coordinates to describe the vertices accordingly. Therefore, we have n columns and m rows
[4] Since for a percolating set A infects all vertices, we have A spans a rectangle, i.e. $\boxed{A} = V$. Also note, any subset $A \subset V$ consists of a collection of sets each spanning a rectangle.

l sized corner, such that no proper subset of A infects one of the vertices in this corner. To be more precise, one of the corners $\boxed{A}^{\urcorner}_{[k]\times[l]}, \boxed{A}^{\ulcorner}_{[k]\times[l]}, \boxed{A}^{\llcorner}_{[k]\times[l]}$, or $\boxed{A}^{\lrcorner}_{[k]\times[l]}$ must remain completely uninfected for any proper subset $A' \subset A$.

We call a set A two-(k,l)-(k',l')-corner-avoiding if A is percolating and there exist two opposing corners, of size k times l and k' times l', that remain uninfected for any proper subset of A. I.e., one of

$$\boxed{A}^{\urcorner}_{[k]\times[l]} \cup \boxed{A}^{\llcorner}_{[k']\times[l']},$$
$$\boxed{A}^{\ulcorner}_{[k]\times[l]} \cup \boxed{A}^{\lrcorner}_{[k']\times[l']},$$
$$\boxed{A}^{\llcorner}_{[k]\times[l]} \cup \boxed{A}^{\urcorner}_{[k']\times[l']},$$
$$\boxed{A}^{\lrcorner}_{[k]\times[l]} \cup \boxed{A}^{\ulcorner}_{[k']\times[l']}$$

remains completely uninfected for any proper subset $A' \subset A$.
If A is one-$(1,1)$-corner-avoiding, we refer to A simply as corner-avoiding. [5]

Morris proved that $\lim_{n\to\infty} \frac{E(n)}{n^2}$ exists and lies within the interval $[\frac{4}{33}, \frac{1}{6}]$. However, he conjectured that $\lim_{n\to\infty} \frac{E(n)}{n^2} = \frac{4}{33}$ [71, Conjecture 1]. We prove that $\lim_{n\to\infty} \frac{E(n)}{n^2} \geq \frac{2}{15}$, disproving Morris' conjecture.

Theorem 49. *For every $3 \leq m, n \in \mathbb{N}$ we have that*

$$\frac{2}{15}mn - \frac{8}{15}m - n(2\log n + \sqrt{n}\log n) \leq E(n,m)$$

We start working towards a proof of the lower bound and give bounds on the size of corner-avoiding sets which use to construct larger minimal percolating sets recursively. For that matter, let $E_{[k]\times[l]}(n,m)$ denote the maximum number of vertices in any one-(l,k)-corner-avoiding set on an $m \times n$ grid.

Lemma 50. *Let $n = 2$. For every $2 \leq m \in \mathbb{N}$, we have that*

$$E_{[1]\times[1]}(2,m) \geq \frac{2}{3}m.$$

Proof. Let

$$m' := \begin{cases} m & \text{if } 0 \equiv m \mod 3 \\ m-1 & \text{else.} \end{cases}$$

[5] Note the difference to [71], where the definition of corner-avoiding sets matches our definition of two-$(2,2)$-$(2,2)$-corner-avoiding sets

Then, define

$$A = \big\{(1,i) : i \in [m'] \ \wedge \ i \equiv 1 \mod 3\big\} \cup \big\{(1,i) : i \in [m'] \ \wedge \ 0 \equiv 1 \mod 3\big\} \cup \{(2,m)\}.$$

It is easy to see (c.f. Figure 3.4) that A is an one-corner-avoiding set with $|A| \geq \frac{2}{3}m$, so the statement of the lemma follows. □

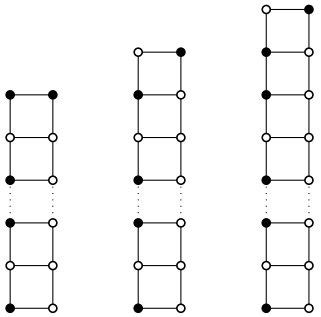


Figure 3.4: Visualization of A (black vertices) for all three cases, from left to right: $0 \equiv m \mod 3$, $1 \equiv m \mod 3$, and $2 \equiv m \mod 3$. Note, all these sets are one-$(1,2)$-corner-avoiding.

The next lemma is key to combine two corner-avoiding sets to obtain a larger one:

Lemma 51. *Let $m \geq 2$ and $t \geq 1$. Then*

$$E_{[1]\times[1]}(5 \cdot 2^t - 3, m) \geq \frac{2}{15} m(5 \cdot 2^t) - (5 \cdot 2^t)^{\frac{3}{2}} \log(5 \cdot 2^t)$$

Proof. We prove the lemma by induction on t. For $t = 1$ and $m = 2$, Lemma 50 gives the desired result. Also, for $t = 1$ and $m \in \{3,4,5\}$ adding to the constructions in Lemma 50 one, two, or three empty columns and adding $\{(n,1)\}$, $\{(n,1)\}$, and $\{(n,1),(n-1,1)\}$ to the set A respectively, gives the result.

So let now $t \geq 2$ and $m \geq 6$, then

$$\begin{aligned}
E_{[1]\times[1]}(5 \cdot 2^t - 3, m) &\overset{*}{\geq} E_{[1]\times[1]}(5 \cdot 2^{t-1} - 3, m-2) + E_{[1]\times[1]}(5 \cdot 2^{t-1} - 3, m-2) + 2 \\
&\geq \frac{4}{15}(m-2)(5 \cdot 2^{t-1}) - 2(5 \cdot 2^{t-1})^{\frac{3}{2}} \log(5 \cdot 2^{t-1}) + 2 \\
&\geq \frac{2}{15} m(5 \cdot 2^t) - \frac{4}{15}(5 \cdot 2^t) - 2(5 \cdot 2^{t-1})^{\frac{3}{2}} \log(5 \cdot 2^{t-1}) \\
&\geq \frac{2}{15} m(5 \cdot 2^t) - (5 \cdot 2^t)\left(\frac{4}{15} + (5 \cdot 2^{t-1})^{\frac{1}{2}} \log(5 \cdot 2^{t-1})\right)
\end{aligned}$$

where $*$ follows in the same vein as in [71, Lemma 4]. A visualization of the construction, combining two one-corner-avoiding sets of respective dimensions into a new one,

can be seen in Figure 3.5.

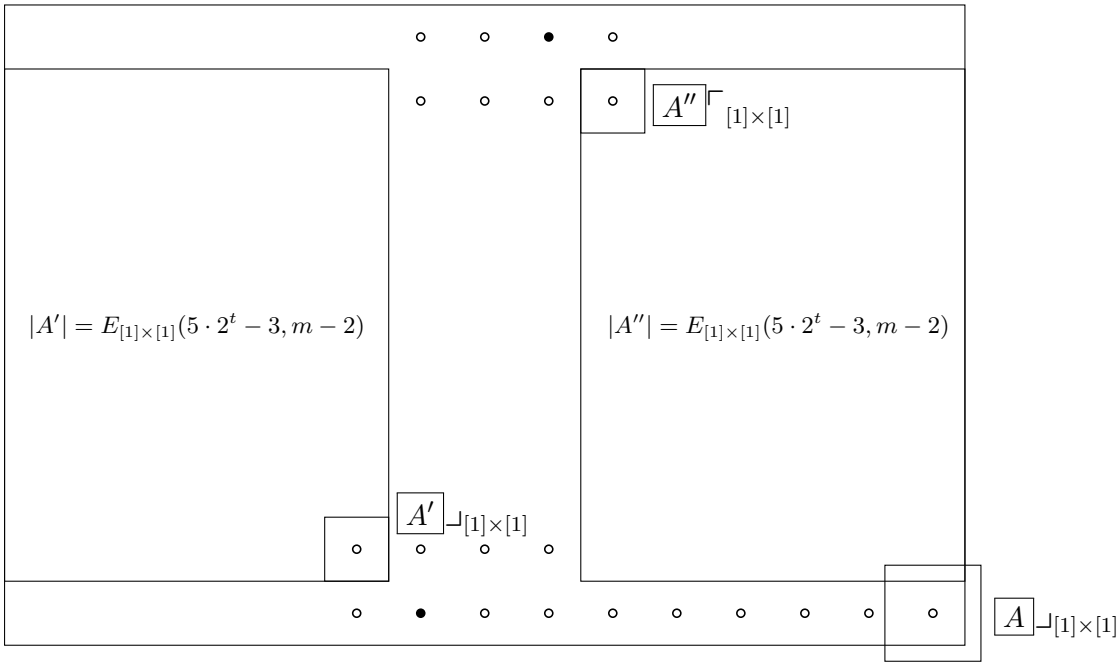


Figure 3.5: Combining two one-corner-avoiding sets into a new one adding two rows and three columns. The new one-corner-avoiding set contains both sets A' and A'' as well as the two (black) vertices.

Finally, note that

$$\begin{aligned}\frac{4}{15} + (5 \cdot 2^{t-1})^{\frac{1}{2}} \log(5 \cdot 2^{t-1}) &= \frac{4}{15} + \frac{1}{\sqrt{2}}(5 \cdot 2^t)^{\frac{1}{2}} \log(5 \cdot 2^{t-1}) \\ &= \frac{2}{15} - \frac{1}{\sqrt{2}}(5 \cdot 2^t)^{\frac{1}{2}} \log 2 + \frac{1}{\sqrt{2}}(5 \cdot 2^t)^{\frac{1}{2}} \log(5 \cdot 2^t) \\ &\overset{t\geq 1}{\leq} (5 \cdot 2^t)^{\frac{1}{2}} \log(5 \cdot 2^t),\end{aligned}$$

and thus

$$E_{[1]\times[1]}(5 \cdot 2^t - 3, m) \geq \frac{2}{15} m(5 \cdot 2^t) - (5 \cdot 2^t)^{\frac{3}{2}} \log(5 \cdot 2^t).$$

□

The next lemma implies the lower bound of Theorem 49.

Lemma 52. *For $m, n \geq 2$ we have that*

$$E(n, m) \geq \frac{2}{15} mn - \frac{8}{15} m - n(2 \log n + \sqrt{n} \log n).$$

Proof. Let $t_1, t_2, \dots t_l \in \mathbb{N}$ uniquely encode $\lfloor \frac{n}{5} \rfloor$ in the following way: Let t_1 be the largest integer such that $n \geq 5 \cdot 2^{t_1}$ and for all $i > 1$, let t_i be the largest integer such that $n - \sum_{j=1}^{i-1}(5 \cdot 2^{t_j}) \geq 5 \cdot 2^{t_i}$. Then, iteratively using the construction from Lemma 51, we have

$$\begin{aligned}
E(n,m) &\geq E_{[1]\times[1]}(5 \cdot 2^{t_1} - 3, m-2) + E_{[1]\times[1]}(n - 5 \cdot 2^{t_1}, m-2) \\
&\geq E_{[1]\times[1]}(5 \cdot 2^{t_1} - 3, m-2) \\
&\quad + \big(E_{[1]\times[1]}(5 \cdot 2^{t_2} - 3, m-4) + E_{[1]\times[1]}(n - 5 \cdot 2^{t_1} - 5 \cdot 2^{t_2}, m-4)\big) \\
&\geq \dots \overset{*}{\geq} E_{[1]\times[1]}(5 \cdot 2^{t} - 3, m-2) \\
&\quad + \big(E_{oc}(5 \cdot 2^{t_2} - 3, m-4) + \dots + E_{[1]\times[1]}(5 \cdot 2^{t_l} - 3, m-2l)\big) \\
&\geq E_{[1]\times[1]}(5 \cdot 2^{t} - 3, m-2l) \\
&\quad + \big(E_{[1]\times[1]}(5 \cdot 2^{t_2} - 3, m-2l) + \dots + E_{[1]\times[1]}(5 \cdot 2^{t_l} - 3, m-2l)\big) \\
&\geq \sum_{i=1}^{l} \frac{2}{15}(m-2l)(5 \cdot 2^{t_i}) - (5 \cdot 2^{t_i})^{\frac{3}{2}} \log(5 \cdot 2^{t_i}) \\
&\geq \frac{2}{15} m(n-4) - \sum_{i=1}^{l} 5 \cdot 2^{t_i}(2l + (5 \cdot 2^{t_i})^{\frac{1}{2}} \log(n)) \\
&\geq \frac{2}{15} mn - \frac{8}{15} m - n(2 \log n + \sqrt{n} \log n).
\end{aligned}$$

Note, for $*$ to hold in the last step, instead of inserting 3 empty columns, we insert additional $n - \sum_{i=1}^{l} 5 \cdot 2^{t_l} \leq 4$ empty columns as shown in Figure 3.6. Furthermore, $\sum_{i=1}^{l} 5 \cdot 2^{t_i} \geq n - 4$.

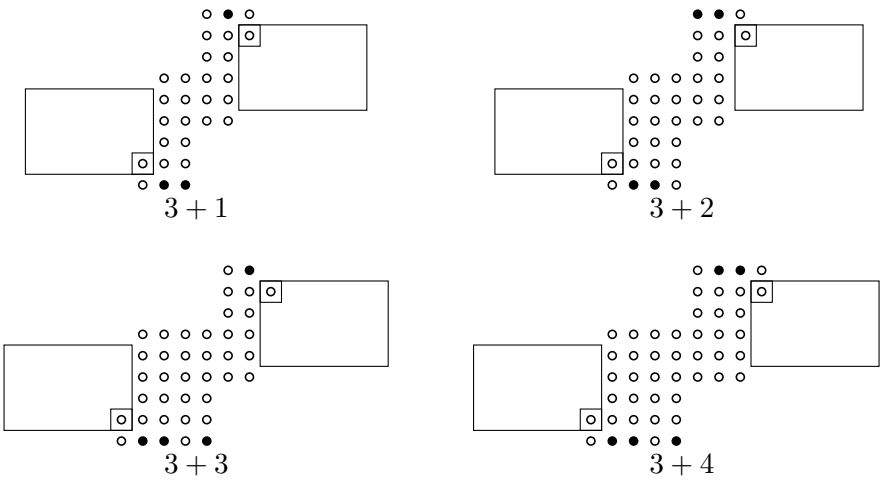


Figure 3.6: Illustration of combining two corner-avoiding sets into a new one, inserting $1, 2, 3$, or 4 additional columns in the last step to obtain a rectangle of width n.

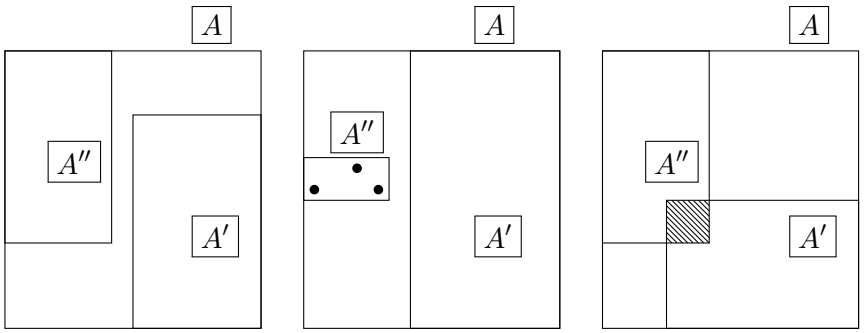


Figure 3.7: Visualization of possible partitions according to Lemma 53

□

Next, we work towards a new upper bound. The main idea is an inductive proof making use of the following observation. Given a minimal percolating set, we can partition this set into several smaller percolating sets. In particular, we can partition it into a sequence of percolating sets that are corner avoiding as the next lemma shows. However, we yet only provide proofs of the bound for grids of size m times n where $m \leq 5$, but we conjecture that $\frac{2}{15}$ is the correct asymptotic constant.

Lemma 53 (Extension Lemma)**.** *Given an m times n grid graph. Let A be a minimal percolating set.*

1. *Then A can be partitioned into two subsets A' and A'' such that A' and A'' both span a rectangle with $A' \cap \boxed{A''} = \emptyset$ and $\boxed{A'} \cap A'' = \emptyset$.*
2. *Furthermore, let $A^* \subset A$ be a subset that spans a rectangle of width and height at least 2. Then there exists a partition $A' \cup A'' = A$ with the following properties. First, $A^* \subseteq A'$ and both, A' and A' span rectangles, $\boxed{A'}$ and $\boxed{A''}$, respectively. Second, $A' \cap \boxed{A''} = \emptyset$ and $\boxed{A'} \cap A'' = \emptyset$. Additionally,*
   - *A' spans a rectangle containing two neighboring corners of $\boxed{A}$ and A'' is a singleton, or*
   - *A' and A'' each span a rectangle and contain opposing corners of $\boxed{A}$. In this case, A'' is a singleton or corner-avoiding.*

Proof. 1.) Observe first, the trivial partition of A into singletons forms a partition of A such that each set spans a rectangle. Trivially, the empty intersection condition is met.

Given any partition $\bigcup_i A^i = A$ such that all A^i span a rectangle. Since A is percolating, at least two of the rectangles $\boxed{A^{i_1}}$, $\boxed{A^{i_2}}$ must meet, i.e., there must exist a vertex

$v \in V$ such that $\operatorname{dist}(v, \boxed{A^{i_1}}) = \operatorname{dist}(v, \boxed{A^{i_2}}) = 1$, for some $i_1 \neq i_2$. Then $A^{i_1} \cup A^{i_2}$ spans a rectangle and we obtain a partition with fewer sets. Assume that there exists A^j and $v \in V$ such that $v \in \boxed{A^{i_1} \cup A^{i_2}} \cap A^j$ or $v \in A^{i_1} \cup A^{i_2} \cap \boxed{A^j}$. Then, since we have a partition of A, either $v \in A^j$ and $v \notin (A^{i_1} \cup A^{i_2})$, or $v \in (A^{i_1} \cup A^{i_2})$ and $v \notin A^j$. In either case, $\boxed{(A^{i_1} \cup A^{i_2}) \cup A^j \setminus \{v\}} = \boxed{(A^{i_1} \cup A^{i_2}) \cup A^j}$ proving redundancy of v. Since A is minimal percolating, this leads to a contradiction. Finally, proceed until only two sets remain.

2.) For the next part, simply let $A' \subset A$ be the largest proper subset of A spanning a rectangle with $A^* \subseteq A'$. Observe that A' together with all singletons in $A \setminus A'$ forms a partition of A meeting all requirements we used in the first part of the proof. Therefore, we can reduce its size to two sets, again, meeting all requirements. Since A' is maximum, we can never join A' with any other subset, therefore we obtain a partition of A with $A^* \subseteq A'$, $A' \cap \boxed{A''} = \emptyset$, and $\boxed{A'} \cap A'' = \emptyset$.

Next, we prove that A' and A'' fulfill one of the additional properties.

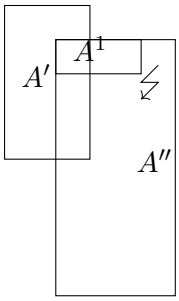
Observe, A must contain at least one vertex on every side of $\boxed{A}$, and therefore by the way we choose to extend A', A' must contain at least one corner of $\boxed{A}$, w.l.o.g. $\boxed{A}_{\lrcorner[1]\times[1]} \in \boxed{A'}$. Note, $\boxed{A}^{\ulcorner}_{[1]\times[1]} \notin \boxed{A'}$ because this would imply $\boxed{A'} = \boxed{A}$. But then, if $\boxed{A'}$ does not contain another corner, $\boxed{A''}$ must be adjacent to the two other sides of $\boxed{A}$ and therefore contains the opposing corner.

So let $|A''| \geq 2$, then A'' must be corner-avoiding:

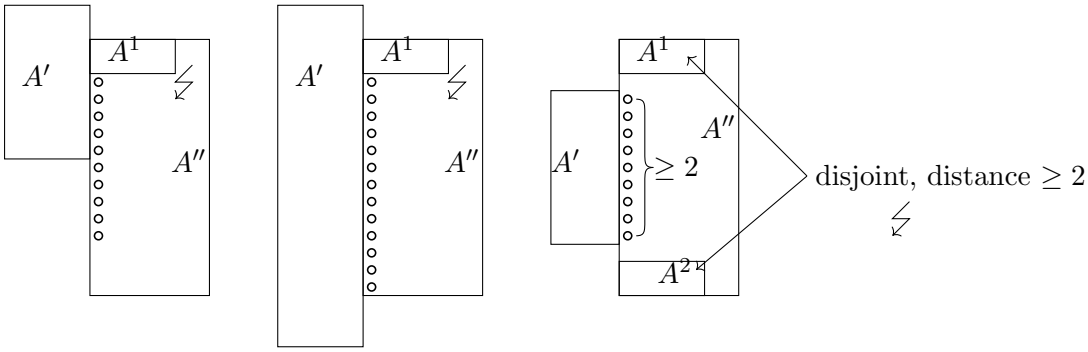
If $\boxed{A'}$ and $\boxed{A''}$ overlap we immediately have that $\boxed{A'}$ contains one corner of $\boxed{A''}$ (and vice versa). W.l.o.g let this corner be $\boxed{A''}_{\lrcorner[1]\times[1]}$. If A'' was not one-$(1,1)$-corner-avoiding, A' would not have been maximal.

Thus, assume the spanned rectangles $\boxed{A'}, \boxed{A''}$ do not overlap and assume for contradiction, that A'' is not corner-avoiding. Consider the closest corner of $\boxed{A''}$ to $\boxed{A'}$. W.l.o.g. let this corner be $\boxed{A''}^{\ulcorner}_{[1]\times[1]}$ and let $A^1 \subset A''$ be the largest subset infecting this corner. Since A' is maximum, $\operatorname{dist}(\boxed{A'}, \boxed{A''}^{\ulcorner}_{[1]\times[1]}) \geq 2$ (see Figure 3.8). Thus, there exists an adjacent corner of $\boxed{A''}$ as depicted in Figure 3.8, w.l.o.g. let this corner be $\boxed{A''}_{\llcorner[1]\times[1]}$, for which we find another maximal set $A^2 \subset A''$ that infects the corner. Now, $A^1 \cup A^2 = A''$ with $A^1 \cap A^2 = \emptyset$, and $\operatorname{dist}(\boxed{A^1}, \boxed{A^2}) \geq 2$, which is a contradiction to $\boxed{A''} = \boxed{A^1 \cup A^2}$.

$\text{dist}(A', A'') = 0$



$\text{dist}(A', A'') = 1$:



$\text{dist}(A', A'') = 2$:

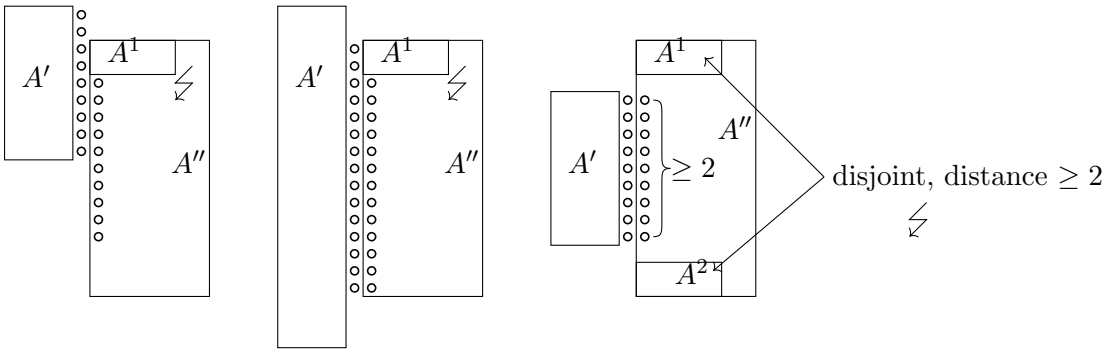


Figure 3.8: The three cases in the proof of Lemma 53. Note, $\boxed{A'}$ has height and width at least 2.

□

Lemma 54 ([71, Theorem 9])**.** *We have that*

- $E(1,m) = \left\lfloor \frac{2(m+1)}{3} \right\rfloor$
- $E(2,m) = \left\lfloor \frac{2(m+2)}{3} \right\rfloor$
- $E(3,m) = \left\lfloor \frac{2(m+3)}{3} \right\rfloor$

For later purposes, we additionally need the following lemma.

Lemma 55. *For all $m \in \mathbb{N}$ we have that $E(4,m) \leq \frac{8}{15}m + \frac{4}{15}(m+4) + \frac{20}{15} = \frac{4}{5}m + \frac{12}{5}$. This implies that $E(4,m) \leq \frac{2}{15} \cdot 4m + \frac{6}{15}(m+4) + \frac{2}{3}$ for all $m \in \mathbb{N}$.*

As we are going to need them in the following, below a table with all values of $E(m,n)$ for $m \leq 11$, $n \leq 4$. Note that the bound from Lemma 55 only gives $E(4,11) \leq 11$, however, the exact value is 10 (see Remark 56).

n \ m	1	2	3	4	5	6	7	8	9	10	11	...	
1	1	2	2	3	4	4	5	6	6	7	8	...	Lemma 54
2	2	2	3	4	4	5	6	6	7	8	8	...	
3	2	3	4	4	5	6	6	7	8	8	9	...	
4	3	4	4	5	6	7	8	8	9	10	*10*	...	Lemma 55/Remark 56

Proof of Lemma 55. We prove the bounds by induction on m. For $m \leq 3$, the statement is true due to [71, Theorem 9]. So, let $m \geq 4$ and A be a maximum minimal percolating set of $[4] \times [m]$, i.e. of size $E(4,m)$.

Assume that A does not contain a proper subset A' such that $\boxed{A'}$ is of width 4. Observe that in this case there exists a partition of A into two proper subsets A' and A'' (possibly one of them consisting of a singleton) that span two rectangles of dimensions $[k'] \times [m']$ and $[k''] \times [m'']$, respectively. W.l.o.g. assume that $k' \geq k''$. From all these possible partitions, choose the one maximizing m'.

Then, if $k' = k'' = 3$, we have that $\mathrm{dist}(\boxed{A'}, A'') \geq 2$ (else A contains a set A' spanning a rectangle of width 4, or we can increase the size of m'). Additionally, A may not contain two consecutive rows that do not contain a vertex in A, thus

$m' + m'' = m - 1$. It follows that

$$E(4,m) \leq E(3,m') + E(3,m'') \leq \left\lfloor \frac{2}{3}(m'+3) \right\rfloor + \left\lfloor \frac{2}{3}(m''+3) \right\rfloor$$

$$\leq \begin{cases} 4 \leq \frac{4}{5}m + \frac{12}{5} & \text{for } m' = 1, m'' = 1, \text{ and } m = 3, \\ 5 \leq \frac{4}{5}m + \frac{12}{5} & \text{for } m' = 2, m'' = 1, \text{ and } m = 4, \\ 6 \leq \frac{4}{5}m + \frac{12}{5} & \text{for } m' = 3, m'' = 1, \text{ and } m = 5, \\ 6 \leq \frac{4}{5}m + \frac{12}{5} & \text{for } m' = 2, m'' = 2, \text{ and } m = 5, \\ 7 \leq \frac{4}{5}m + \frac{12}{5} & \text{for } m' = 3, m'' = 2, \text{ and } m = 6, \\ \frac{2}{3}(m-1+6) = \frac{2}{3}(m+5) \leq \frac{4}{5}m + \frac{12}{5} & \text{for } m \geq 7. \end{cases}$$

If $k' = 3$ and $k'' = 2$, we again have that $\text{dist}(\boxed{A'}, A'') \geq 2$. Additionally, A may not contain two consecutive rows that do not contain a vertex in A, thus again $m' + m'' = m - 1$. Note, $E(k,m) \leq E(l,m)$ for all $k \geq l$, in particular $E(2,m'') \leq E(3,m'')$. Thus, the claim follows by the same calculation as above.

Next, if $k' \in \{2,3\}$ and $k'' = 1$, this implies that $m'' = 1$. Otherwise, $|A''| \geq 2$ and thus $\text{dist}(\boxed{A'}, A'') \geq 2$, which implies that $\text{dist}(\boxed{A'}, \boxed{A''}) \geq 2$. But then, the union $A' \cup A'' = A$ would not be percolating. So let $k' = 3$ and $k'' = m'' = 1$, then, since $m \geq 4$, we have that

$$E(4,m) \leq E(3,m') + 1 = \left\lfloor \frac{2}{3}(m'+3) \right\rfloor + 1 \leq \frac{4}{5}m + \frac{12}{5}.$$

The case for $k' = 2$ and $k'' = 1$ follows from $E(2,m) \leq E(3,m)$.

Finally, assume that $k' = k'' = 2$. Then, we obtain $\text{dist}(\boxed{A'}, \boxed{A''}) = 1$ from $\text{dist}(\boxed{A'}, A'') \geq 2$ and thus $m' + m'' = m$. Hence,

$$E(4,m) = E(2,m') + E(2,m'') = \left\lfloor \frac{2}{3}(m'+2) \right\rfloor + \left\lfloor \frac{2}{3}(m''+2) \right\rfloor \leq \frac{2}{3}(m+4)$$
$$\overset{m \geq 4}{\leq} \frac{4}{5}m + \frac{12}{5}.$$

Thus, if A does not contain a proper subset A' spanning a rectangle of width 4, the statement is true.

Now, for the remainder of the proof, let $A' \subset A$ be such a set spanning a rectangle of width 4, i.e. of dimensions $[4] \times [m']$. Among all possible choices, let A' be such that $\boxed{A'}$ has maximum distance to $(4,m)$ and $(4,1) \in \boxed{A'}$ (We assume that this set A' exists, as if it does not exist, interchange labels 1 and m).

If the rectangle spanned by A' does not contain row $(m-1)$ or m, there must exist

a set $A'' \subset A$ that spans $[4] \times [m-3]$ or $[4] \times [m-2]$. In either case, by induction, we have that

$$E(4,m) \leq \begin{cases} E(4,m-3)+2 \leq \frac{4}{5}(m-3)+\frac{12}{5}+2 \leq \frac{4}{5}m+\frac{12}{5}, \\ E(4,m-2)+1 \leq \frac{4}{5}(m-2)+\frac{12}{5}+1 \leq \frac{4}{5}m+\frac{12}{5}. \end{cases}$$

Finally, in the remaining case, we have that $\boxed{A'} = [4] \times [m-1]$ and A' does not contain any set A'' spanning a rectangle of width 4. Thus, just like A, the set A' may be partitioned into two sets A'' and A''' spanning two rectangles of dimensions $[k''] \times [m'']$ and $[k'''] \times [m''']$. Assume that A'' is chosen in such a way that m'' is maximal.

Observe that, if $k'' = k''' = 3$, $A \setminus A'$ cannot be a set spanning a rectangle of width 4, and neither can be $A \setminus A''$. But then, either $A \setminus A'$ or $A \setminus A''$ spans a rectangle of dimension $[k''] \times [m''+1]$ or $[k'''] \times [m'''+1]$, respectively. In either case we have $(m''+m'''+1) = m-1$ may apply the same calculation as above to derive that

$$E(4,m) \leq E(3,m''+1) + E(3,m''') \leq \frac{4}{5}m + \frac{12}{5}.$$

If $\min\{k'', k'''\} \leq 2$, let w.l.o.g. $k'' \leq 2$. We have that $m''+m''' \leq m-2$ and hence,

$$E(4,m) \leq E(2,m'') + E(3,m''') + 1 = \left\lfloor \frac{2}{3}(m''+2) \right\rfloor + \left\lfloor \frac{2}{3}(m'''+3) \right\rfloor + 1$$

$$\leq \begin{cases} 5 \leq \frac{4}{5}m+\frac{12}{5} & \text{for } m''=1, m'''=1, \text{ and } m=4, \\ 6 \leq \frac{4}{5}m+\frac{12}{5} & \text{for } m''=1, m'''=2, \text{ and } m=5, \\ 7 \leq \frac{4}{5}m+\frac{12}{5} & \text{for } m''=1, m'''=3, \text{ and } m=6, \\ 6 \leq \frac{4}{5}m+\frac{12}{5} & \text{for } m''=2, m'''=2, \text{ and } m=6, \\ 7 \leq \frac{4}{5}m+\frac{12}{5} & \text{for } m''=2, m'''=3, \text{ and } m=7, \\ \frac{2}{3}(m-2+5)+1 = \frac{2}{3}m+3 \leq \frac{4}{5}m+\frac{12}{5} & \text{for } m \geq 7. \end{cases}$$

This finishes the proof of Lemma 55. □

Remark 56. With the same analysis, we can prove an even stronger statement for $m \geq 5$. Together with an easy construction we can actually prove that $E(4,m) = \lfloor \frac{2(m+5)}{3} \rfloor$ for all $m \geq 5$.

Lemma 57. *For all $m \in \mathbb{N}$ we have that $E(5,m) \leq \frac{10}{15}m + \frac{4}{15}(m+5) + \frac{4}{3} = \frac{14}{15}m + \frac{8}{3}$.*

Proof. For $m \leq 4$, the statement is true due to [71, Theorem 9] together with Lemma 55. So, let $m \geq 5$ and A be a maximum minimal percolating set of $[5] \times [m]$, i.e., of size $E(5,m)$.

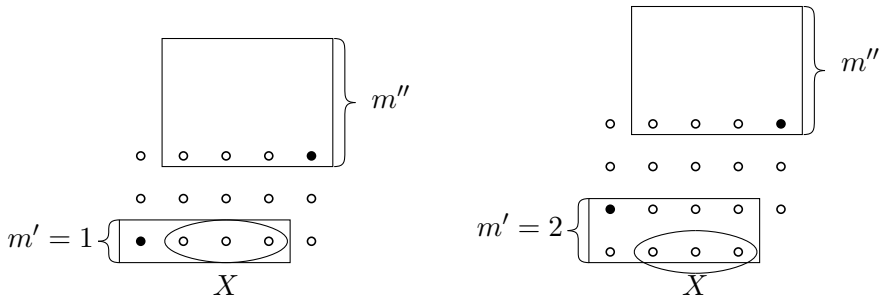


Figure 3.9: If the set X contains more than 2 vertices in A', which it would have to in order to be spanning the rectangle $[4] \times [1]$ and $[4] \times [2]$, respectively, one vertex would be obsolete.

We proceed with the same case distinctions as in the proof of Lemma 55. Assume first that A does not contain a proper subset A' spanning a rectangle of width 5. Observe that in this case there exists a partition of A into two proper subsets A' and A'' (possibly consisting of a singleton) that span two rectangles of dimensions $[k'] \times [m']$ and $[k''] \times [m'']$, respectively. W.l.o.g. assume that $k' \geq k''$. Finally, from all such possible partitions choose the one with maximal m'.

Then, if $k' = 4$ and $k'' \in \{2, 3, 4\}$, we have that $\mathrm{dist}(A', A'') \geq 2$ (else A contains a set A' spanning a rectangle of width 5, or we can increase the size of m' — note $k'' > 1$, so $|A''| \geq 2$). Additionally, A may not contain two consecutive rows that do not contain a vertex in A, thus we necessarily have $m' + m'' = m - 1$. Note once more, $E(k, m) \leq E(l, m)$ for all $k \geq l$, therefore, it immediately follows that

$$E(5, m) \leq E(4, m') + E(4, m'') \leq \frac{4}{5}m' + \frac{12}{5} + \frac{4}{5}m'' + \frac{12}{5} = \frac{4}{5}m + 4.$$

Since $E(5, m)$ must be integral, we actually have $E(5, m) \leq \lfloor \frac{4}{5}m + 4 \rfloor$. This integral condition implies that $E(5, m) \leq \frac{14}{15}m + \frac{8}{3}$ for all $m \geq 6$. Observe, if $m = n = 5$, it is impossible to partition the set A into two sets A', A'' that span disjoint rectangles of dimensions $[4] \times [m']$ ($m' \geq 2$) and $[4] \times [2]$, respectively. If you could, one vertex in A'' must be obsolete. So, in case $n = m = 5$, it follows that $k'' \leq 3$ (c.f. Figure 3.9) and therefore

$$E(5, m) \leq E(4, m') + E(3, m'') \leq \begin{cases} 3 + 4 \leq \frac{14}{15}m + \frac{8}{3} & \text{if } m' = 1,\ m'' = 3 \\ 3 + 3 \leq \frac{14}{15}m + \frac{8}{3} & \text{if } m' = 2,\ m'' = 2 \\ 4 + 2 \leq \frac{14}{15}m + \frac{8}{3} & \text{if } m' = 3,\ m'' = 1, \end{cases} \tag{3.1}$$

completing the case for $k' = 4, k'' \in \{2, 3, 4\}$.

If $k' = 4$ and $k'' = 1$, we have $|A''| = 1$ and thus,

$$E(5,m) \leq E(4,m') + 1 \leq \left\lfloor \frac{4}{5}m + \frac{12}{5} \right\rfloor + 1 \overset{m\geq 5}{\leq} \frac{14}{15}m + \frac{8}{3}.$$

If $k' = 3$ and $k'' \in \{2,3\}$, we have

$$E(5,m) \leq E(3,m') + E(3,m'') = \frac{2}{3}(m'+3) + \frac{2}{3}(m''+3) \leq \frac{2}{3}(m+6) \leq \frac{2}{3}m + 4 \overset{m\geq 5}{\leq} \frac{14}{15}m + \frac{8}{3},$$

while, if $k' = 3$ and $k'' = 1$, we have that

$$E(5,m) \leq E(3,m') + 1 \leq \frac{2}{3}(m+3) + 1 \overset{m\geq 5}{\leq} \frac{14}{15}m + \frac{8}{3}.$$

Finally, if $k' = 2$ and $k'' = 2$, $m' + m'' \leq m + 2$ (both spanned rectangles must overlap in one and may overlap in most two rows), and thus

$$E(5,m) \leq E(2,m') + E(2,m'') = \frac{2}{3}(m'+1) + \frac{2}{3}(m''+1) \leq \frac{2}{3}m + \frac{8}{3} \leq \frac{14}{15}m + \frac{8}{3}.$$

On the other hand, if $k' = 2$ and $k'' = 1$, we have that $m'' = 1$, and thus

$$E(5,m) \leq E(2,m') + 1 = \frac{2}{3}(m+1) + 1 \leq \frac{14}{15}m + \frac{8}{3}.$$

Note that $k' = k'' = 1$ is not possible.

So, assume that A contains a proper subset A' spanning a rectangle of width 5, i.e. of dimensions $[5] \times [m']$. Just like in the proof of Lemma 55 we choose the set A' in such a way that the spanned rectangle contains row 1 or m but maximizes the distance to row m or 1, respectively, i.e. minimizes m' containing row 1 or row m. W.l.o.g. let it contain row 1.

If $m' < m - 1$, we have that

$$\begin{aligned} E(5,m) &\leq E(5,m') + \frac{2}{3}(m - m' + 2) - 1 \\ &\leq \frac{14}{15}m' + \frac{8}{3} + \frac{2}{3}(m - m') + \frac{1}{3} = \frac{14}{15}m - \frac{4}{15}(m - m') + \frac{1}{3} \\ &\leq \frac{14}{15}m + \frac{8}{3}. \end{aligned}$$

If $m' = m - 1$, we can partition A' into two non-empty sets A'' and A''', such that A'' and A''' span two rectangles of dimensions $[k''] \times [m'']$ and $[k'''] \times [m''']$ with $k'' \geq k'''$.

Additionally, assume that A'' is chosen in such a way that m'' is maximal.

Then, if $k'' = k''' = 4$, by the choice of A' we have that $A \setminus A'$ spans a rectangle of width at most 4, which otherwise, contradicts the choice of A' (because $A \setminus A'$ then forms a set spanning a rectangle of dimensions $[5] \times [m'' + 1]$, where $m''' + 1 < m'' + m''' + 1 = m'$). So, together with the fact that $m'' + m''' + 1 = m - 1$, we have that

$$E(5, m) \leq E(4, m') + E(4, m'' + 1) \leq \frac{4}{5}m' + \frac{12}{5} + \frac{4}{5}(m'' + 1) + \frac{12}{5} = \frac{4}{5}m + 4$$

Since $E(5, m)$ must be integral, we actually have $E(5, m) \leq \lfloor \frac{4}{5}m + \frac{16}{5} \rfloor$. This integral condition implies that $E(5, m) \leq \frac{14}{15}m + \frac{8}{3}$ for all $m \geq 6$. Observe, if $m = n = 5$, it is impossible to partition the set A into two sets that span disjoint rectangles A', A'' of dimensions $[4] \times [m']$ and $[4] \times [m'']$, respectively. If you could, one vertex in A'' or A' must be obsolete (c.f. argumentation in Figure 3.9).

If $k'' = 4$ and $k''' = 3$, because of $\text{dist}(\boxed{A''}, A''') > 2$, we have $m' + 1 = (m'' + m''' + 1) + 1 = m$. Thus,

$$\begin{aligned} E(5, m) &\leq E(4, m'') + E(3, m''') + 1 \\ &\leq \begin{cases} 4 + 3 + 1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m'' = 2, m''' = 2, \text{ and } m = 6, \\ 4 + 4 + 1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m'' = 2, m''' = 3, \text{ and } m = 7, \\ 4 + 3 + 1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m'' = 3, m''' = 2, \text{ and } m = 7, \\ 5 + 3 + 1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m'' = 4, m''' = 2, \text{ and } m = 8, \\ 5 + 4 + 1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m'' = 4, m''' = 3, \text{ and } m = 9, \end{cases} \end{aligned}$$

and for $m'', m''' \geq 4$, and $m \geq 10$, we have that

$$\begin{aligned} E(5, m) &\leq E(4, m'') + E(3, m''') + 1 \leq \frac{4}{5}m'' + \frac{12}{5} + \lfloor \frac{2}{3}(m''' + 3) \rfloor + 1 \\ &\leq \frac{14}{15}m + \frac{8}{3} + \frac{7}{5} - \frac{2}{15}(m + m'') \leq \frac{14}{15}m + \frac{8}{3}. \end{aligned}$$

If $k'' = 4$ and $k''' = 2$ we may use the same calculation as above together with $E(2, m''') \leq E(3, m''')$.

If $k'' = 4$ and $k''' = 1$ we immediately have that $m''' = 1$ and $m'' + m''' + 1 = m$, and thus

$$E(5, m) \leq E(4, m'') + 1 + 1 \leq \frac{4}{5}m'' + \frac{12}{5} + 2 = \frac{4}{5}(m - 2) + \frac{22}{5} \leq \frac{14}{15}m + \frac{8}{3}.$$

If $k'' = 3$ and $k''' = 3$, again, due to $\text{dist}(\boxed{A''}, A''') > 2$, we have $m' + 1 = m'' +$

$m''' + 1 = m$. Therefore,

$$E(5,m) \leq E(3,m'') + E(3,m''') + 1$$
$$\leq \begin{cases} 3+3+1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m''=2, m'''=2, \text{ and } m=5, \\ 4+3+1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m''=3, m'''=2, \text{ and } m=6, \\ 4+3+1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m''=4, m'''=2, \text{ and } m=7, \\ 4+4+1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m''=4, m'''=3, \text{ and } m=8, \\ 4+4+1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m''=4, m'''=4, \text{ and } m=9, \\ 5+4+1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m''=5, m'''=4, \text{ and } m=10, \\ 5+5+1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m''=5, m'''=5, \text{ and } m=11, \\ 6+5+1 \leq \frac{14}{15}m + \frac{8}{3} & \text{for } m''=6, m'''=5, \text{ and } m=12, \end{cases}$$

and for $m'', m''' \geq 6$, and $m \geq 13$ we have that

$$E(5,m) \leq E(3,m'') + E(3,m''') + 1 \leq \frac{2}{3}m'' + 2 + \frac{2}{3}m''' + 2 + 1 \leq \frac{2}{3}(m-1) + 5$$
$$\leq \frac{14}{15}m + \frac{8}{3}$$

If $k'' = 3$ and $k''' = 2$, we have $m' + 1 = m'' + m''' + 1 = m$ and thus with $E(2,m''') \leq E(3,m''')$ the desired bound follows.

If $k'' = 3$ and $k''' = 1$, $m''' = 1$ is immediate, and we get

$$E(5,m) \leq E(3,m'') + 1 + 1 \leq \frac{2}{3}m'' + 2 + 2 \leq \frac{14}{15}m + \frac{8}{3}.$$

If $k'' = 2$ and $k''' = 2$, we deduce that $m'' + m''' \leq m' + 2 = m + 1$ (c.f. Figure 3.9). Hence, with $m \geq 5$,

$$E(5,m) \leq E(2,m'') + E(2,m''') + 1 \leq \frac{2}{3}(m''+1) + \frac{2}{3}(m'''+1) + 1 \leq \frac{2}{3}(m+3) + 1$$
$$\leq \frac{14}{15}m + \frac{8}{3}.$$

Finally, note that $k'' \in \{1,2\}$ and $k''' = 1$ are not possible. This concludes the proof of Lemma 57. □

3.1.4 Open Problems

We conjecture that the lower bound that we gave in Theorem 49 is tight for square grids, i.e. $\lim_{n\to\infty} \frac{E(n)}{n^2} = \frac{2}{15}$. Even more, we think the following conjecture is true.

Conjecture 58. *Let A be an one-(k,l)-corner-avoiding set for a n by m grid graph with $m,n \geq 2$, then*

$$|A| \leq \frac{2}{15}mn + \frac{2}{5}(m+n) + \frac{2}{3} - \frac{2}{15}k \cdot l.$$

For $\min m,n \leq 5$, the statement of the conjecture is true, following the same argumentation from the proofs of Lemma 55 and Lemma 57. The reason we think this conjecture is true is the following:

We can find a partition of any minimal percolating set into corner avoiding subsets repeatedly applying Lemma 53. Also, one obtains larger corner-avoiding sets from joining smaller corner-avoiding sets. If we "properly" join them pairwise[6], we can prove that we cannot exceed the bound in Conjecture 58. However, an extremely nested case analysis is needed to make the bound work for this decomposition.

This conjecture would imply that $E(n) \leq \frac{2}{15}n^2 + \frac{4}{5}n + \frac{2}{3}$.

[6] similarly to the construction in Lemma 51

3.2 Phase Transition for a Non-Attractive Infection Process

This section is a joint work with Markus Heydenreich, Kilian Matzke, and Cristina Toninelli and has been published in [48]. We consider a non-attractive three-state contact process on $\mathbb{Z}$ and prove that there exists a regime of survival as well as a regime of extinction. In more detail, the process can be regarded as an infection process in a dynamic environment, where uninfected sites are either healthy or passive. Infected sites can recover only if they have a healthy site nearby, whereas uninfected sites may become infected only if there is no healthy and at least one infected site nearby. The transition probabilities are governed by a global parameter q: for large q, the infection dies out, and for small enough q, we observe its survival. The main result is obtained by a coupling to a discrete time Markov chain, using its drift properties in the respective regimes.

3.2.1 Related Work

The classical contact process, as introduced by Harris in 1974 [55], has been a central topic of research in interacting particle systems. It is formally defined as $\{0,1\}^{\mathbb{Z}^d}$-valued spin system, where 1's flip to 0's at rate 1, and flips from 0 to 1 occur at rate λ times the number of neighbors in state 1, where $\lambda > 0$ is a parameter of the model. Commonly, the lattice sites are called 'individuals', which are either *infected* (i.e., in state 1) or *healthy* (i.e., in state 0). Many fundamental questions have been settled for this model, the results are summarized in the monographs by Liggett and Durrett in [29, 66, 67].

Among the most important results are the existence of a phase transition for survival of a single infected particle, the complete convergence theorem, and extinction of the critical contact process. Much more refined results have appeared in recent years. In view of these successes, it may seem surprising that results are considerably sparse as soon as multitype contact processes are considered. Results have only been achieved in very specific situations, examples are the articles by Cox and Schinazi [28], Durrett and Neuhauser [30], Durrett and Swindle [31], Konno et al. [62], Neuhauser [72], and Remenik [76] for various models.

Our focus here is on the contact process with three types, and this carries already severe complications. A fair number of models considered in the literature stems from a biological context (either evolvement of biological species or vegetation models); typical questions that have been considered are coexistence versus extinction and phase transitions. Examples are the work of Broman [21] and Remenik [76].

There are two features that are shared by all of these models: they are monotonic and they are (self-)dual (we refer to [67] for a definition of these terms). These two properties are crucial ingredients in the analysis; if they fail, then most of the known tools fail. This might be illustrated by looking at Model A in [14], which is a certain

3-state contact process. Even though there are positive rates for transitions between the various states of this model and apparent monotonicity, the lack of any usable duality relation prevented all efforts in proving convergence to equilibrium for that model.

For the model considered in the present section, it appears that there is no duality relation that we can exploit and monotonicity is restricted to a very particular situation only. Yet we are able to prove the occurrence of a phase transition by means of coupling to certain discrete-time Markov chains and analyzing drift properties of these chains. We believe that the technique presented here is useful in greater generality. A motivation for studying this process stems from the connection with the out of equilibrium dynamics of kinetically constrained models, as we will explain in detail in Section 3.2.3. We believe that the proof techniques apply in similar situations.

3.2.2 The Model

Our state space is $\Omega = \{0, 1, 2\}^{\mathbb{Z}}$, equipped with the product topology (which makes Ω compact). Further, $q \in [0, 1]$ is a parameter and $(\eta_t)_{t\geq 0}$ is a Markov process on Ω. We say that at time t,

$$\text{site } x \text{ is } \begin{cases} \textit{healthy} & \text{if } \eta_t(x) = 0, \\ \textit{passive} & \text{if } \eta_t(x) = 1 \text{ and} \\ \textit{infected} & \text{if } \eta_t(x) = 2. \end{cases}$$

Informally, we can describe the process as follows. Each site x independently waits an exponential time with intensity 1 and then updates its state according to the following rules:

- If at least one neighboring site is healthy, then x becomes healthy with probability q and passive w.p. $1 - q$.
- If at least one neighbor is infected and none is healthy, then a previously healthy x becomes infected w.p. $1 - q$, a previously passive x becomes infected w.p. q and remains in its state otherwise.

We point out several simple properties of the process. First, as long as a site has only passive neighbors, it is blocked in the sense that it will not update. Second, if there are no infected sites at time t, then the same holds for all $t' > t$. Third, if the infected sites form an interval on $\mathbb{Z}$ at time t, they also form an interval for all $t' > t$; that is, an infected strip will not exhibit holes. And fourth, if there are no healthy sites, then infection spreads unhindered at rate q.

For a more formal description, the process can be characterized by its probability

generator, which is the closure of the operator

$$\mathcal{L}f(\eta) = \sum_{x\in\mathbb{Z}} \Big[c_x(\eta) q (f(\eta^{x,0}) - f(\eta)) + c_x(\eta)(1-q)(f(\eta^{x,1}) - f(\eta)) \\ + \bar{c}_x(\eta) \left[q\, \mathbb{1}_{\{\eta(x)=1\}} + (1-q)\mathbb{1}_{\{\eta(x)=0\}} \right] (f(\eta^{x,2}) - f(\eta)) \Big],$$

$$f \in \Big\{ f : \Omega \to \mathbb{R} \text{ cont.} : \sum_{x\in\mathbb{Z}} \sup\{|f(\eta) - f(\eta')| : \eta, \eta' \in \Omega, \eta(y) = \eta'(y) \text{ for all } y \neq x\} = 0 \Big\}.$$

Here, $c_x(\eta) = \mathbb{1}_{\{\eta(x-1)\cdot\eta(x+1)=0\}}$ and $\bar{c}_x(\eta) = \mathbb{1}_{\{\eta(x-1)\cdot\eta(x+1)\geq 2\}}$. Furthermore, $\eta^{x,i}$ is the configuration where $\eta^{x,i}(y) = \eta(y)$ for all $y \neq x$ and $\eta^{x,i}(x) = i$, $x, y \in \mathbb{Z}, i \in \{0,1,2\}$.

Another way of constructing the process is via a graphical representation analogous to [66, Chapter III, Section 6]. We briefly describe this representation, as we shall rely on it on several occasions at later stages.

We start with a space-time diagram $\mathbb{Z} \times [0,\infty)$, and equip each site $x \in \mathbb{Z}$ with four independent Poisson clocks N_0^x, N_1^x, N_{20}^x and N_{21}^x of rates $q, 1-q, 1-q$ and q respectively. These four clocks are independent of the other sites' clocks. Suppressing the dependence on site x, an event of N_0 at time t is called an event of type H at (x,t) in the space-time diagram. Accordingly, we have events P for N_1, and events I_0 and I_1 for N_{20} and N_{21}.

Given a realization of the set of Poisson clocks and some initial configuration η of $\{0,1,2\}^{\mathbb{Z}}$, we can now determine the state of $\eta_t(x)$: We follow the line $\{(x,t) : t \geq 0\}$, starting at $t = 0$. If an event of type H or P takes place at time t_0, and if $\eta_{t_0}(x+1)\cdot\eta_{t_0}(x-1) = 0$, then x becomes healthy or passive respectively (this is independent of its state before time t_0). If the event at time t_0 is of type I_0 we set x to be infected, provided that both $\eta_{t_0}(x-1)\cdot\eta_{t_0}(x+1) \geq 2$ and $\eta_{t_0^-}(x) = 0$. Similarly, if the event is of type I_1 and both $\eta_{t_0}(x-1)\cdot\eta_{t_0}(x+1) \geq 2$ and $\eta_{t_0^-}(x) = 1$, we set x to be infected.

Note that, depending on the environment of x, at least one of the two is true: Events of types H, P do not change the state of x or events of type I_0, I_1 do not change the state of x. So, if x is neither surrounded by passive neighbors nor infected with no healthy neighbors, it updates at rate 1. Indeed, these are the desired dynamics and the distribution of η_t is the one described informally above.

Without mentioning it explicitly, we shall repeatedly make use of this construction and speak of clock rings of a site x, by which we mean any event of the four Poisson clocks of x. In that sense, we can think of x as possessing only one clock N^x of rate 1. On an event of N^x, if x has a healthy neighbor, we declare the event to be of type H or P with probability q or $1-q$. If x has no healthy but an infected neighbor and is healthy (passive) itself, we declare the event of type I_0 (I_1). In all other cases, nothing

happens on the event.

For an initial configuration $\eta \in \Omega$, we denote by $\mathbb{P}^\eta$ the corresponding probability measure. This superscript will be dropped for the sake of convenience if context permits.

As we wrote earlier, monotonicity is an important tool in the analysis of such processes. One monotonicity property the (η_t) process exhibits is the following.

Claim 3. For arbitrary $\eta \in \Omega$ and $x \in \mathbb{Z}$, we have that

$$\mathbb{P}^{\eta'}\left[\eta_t \notin \{0,1\}^{\mathbb{Z}} \text{ for all } t \geq 0\right] \geq \mathbb{P}^{\eta''}\left[\eta_t \notin \{0,1\}^{\mathbb{Z}} \text{ for all } t \geq 0\right],$$

where $\eta' = \eta^{x,2}$ and $\eta'' \in \{\eta^{x,1}, \eta^{x,0}\}$.

In words, additional infected sites cannot decrease the chance of the infection's survival. However, the same is not necessarily true anymore for $\eta' = \eta^{x,1}$ and $\eta'' = \eta^{x,0}$.

Proof. We couple the two processes with initial configurations η' and η'' via their graphical representation in the sense that both are equipped with the same realization of Poisson clock events. It is then a simple consequence of the definition of the dynamics and corresponding transition rates that, almost surely, $\eta'_t(x) \in \{\eta''_t(x), 2\}$ for all $t \geq 0$ and $x \in \mathbb{Z}$. □

3.2.3 Results and Discussion

Our main result is a phase transition for (η_t) in the parameter q: if q is very close to 0, then any number of initially infected sites survives with positive probability, whereas if q is close to 1, then the infection dies out with probability 1.

Theorem 59. *There exist values $0 < q_0 < q_1 < 1$ such that*

(i) for any initial configuration $\eta \notin \{0,1\}^{\mathbb{Z}}$, we have

$$\mathbb{P}^\eta\left[\eta_t \notin \{0,1\}^{\mathbb{Z}} \text{ for all } t \geq 0\right] > 0 \qquad \text{for all } q \leq q_0,$$

(ii) and for any initial configuration η with $\sup_{x\in\mathbb{Z}} \inf_{y\in\mathbb{Z}}\{|x-y| : \eta(y) = 0\} < \infty$, we have

$$\mathbb{P}^\eta\left[\eta_t \in \{0,1\}^{\mathbb{Z}}\right] \xrightarrow{t\to\infty} 1 \qquad \text{for all } q \geq q_1.$$

We thus prove the existence of different regimes without relying on duality properties. Since there is no monotonicity that can be exploited here, we cannot rule out that there are more than one transitions between the regimes "the infection dies out" and "the infection survives".

Note that the case $q = 0$ is degenerate and of little interest, as it admits traps: If there is a site $x \in \mathbb{Z}$ and a time $t \geq 0$ such that we exhibit $(\eta_t(x), \eta_t(x+1), \eta_t(x+2)) = (1, 0, 1)$, then this triple will remain fixed for all $t' \geq t$.

A very related process to the one just introduced is the simpler version for which, informally, the second condition is altered to: "If at least one neighbor of x is infected and none is healthy, then x becomes infected." It is clear that the set of infected sites in this version dominates our process. However, the same proof techniques used below yield similar results to Theorem 59 (namely, also a phase transition).

Connections to Kinetically Constrained Models.

This model has an indirect connection with Fredrickson-Andersen 1 spin facilitated model (FA1f) [16, 40, 41]. In this case, the configuration space is $\{0, 1\}^{\mathbb{Z}}$ and the dynamics are defined as follows: a site x with occupation variable 0 flips to 1 at rate $1 - q$ if and only if at least one among its nearest neighbors is in state zero; a site x with occupation variable 1 flips to 0 at rate q if and only if at least one among its nearest neighbors is in state zero. Note that the constraint for the $0 \to 1$ and the $1 \to 0$ updates are the same and the dynamics satisfies detailed balance w.r.t. the product measure μ with $\mu(\eta(x) = 0) = q$. Note also that the dynamics of our contact process coincide with the FA1f dynamics if we start from a configuration which does not contain infected sites.

A non-trivial problem for FA1f dynamics is to determine convergence to the equilibrium measure μ for some reasonable initial measure, e.g. an initial product measure with density of healthy sites different from q [16]. We will now explain how our results provide an alternative approach to prove convergence to equilibrium in a restricted density regime. A possible strategy to prove convergence to equilibrium for FA1f dynamics started from an initial configuration η_0 is to couple it with some $\tilde{\eta}_0$ distributed according to μ. This gives rise to a process with 4 states $\{0, 1, 2^{\downarrow}, 2^{\uparrow}\}$. Here, 0 represent sites where both configurations are 0; 1 sites where both configurations are 1; $2^{\downarrow}$ sites where η is 0 and $\tilde{\eta}$ is 1; and $2^{\uparrow}$ sites where η is 1 and $\tilde{\eta}$ is 0. If we now denote the union of sites in state $2^{\downarrow}$ and $2^{\uparrow}$ as "infected sites", then if infection dies out, the original process started in η_0 is distributed with the equilibrium measure (since there are no more discrepancies with the process evolved from $\tilde{\eta}$ which is at equilibrium at any time). It is not difficult to verify that the dynamics of the 4 state contact process induced by the standard coupling among two configurations evolving with FA1f dynamics are such that the union of sites in state $2^{\downarrow}$ and $2^{\uparrow}$ is dominated by the infected sites of our 3-state contact process. Thus, when infection dies out for our process it also dies out for the 4-state contact process and from our Theorem 59 (ii) we get convergence to equilibrium for $q \geq q_1$ for the FA1f dynamics. This result was already proven by a completely different technique in Blondel et al. [16] for parameter $q > 1/2$. Notice that convergence to equilibrium is expected to hold for FA1f dynamics

at all $q > 0$ starting from η satisfying the hypothesis of our Theorem 59 (ii), namely infection should always disappear in the 4 state contact process. This is certainly not the case for our 3 state contact process which has a survival extinction transition, as proved by Theorem 59 (i).

3.2.4 The Small q Regime

In this subsection, we prove assertion (i) of Theorem 59. First, We define

$$\Omega^* = \{\eta \in \Omega : \exists\, a \leq b \in \mathbb{Z} \text{ s.t. } \{x : \eta(x) = 2\} = [a, b] \cap \mathbb{Z}\},$$

the set of configurations where infected sites form a finite, nonempty interval.

Proposition 60. *Consider some $\eta \in \Omega^*$. Then there exists $0 < q_0$ such that*

$$\mathbb{P}^\eta \left[\eta_t \notin \{0,1\}^{\mathbb{Z}} \text{ for all } t \geq 0\right] > 0 \qquad \text{for all } q \leq q_0,$$

For the proof, we observe first that the set of sites in state 2, which we call the *infected cluster*, is always connected. We would like to focus on the behavior of the infected boundary sites and so, due to symmetry, on

$$\mathcal{I}(t) := \sup\{x \in \mathbb{Z} : \eta_t(x) = 2\},$$

the position of the rightmost infected site. If there is only one infected site (thus, leftmost and rightmost infected site coincide), both with positive probability the next change in number of infected sites might result in zero (extinction of the infection) or two infected sites. If the number of infected sites is at least two, only the status on the sites to the right of the rightmost infected site have direct influence on the 'movement' of $\mathcal{I}(t)$.

In (an informal) summary, if the infection shrinks to size one, it recovers with positive probability to size at least two. If we show that from there, infection spreads with positive probability, we obtain our result. Therefore, we focus on this latter regime in the following.

With this in mind, we now introduce a Markov chain, which can be interpreted as a simplified model of the rightmost infected site and its local right neighborhood, and prove a drift property for it. This shall turn out to be useful when coupling this auxiliary Markov chain to our original process in Section 3.2.4.

An Auxiliary Markov Chain

We define a (discrete time) Markov chain $(Y_i)_{i\geq 0}$ living in the (countable) state space $\mathcal{S} = \mathbb{Z} \times \{0,1\}^3$. We denote its first coordinate as the chain's *level* or *state* and thus

can partition $\mathcal{S}$ into its n-states

$$\mathcal{S}_n := \{(\omega_1, \omega_2, \omega_3, \omega_4) \in \mathcal{S} : \omega_1 = n\}$$

for $n \in \mathbb{Z}$. The Markov chain is defined by its transition graph shown in Figure 3.10. The subgraphs induced by $\mathcal{S}_n$ are isomorphic, and furthermore, two states from $\mathcal{S}_n$ and $\mathcal{S}_m$ for $|m - n| \geq 2$ have transition probability zero. Hence, for simplicity, we can restrict ourselves to depicting the transition graph induced by $\mathcal{S}_n$, with additional states in $\mathcal{S}_{n\pm1}$ along with their respective transition probabilities. We denote the probability measure of this Markov chain by $\mathbb{P} = \mathbb{P}_q$ (we trust that this causes no confusion with the measure of the interacting particle system).

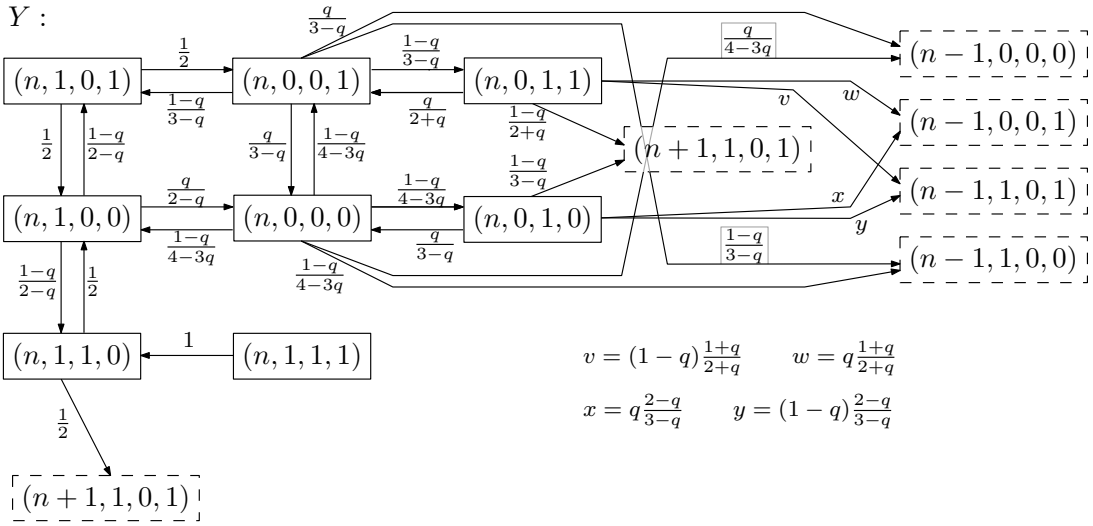


Figure 3.10: Transition subgraph of Y induced by $\mathcal{S}_n$ and its neighboring states.

We define the stopping time τ to be the first time the Markov chain changes its level:

$$\tau := \min\{i \in \mathbb{N} : \exists n \in \mathbb{Z} : Y_0 \in \mathcal{S}_n, Y_i \in \mathcal{S}_{n\pm1}\}.$$

Using Y_i^m (for $1 \leq m \leq 4$) to access the mth component of the state which Y is in at time i, say that Y_i is a progressive step (progress) if $Y_i^1 = Y_{i-1}^1 + 1$ and similarly call Y_i a regressive step (regress) if $Y_i^1 = Y_{i-1}^1 - 1$. We say that a natural number i is a step time (step) if Y_i is either a progressive or a regressive step. The desired drift property is stated in

Lemma 61. *There exists $0 < q_0 < 1$ such that for all $0 < q < q_0$, we have*

$$\mathbb{E}\left[Y_\tau^1 - Y_0^1\right] > 0.$$

Let us start with the following observation, which is an immediate consequence of the definition of the Markov chain dynamics.

Observation 62. *Let $Y_0 \in \mathcal{S}_n$. If $Y_\tau \in \mathcal{S}_{n+1}$, then necessarily $Y_\tau = (n+1, 1, 0, 1)$. On the other hand, conditional on $Y_\tau \in \mathcal{S}_{n-1}$, we have $Y_\tau \in \{(n-1, 0, 0, 1), (n-1, 0, 0, 0)\}$ with probability q and $Y_\tau \in \{(n-1, 1, 0, 1), (n-1, 1, 0, 0)\}$ with probability $1 - q$.*

We call $\mathcal{G}_n := \{(n, 1, 0, 1), (n, 1, 0, 0)\}$ the two '*good*' n-states. After every step, Y ends up in one of the two good states with probability tending to 1 for $q \to 0$. It is therefore of special interest to study the behavior of Y started in either of the two good states. We do so via

Lemma 63. *Let $n \in \mathbb{Z}$ and $\varepsilon > 0$. Then there exists $0 < q_0 < 1$ such that*

$$\begin{aligned}
\frac{1}{2} - \varepsilon &< \mathbb{P}[Y_\tau \in \mathcal{S}_{n+1} \mid Y_0 = (n, 1, 0, 1)] < \frac{4}{7} + \varepsilon, \\
\frac{2}{3} - \varepsilon &< \mathbb{P}[Y_\tau \in \mathcal{S}_{n+1} \mid Y_0 = (n, 1, 0, 0)]
\end{aligned}$$

for all $q < q_0$.

Proof. The proof proceeds by counting paths in the transition graph. We define

$$\begin{aligned}
\theta_1 &:= \mathbb{P}[Y_\tau \in \mathcal{S}_{n+1} \mid Y_0 = (n, 1, 1, 0)], \\
\theta_2 &:= \mathbb{P}[Y_\tau \in \mathcal{S}_{n+1} \mid Y_0 = (n, 1, 0, 0)], \\
\theta_3 &:= \mathbb{P}[Y_\tau \in \mathcal{S}_{n+1} \mid Y_0 = (n, 1, 0, 1)].
\end{aligned}$$

We also set $a := \frac{1-q}{2(3-q)}$ to be the *weight* of the 2-cycle between states $(n, 1, 0, 1)$ and $(n, 0, 0, 1)$. The weight of a cycle is the probability that the Markov chain transitions along this cycle in the transition graph. As a path may use this cycle arbitrarily often, we have

$$\begin{aligned}
\theta_1 &= \frac{1}{2}\left(1 + \theta_2\right), \\
\theta_2 &\geq \frac{1-q}{2-q}\theta_1 + \frac{1-q}{2-q}\theta_3, \\
\theta_3 &\geq \left(\frac{1}{2}\theta_2 + \frac{1-q}{2+q}a\right)\sum_{k\geq 0} a^k = \frac{1}{1-a}\left(\frac{1}{2}\theta_2 + \frac{1-q}{2+q}a\right).
\end{aligned}$$

We briefly highlight which paths constitute the lower bound on θ_3: First, walk along the 2-cycle between $(n, 1, 0, 1)$ and $(n, 0, 0, 1)$ an arbitrary number of times and then

either move to $(n,1,0,0)$ (which then gets us to θ_2 by the Markov property) or move straight to $\mathcal{S}_{n+1}$ via $(n,0,1,1)$. This leads to the explicit lower bounds

$$\begin{aligned}\theta_1 &\geq \frac{15-2q-q^3}{18+9q-2q^2-q^3},\\ \theta_2 &\geq \frac{12-13q+2q^2-q^3}{18+9q-2q^2-q^3},\\ \theta_3 &\geq \frac{3-3q}{6+5q+q^2}.\end{aligned}$$

For small q, all of these values are strictly larger than $\frac{1}{2}$, except for θ_3, where we have $\theta_3 \nearrow \frac{1}{2}$ as $q \to 0$. Finally set $b := \frac{1-q}{2(2-q)}$ to be the weight of a 2-cycle between states $(n,1,0,1)$ and $(n,1,0,0)$ and observe that, by counting paths ending in $\mathcal{S}_{n-1}$, we have

$$1-\theta_3 \geq \left(\frac{1}{2(3-q)} + \frac{(1-q)(1+q)}{2(3-q)(2+q)}\right)\left(\sum_{n\geq 0}\sum_{k=0}^{n}\binom{n}{k}a^k b^{n-k}\right) = \frac{6-q-3q^2+q^3}{14+q-3q^2}.$$

In the first parenthesis, the first term comes from paths ending in $(n-1,0,0,0)$ and $(n-1,1,0,0)$, whereas the second terms comes from paths ending in $(n-1,0,0,1)$ as well as $(n-1,1,0,1)$. The lemma follows for q sufficiently small.

□

Proof of Lemma 61. We start by defining

$$\tau_2 := \min\{i > \tau : \exists n \in \mathbb{Z} : Y_\tau \in \mathcal{S}_n, Y_i \in \mathcal{S}_{n\pm 1}\}$$

to be time of the first level change after τ and actually prove $\mathbb{E}\left[Y^1_{\tau_2} - Y^1_0\right] > 0$. Noting that after two level changes, Y^1 will either have increased or decreased by 2 or not changed at all, the lemma follows from proving

$$\mathbb{P}\left[Y_{\tau_2} \in \mathcal{S}_{n+2} \mid Y_0 \in \mathcal{S}_n\right] > \mathbb{P}\left[Y_{\tau_2} \in \mathcal{S}_{n-2} \mid Y_0 \in \mathcal{S}_n\right]. \tag{3.2}$$

for any integer n. Recalling Observation 62, we can restrict ourselves to proving (3.2) for Y_0 in one of the two good n-states $\mathcal{G}_n = \{(n,1,0,1),(n,1,0,0)\}$, as we are allowed to choose q_0 sufficiently small. Combining Observation 62 with Lemma 63, we have

$$\tilde{\alpha} := \mathbb{P}\left[Y_{\tau_2} \in \mathcal{S}_{n+2} \mid Y_0 \in \mathcal{G}_n\right] \geq \min_{\omega \in \mathcal{G}_n}\left(\mathbb{P}\left[Y_\tau \in \mathcal{G}_{n+1} \mid Y_0 = \omega\right]\right)^2 > \left(\frac{1}{2}-\varepsilon\right)^2$$

for some $\varepsilon > 0$ and q appropriately small. Recalling that $a := \frac{1-q}{2(3-q)}$ was the weight of a 2-cycle between states $(n,1,0,1)$ and $(n,0,0,1)$ and $b := \frac{1-q}{2(2-q)}$ the weight of a 2-cycle

between states $(n,1,0,1)$ and $(n,1,0,0)$, and setting $\omega = (n,1,0,1)$, $\omega' = (n,0,0,1)$ as well as $\omega'' = (n-1,1,0,0)$, we have

$$\begin{aligned}\kappa &:= \mathbb{P}\left[Y_\tau \in \mathcal{G}_{n-1}, Y_{\tau-1} = \omega' \mid Y_0 = \omega\right] \\ &\geq \frac{1-q}{2(3-q)}\left(\sum_{m\geq 0}\sum_{k=0}^{m}\binom{m}{k}a^k b^{m-k}\right) = \frac{1-q}{2(3-q)}\cdot\frac{1}{1-a-b} \\ &= \frac{2-3q+q^2}{7-3q},\end{aligned}$$

with the bound obtained simply by counting paths from ω to ω'' which pass through ω' in their second to last step. With ε small enough ($\varepsilon < 1/100$ say) and Observation 62, we are now able to bound $\alpha := \mathbb{P}\left[Y_{\tau_2} \in \mathcal{S}_{n-2} \mid Y_0 = \omega\right]$, the probability of double regress from ω, as follows:

$$\begin{aligned}\alpha &= \mathbb{P}\left[\{Y_{\tau_2} \in \mathcal{S}_{n-2}\} \cap \{Y_\tau \in \mathcal{S}_{n-1}\} \mid Y_0 = \omega\right] \\ &\leq q + (1-q)\cdot\mathbb{P}\left[\{Y_{\tau_2} \in \mathcal{S}_{n-2}\} \cap \{Y_\tau \in \mathcal{G}_{n-1}\} \cap \{Y_{\tau-1} = \omega'\} \mid Y_0 = \omega\right] \\ &\quad + q + (1-q)\cdot\mathbb{P}\left[\{Y_{\tau_2} \in \mathcal{S}_{n-2}\} \cap \{Y_\tau \in \mathcal{G}_{n-1}\} \cap \{Y_{\tau-1} \neq \omega'\} \mid Y_0 = \omega\right].\end{aligned}$$

Rearranging by defining $B = \{Y_\tau \in \mathcal{G}_{n-1}\} \cap \{Y_{\tau-1} = \omega'\}$ and $B' = \{Y_\tau \in \mathcal{G}_{n-1}\} \cap \{Y_{\tau-1} \neq \omega'\}$ and observing that the event B implies that $Y_\tau = \omega''$, we continue to find that

$$\begin{aligned}\alpha &\leq 2q + (1-q)\sum_{A\in\{B,B'\}}\mathbb{P}\left[Y_{\tau_2} \in \mathcal{S}_{n-2} \mid A\right]\cdot\mathbb{P}\left[A \mid Y_0 = \omega\right] \\ &\leq 2q + (1-q)\cdot\mathbb{P}\left[Y_{\tau_2} \in \mathcal{S}_{n-2} \mid Y_\tau = \omega''\right]\cdot\kappa \\ &\quad + (1-q)\cdot\mathbb{P}\left[Y_{\tau_2} \in \mathcal{S}_{n-2} \mid Y_\tau = \omega\right]\cdot\left(\mathbb{P}\left[Y_\tau \in \mathcal{G}_{n-1} \mid Y_0 = \omega\right] - \kappa\right) \\ &\leq 2q + (1-q)\left(\kappa(1-\theta_2) + (1-q)(1-\theta_3)(1-\theta_3-\kappa)\right) \\ &< 2q + (1-\theta_3)^2 + \kappa(\theta_3-\theta_2) \\ &< 2q + \left(\frac{1}{2}+\varepsilon\right)^2 + \left(\frac{2}{7}-\varepsilon\right)\left(\frac{4}{7}-\frac{2}{3}\right) \\ &< 2q + \tilde{\alpha} - \frac{4}{21}\left(\frac{1}{7} - 11\varepsilon\right) < \tilde{\alpha}\end{aligned}$$

for our chosen ε and q sufficiently small, where θ_i have been defined in the proof of Lemma 63. Note that again we make heavy use of the strong Markov property as well as the bounds from Lemma 63. □

The Coupling

We are now ready to return to our process. Recall that Y should be thought of as a model of the right neighborhood of the rightmost infected site in the original process. Intuitively speaking, we want to find a coupling such that $Y^0 \leq \mathcal{I}(t)$ at any given time—this, however, is ill-defined. To make it more precise, let us first formally build towards the discrete version of the segment of the process that is of interest (i.e., the right neighborhood of the rightmost infected site). For $(\eta_t)_{t\geq 0}$ a realization of the process in Ω^*, we define the map $\Phi : \Omega^* \to \mathbb{Z} \times \{0,1\}^4$ as

$$\Phi(\eta_t) = \Big(\mathcal{I}(t), (\eta_t(\mathcal{I}(t)+i))_{1\leq i\leq 4}\Big).$$

Hence, $(\Phi_t)_{t\geq 0} = (\Phi(\eta_t))_{t\geq 0}$ is the segment of the process we are interested in. Let $(s_i(x))_{i\in\mathbb{N}}$ be the sequence of clock rings for site x. That is, $s_1 \sim \text{Exp}(1)$ and $(s_{i+1}(x) - s_i(x)) \sim \text{Exp}(1)$ for all $i \in \mathbb{N}$. This allows us to define $(R_i)_{i\in\mathbb{N}_0}$, the sequence of times of clock rings of the process restricted to $(\eta_t(\mathcal{I}(t)+i))_{0\leq i\leq 4}$, as $R_0 = 0$ and

$$R_{i+1} = \inf\{s_j(x) : s_j(x) > R_i, x \in \{\mathcal{I}(R_i) + l : 0 \leq l \leq 4\}, j \in \mathbb{N}\}$$

for all $i \geq 0$. We are interested in the process $(X_i)_{i\in\mathbb{N}_0}$, a subset of $(\overline{X})_{i\in\mathbb{N}_0}$, where $\overline{X}_0 = X_0 = \Phi(\eta_{R_0})$ and

$$\begin{aligned} \overline{X}_i &= \Phi(\eta_{R_i}), \\ X_i &= \Phi(\eta_{R_l}), \quad l = \inf\{k \geq i : \Phi(\eta_{R_k}) \neq \Phi(\eta_{R_{i-1}})\} \end{aligned}$$

for all $i \geq 1$. In words, $(\overline{X})_i$ is the embedded discrete time chain of $(\Phi)_t$, and X is the chain obtained from $\overline{X}$ by removing all of the self-loops. Process X is the one which, in certain time windows, behaves very much like Y. To make this precise, we define $(R'_i)_{i\in\mathbb{N}_0}$ with $R'_0 = 0$ and

$$R'_{i+1} = \inf\Big\{\big\{t > R'_i : \mathcal{I}(R'_i) \neq \mathcal{I}(t)\big\} \cup \{s_j(\mathcal{I}(R'_i)+4) : s_j(\mathcal{I}(R'_i)+4) > R'_i, j \in \mathbb{N}\}\Big\}$$

to be the times when either the position of the rightmost infected particle changes or the clock at the site determining the boundary condition, $\mathcal{I}(\,\cdot\,)+4$, rings. We call $W_i := [R'_i, R'_{i+1})$ the *stable windows* for all $i \geq 0$. A stable window closes whenever the boundary site rings or the infected site moves. We can now proceed to describe the behavior of X in a stable window. As we did for Y, we can partition the state space of X into its levels and, as no confusion arises this way, call them $\mathcal{S}_n$ as well. The dynamics within W_i depend only on $\Phi(\eta_{R'_i})$, namely the initial state also encoding the boundary conditions, and are therefore Markovian. Given this initial state for

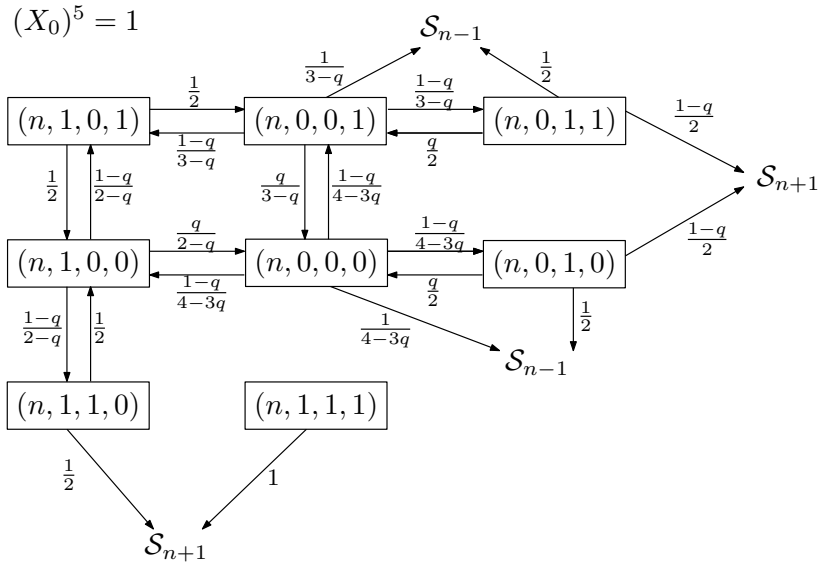


Figure 3.11: Transition subgraph of X in window W_0 induced by $\mathcal{S}_n$ for passive initial boundary conditions. The last component $X^5 = 1$ was dropped in the display of the states, as it holds the constant value 1 throughout the time window W_0. Note that entering $\mathcal{S}_{n\pm1}$ closes W_0.

W_0, we can depict the transition graph in a very similar way as the one for Y, as the subgraphs induced by the levels are again isomorphic, the states in neighboring levels are terminal as they 'close' the window W_0. Conditional on the boundary conditions, the two transition graphs are shown in Figure 3.11 and Figure 3.12.

Proof of Proposition 60. The informal proof outline is the following: We glue together windows until all such combined windows end in level changes. We then use a coupling to retain the drift property of Y within any such window via domination, while making sure to uphold this domination when progressing into the next window. As there is an obvious bijection between states of X and Y we speak of a synchronized coupling (within a certain time frame) if both Markov chains draw from the same source of randomness so that the states of X and Y are the same. It is clear that we cannot maintain such a synchronized coupling, as the transition probabilities are not exactly the same for either boundary condition. The strategy will be to uphold a synchronized coupling and, as soon as it fails, guarantee the right domination until the coupling re-synchronizes.

We now give a rigorous coupling of X and Y which is maintained until X changes its level. We therefore have to describe how X and Y are coupled within a window and how this coupling is carried over into a new window, provided that the closed windows

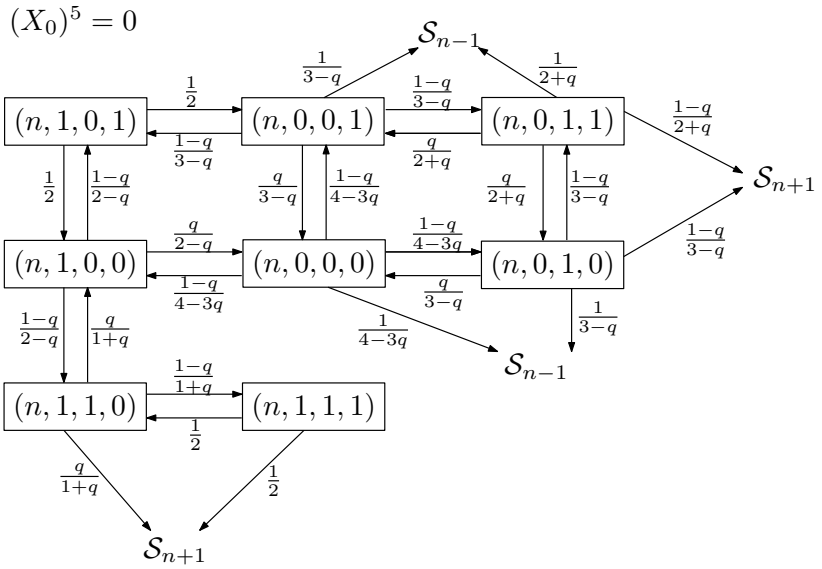


Figure 3.12: Transition subgraph of X in window W_0 induced by $\mathcal{S}_n$ for healthy initial boundary conditions, analogous to Figure 3.11. In contrast, the last component is of constant value 0.

did not result in a level change. For simplicity, call $\bar{X}$ the Markov chain dynamics of X under boundary conditions $(X_0)^5 = 1$ and $\mathring{X}$ under boundary conditions $(X_0)^5 = 0$. Assume $X \in \{\bar{X}, \mathring{X}\}$ starts in state

$$\omega \in \mathcal{H} = \{(n,1,0,1), (n,0,0,1), (n,1,0,0), (n,0,0,0)\}.$$

The transition probabilities from any state in $\mathcal{H}$ are identical in $\bar{X}, \mathring{X}$ and Y, so we can start Y in ω and maintain a synchronized coupling. On the side, we keep a sequence $(Z_j)_{j\in\mathbb{N}}$ of i.i.d. random variables which are uniform on $[0,1]$ and independent of everything else, to determine what to do when we leave the n-states $\mathcal{H}$.

If X finds itself in $\omega \in \{(n,0,1,1), (n,0,1,0)\}$ in step j, we evaluate Z_j: Each neighboring state ω' of ω in X (the set of neighbors is different for $\bar{X}$ and $\mathring{X}$) is assigned an interval $J_X^{\omega'}$ in $[0,1]$ of length the respective transition probability so that no intervals intersect. The same is done for Y so that $J_Y^{\omega'} \subseteq J_X^{\omega'}$ for any $\omega' \in \mathcal{S}_n \cup \mathcal{S}_{n+1}$. This is possible as all the respective edges in Y are of less or equal weight than they are in both $\bar{X}$ and $\mathring{X}$.

If $Z_j \in J_Y^{\omega'} \subseteq J_X^{\omega'}$, both Markov chains transition to ω' and the synchronized coupling is maintained. Otherwise, we must have $Z_j \in J_Y^{\tilde{\omega}}$ for some $\tilde{\omega} \in \mathcal{S}_{n-1}$. In this case, we pause Y and let X run until it changes its level.

Finally, assume X finds itself in $\omega \in \{(n,1,1,0), (n,1,1,1)\}$. The case $\omega = (n,1,1,1)$

is clear: In $\bar{X}$, the next step is to $\mathcal{S}_{n+1}$, closing the window. Similarly, in $\mathring{X}$, either X progresses and closes the windows or X remains synchronized and transitions to $(n, 1, 1, 0)$, which is the only option for Y.

We are left to treat $\omega = (n, 1, 1, 0)$. Clearly, there is a synchronized coupling in $\bar{X}$. On the other hand, if $X = \mathring{X}$, we evaluate Z_j again, but this time we choose the intervals such that $J_X^{\omega'} \subseteq J_Y^{\omega'}$ for all neighboring states but $(n, 1, 1, 1)$. This way, we either maintain a synchronized coupling, or $Z_j \in J_X^{\omega'}$ for $\omega' = (n, 1, 1, 1)$. In this case, we pause Y and let X run to ω' and according to the Z_j until either X transitions from ω' to $\mathcal{S}_{n+1}$ or until X is back in ω in step l and $Z_l \in J_X^{\omega''} \subseteq J_Y^{\omega''}$. We then restart Y, now synchronized with X again. Note that conditional on the last event Y goes to both possible states with probability $1/2$, which agrees with its dynamics.

It is not hard to verify that the dynamics along which both $X \in \{\bar{X}, \mathring{X}\}$ and Y run is according to its given transition probabilities. If the coupling stops being synchronized, then either Y stranded in regress or Y pauses and X is in $(n, 1, 1, 1)$. In either case, the closing of a window is not a problem: If Y is already in $\mathcal{S}_{n-1}$ and X continued to run, the domination is guaranteed. if Y is paused, then X has no option but to either progress or re-synchronize with Y. Hence, $Y^1 \leq X^1$ and the drift carries over to X.

This allows us to maintain the desired domination throughout stable windows until a closing window changes the level of X. We now repeat an argument already made for Y: If X regresses, it transitions to some state $(n-1, 1, x, y)$ with probability tending to 1 as $q \to 0$. We can thus restrict our attention to windows where X starts from one of these states. Upon entering a new window via level change, we re-synchronize the coupling by restarting Y from the same state as X. As Y has a positive drift when started from one of these states, so does X by domination.

If X progresses, it enters the new level in some state $(n+1, 1, x, y)$ by definition of the dynamics. Again, we restart Y in the state that yields the synchronized coupling. From these four states, Y has a positive drift, which proves the theorem. □

Proof of Theorem 59 (i). The assertion is a direct consequence of Proposition 60 via Claim 3. □

3.2.5 Extinction for Large q

We import some notation from Section 3.2.4. Namely, let $(X_i)_{i\in\mathbb{N}} \subset \mathbb{Z} \times \{0, 1\}^4$ be the discrete time process describing the rightmost infected site and its neighbors and let $\mathcal{S}_n$ denote all n-levels of its state space. Similar to how τ and τ_2 were defined for the Markov chain Y in that subsection, we define τ_i for $i \geq 0$ as

$$\tau_{i+1} = \inf\{j \geq \tau_i : \exists n \in \mathbb{Z} : X_{\tau_i} \in \mathcal{S}_n, X_j \in \mathcal{S}_{n\pm1}\},$$

where we set $\tau_0 = 0$, to be the sequence of level changes of X. We abbreviate $\tau = \tau_1$ when it is convenient. We next define two stopping times describing the length of consecutive progressive and regressive steps, respectively. That is, we set

$$\overrightarrow{\tau} := \sup\{i \in \mathbb{N}_0 : \exists n \in \mathbb{Z} : X_0 \in \mathcal{S}_n, X_{\tau_i} \in \mathcal{S}_{n+i}\},$$
$$\overleftarrow{\tau} := \sup\{i \in \mathbb{N}_0 : \exists n \in \mathbb{Z} : X_0 \in \mathcal{S}_n, X_{\tau_i} \in \mathcal{S}_{n-i}\}$$

and call $\overrightarrow{\tau}$ a progressive and $\overleftarrow{\tau}$ a regressive interval, respectively. It is clear that we can partition X into alternating progressive and regressive intervals. Our aim is to prove that the length of a progressive interval is, in expectation, less than the length of a regressive one. Note that if τ is a regressive step, then $X_\tau \in \mathcal{G}_n$ for some integer n, where

$$\mathcal{G}_n = \{(n, z_2, 0, z_4, z_5) : z_i \in \{0, 1\} \text{ for } i = 2, 4, 5\}.$$

Similarly, if τ is a progressive step, then $X_\tau \in \mathcal{B}_n$ for some n, with

$$\mathcal{B}_n = \{(n, 1, z_3, z_4, z_5) : z_i \in \{0, 1\} \text{ for } i = 3, 4, 5\}.$$

The following lemma is the main step in the proof of Theorem 59 (ii).

Lemma 64. *In the above notation, we have that*

$$\mathbb{E}[\overrightarrow{\tau} \mid X_0 \in \mathcal{G}_n] < \mathbb{E}[\overleftarrow{\tau} \mid X_0 \in \mathcal{B}_n]$$

for $n \in \mathbb{Z}$ and q sufficiently large.

Proof of Theorem 59 (ii). As observed above, starting at τ, any progressive interval must start from a $\mathcal{G}$ state, whereas any regressive interval must start from a $\mathcal{B}$ state. Hence, the conditioning in Lemma 64 is not a restriction and the rightmost infected site is dominated by a $\mathbb{Z}$-valued random walk with negative drift, which yields the claimed result.

□

Turning towards the proof of Lemma 64, a key observation is the fact that, when q is sufficiently large, healthy sites drift towards each other. More precisely, given a connected set of passive sites with healthy boundary conditions, we expect the size of this set to decrease with time. With this in mind we define

$$\xi^x(\eta) = \inf\{|y - x| : y \in \mathbb{Z}, \eta(y) = 0\},$$

for some $\eta \in \Omega$ and $x \in \mathbb{Z}$, i.e. the distance of x to the next healthy site in η.

Lemma 65. *Let $q > 1/2$ and $\nu \in \{0,1\}^{\mathbb{Z}}$. Assume that $\kappa := \xi^x(\nu) < \infty$ for $x \in \mathbb{Z}$. Then for the process (η_t) with $\eta_0 = \nu$, we have*

$$\mathbb{E}^\nu\left[\xi^x(\eta_t)\right] \leq \max\{1, \kappa + t(1-2q)\} \quad \forall t \geq 0.$$

Proof. Let $\eta = \eta_0$ be state of the process at time 0. Due to translation invariance and symmetry, we shall consider site $x = 0$ and assume the closest healthy site is located at $\xi_t = \xi^0(\eta_t) \gg 1$ for all times t. Since we are only interested in an upper bound, we always assume that $\eta(\xi_t + 1) = 0$. In doing this, we obtain a process whose ξ_t-value dominates the original one. We thus end up with the following simplification:

- If site ξ_t updates, then with probability $1-q$, it becomes passive and $\xi_{t^+} = \xi_t + 1$,
- if site $\xi_t - 1$ updates, then with probability q, it becomes healthy and $\xi_{t^+} = \xi_t - 1$,

and those are the only updates changing the position of ξ_t. Hence, the expected change of η_t after an update is $1-2q < 0$. The number of updates in $[0,t]$ of these two sites is 2 Poisson(t)-distributed, and with probability $1/2$, an update yields a change of position, so N_t, the number of position changes in $[0,t]$, is Poisson(t)-distributed. Hence, as all of this remains true for $\xi_t \geq 1$, the statement follows by Wald's lemma. □

Note that Lemma 65 is very much in the spirit of Proposition 4.1 in [16], even though we need a much weaker statement to prove Lemma 64, namely that $E^\nu\left[\xi^x(\eta_t)\right]$ is not increasing.

Proof of Lemma 64. We begin by considering $\overleftarrow{\tau}$ and noting that, no matter the boundary conditions,

$$\begin{aligned}&\mathbb{P}[X_{\tau_2} \in \mathcal{S}_{n-1} \mid X_{\tau_1} \in \mathcal{S}_n, X_0 \in \mathcal{S}_{n+1}] = \mathbb{P}[X_\tau \in \mathcal{S}_{n-1} \mid X_0 \in \mathcal{G}_n] \geq \alpha(q)\\ &= \min\left\{\frac{q}{2-q} \cdot \frac{1}{4-3q}, \frac{1}{3-q}\left(1 + \frac{q}{4-3q}\right)\right\} \xrightarrow{q\to 1} 1.\end{aligned}$$

In words, following a regressive step, we witness another regressive step with probability at least $\alpha \to 1$. That is because from $\mathcal{G}_n$, X ends up in another regressive step within three steps or less, regardless of a change of boundary conditions during that time. As a direct consequence, $\mathbb{E}[\overleftarrow{\tau} \mid X_0 \in \mathcal{G}_n] \geq (1-\alpha(q))^{-1}$ gets arbitrarily large for $q \to 1$. On the other hand, we have that there exists $\beta < 1$ such that

$$\mathbb{P}[X_\tau \in \mathcal{S}_{n+1} \mid X_0 \in \mathcal{B}_n \backslash \{(n,1,1,1,1)\}] \leq \beta$$

for all q not too small ($q > 1/2$ say), and thus

$$\begin{aligned}\mathbb{E}\left[\overrightarrow{\tau} \mid X_0 \in \mathcal{B}_n \backslash \{(n,1,1,1,1)\}\right] &\leq \beta\left(1 + \mathbb{E}\left[\overrightarrow{\tau} \mid X_0 \in \mathcal{B}_n\right]\right) \\ &\leq \beta\left(1 + \mathbb{E}\left[\overrightarrow{\tau} \mid X_0 = (n,1,1,1,1)\right]\right).\end{aligned}$$

So if we can bound the last quantity by some constant, we are done. This is where Lemma 65 comes in. We bound this expectation by "jumping" to the closest healthy site, infecting all passive sites on the way. More precisely, we progress the infection by force until reaching a state in $\mathcal{G}_n$.

$$\begin{aligned}&\mathbb{E}\left[\overrightarrow{\tau} \mid X_0 = (n,1,1,1,1)\right] \\ &\leq \sum_{i=0}^{\infty}\left(\mathbb{E}\left[\overrightarrow{\tau} \mid X_0 \in \mathcal{G}_n, \xi^{\mathcal{I}(0)}(\eta_0) = i+2\right] + i\right)\mathbb{P}\left[\xi^{\mathcal{I}(0)}(\eta_0) = i+2\right] \\ &\leq \sum_{i=0}^{\infty} i\mathbb{P}\left[\xi^{\mathcal{I}(0)}(\eta_0) = i\right] + \mathbb{E}\left[\overrightarrow{\tau} \mid X_0 \in \mathcal{G}_n\right]\sum_{i=0}^{\infty}\mathbb{P}\left[\xi^{\mathcal{I}(0)}(\eta_0) = i+2\right] \\ &\leq \mathbb{E}\left[\xi^{\mathcal{I}(0)}(\eta_0)\right] + \mathbb{E}\left[\overrightarrow{\tau} \mid X_0 \in \mathcal{G}_n\right],\end{aligned}$$

which is bounded by a constant, combining Lemma 65 with the fact that the second term goes to 0 as $q \to 1$. □

3.2.6 Open Problems

As mentioned in Section 3.2.3, due to the lack of monotonicity, we cannot rule out that there exist more than one phase transition. However, we conjecture the following statement to be true.

Conjecture 66. *The function* $q \mapsto \mathbb{P}^{\eta}\left[\eta_t \notin \{0,1\}^{\mathbb{Z}} \text{ for all } t \geq 0\right]$ *is decreasing in* $q \in (0,1]$.

This would imply a critical value q_c such that if $q < q_c$ the infection survives with positive probability, while if $q > q_c$ the infection dies out with probability 1.

3.3 Opinion Formation

In this section, we study a model of opinion formation on a network given as a process on a graph. According to some order, each vertex gets to take a binary decision. For that, it first randomly decides to be in favor of one of the opinions. However, a vertex only publicly votes for its favorite opinion if it is not manipulated by its neighbors, i.e., if the number of neighboring voters for the respective opinions do not differ by more than some constant. If the vertex is manipulated by its neighbors, it follows the majority in its neighborhood and chooses to vote for the opinion voted for by most of its neighbors.

As the order in which vertices should take their votes influences the total outcome, finding to whatever means, a good order is a natural algorithmic question at hand. In particular, finding an order that maximizes the expected number of vertices voting for one opinion. Basically, we differentiate between two types of orders: Namely, non-adaptive orders, where the ordering in which the vertices are voting has to be fixed in advance, and adaptive orders, allowing for changes of the order after each vote. We also generalize orders to weak orders, allowing more than one vertex to take a vote simultaneously.

In the non-adaptive setting, we prove an upper bound and give a characterization of graphs for which there exists a weak order achieving this bound. Finally, we also present a polynomial time algorithm that computes a weak order adaptively with a best possible performance guarantee.

This section is based on joint work with Susanne Albers [1].

3.3.1 Motivation and Related Work

The binary opinion formation process is defined on a possibly directed graph and can informally be described as follows. According to some order, each vertex gets to take a vote on two opinions $\mathcal{Y}$ and $\mathcal{N}$. It decides to be in favor of opinion $\mathcal{Y}$ with probability p and to be in favor of opinion $\mathcal{N}$ with probability $1 - p$ independently of all other vertices. Then, the vertex votes for its favorite opinion if it is not manipulated by its neighbors, i.e., if the number of voters for $\mathcal{N}$ and the number of voters for $\mathcal{Y}$ that influence the given vertex do not differ by more than some constant c. But if they do, the vertex chooses to vote for the opinion voted for by most of its influencing neighbors.

In our model, we assume that the parameter/probability p and influence relations, which are resembled by a graph, are publicly known. However, initial opinions are not. We mainly study how to maximize the (expected) number of voters for one opinion, here $\mathcal{Y}$. Yet, we also give a result that can be used to design untampered voting processes, i.e., an order in which manipulations cannot occur.

This process of opinion formation and the interest in finding an order has many

applications in a variety of fields, especially in economic theory and social sciences, e.g, in many public elections.

Since the US elections in 2016 election rigging is on everyone's lips and our underlying model has similarities with the US presidential elections. The different ballot dates for the 50 states empose a weak order that is mostly fixed. Results from previous ballot days may affect upcoming elections, in particular the voters, in other states. In this work, we show that the order in which states hold their elections can have a significant impact on the outcome. More precisely, modifying the order in which states must hold their respective election days can help to promote a candidate or party. We also provide techniques to design an election process, in which, independent of the preferences, influence relations do affect the outcome.

Yet, another application from economy is the advertisement of products [3]. Opinions here need not to be binary, however, it is either choosing the product or choosing a competitor's product. Of course, customers are influenced by their friends or idols that already have chosen/voted for or against the item. Again, most likely, such influencing relations are not necessarily symmetric, motivating directed graphs. Obviously, maximizing the number of (expected) customers in the end maximizes the profit. Also, the model is applicable to advertisements where one would present ads to some network of individuals in some order to maximize the number of customers in the end. In social networks like Twitter, YouTube or Instagram for example, companies commonly use Influencer marketing, where products are presented by so-called influencers to their followers [22]. For more direct applications and motivation of the process, we would like to refer to the paper by Arthur [5], which introduces this process of opinion formation on undirected graphs and names several more applications.

Algorithmically, we want to compute an order that maximizes the expected number of vertices voting for $\mathcal{Y}$. Basically, we group orders into two types. First, we study orders where the order in which the vertices are voting is fixed in advance. Second, we allow the order to be changed adaptively, which means we decide for the next voter only after the previous voter made a decision. Orders of the first kind we call non-adaptive and name orders of the second kind adaptive orders. Both kinds of orders have already been considered, for example, by Chierichetti et al. in [25].

Needless to say that there are many variants of the opinion formation process itself. The model where the initial acceptance probabilities and the threshold values c may differ for each vertex was studied by Hajiaghayi et al. in [53]. In their model threshold values for each vertex may even be drawn at random from a given probability distribution. Their main results are regarding non-adaptive orders and mainly for the complete graph.

Other than choosing an order that maximizes the number of votes for one opinion other concepts such as seeding exist. The idea is to place seeds in the network to achieve a large number of voters in the end. For a short review of results and models see, e.g.,[59]. Last but not least, it is noteworthy that the model of opinion formation

has close relations to c-neighbor bootstrap percolation, as $\mathcal{Y}$ voting vertices can be interpreted as an infection spreading over the graph. If the initial opinion of each vertex was known publicly, the maximal achievable number of $\mathcal{Y}$ votes corresponds to the size of the infected set at the end of the corresponding c-neighbor bootstrap percolation process. We shall use bootstrap percolation as a bound for an optimal order for our opinion formation process in the subsection on computational results. For a short review of results, the reader is pointed to Section 3.1.1.

3.3.2 The Model

Given a directed graph $G = (V, A)$ on n vertices. Recall, given a vertex v, $\delta^-(v)$ denotes the in-neighbors of v, more precisely

$$\delta^-(v) := \big\{u \in V : (u, v) \in A\big\}.$$

In this section, we abuse notation and say a vertex v is adjacent to some other vertex u if $(u, v) \in A$. Note, in this terminology, if v is adjacent to u the vertex u not necessarily is adjacent to v.
We say a vertex u influences another vertex v if v is adjacent to u, more formally if $u \in \delta^-(v)$.

By $\mathbb{P}[\cdot]$ we always denote a probability measure and by $\mathbb{E}[\cdot]$ an expectation, where it always should be clear from the context which probability space we are dealing with. Usually, this space will be a product Bernoulli measure with parameter p on $\Omega = \{\mathcal{Y}, \mathcal{N}\}^n$ which is the set of all possible initial voting configurations of the vertices $V = [n]$. The parameter p always refers to some Bernoulli probability. Meanwhile, π shall always denote some (weak) order of the vertex set produced by an algorithm. Especially as the order π must not be an order that is fixed in advance, but might change depending on the votes, we would like to stress, that we slightly abuse notation here. Let the two random variables Y_π and N_π denote the number of vertices voting for opinion $\mathcal{Y}$ and opinion $\mathcal{N}$ at the end of the voting process according to the order π, respectively.

Having set most of our notation, following we give a formal definition of the process of opinion formation with parameters $p, \sigma, \upsilon_{\mathcal{I}}$ and $\upsilon_{\bar{\mathcal{I}}}$ as given for undirected graphs in [25]. The parameter $\upsilon_{\mathcal{I}}$ shall be the payoff that an individual receives if it votes for its initial opinion $\mathcal{I}$ and else receives $\upsilon_{\bar{\mathcal{I}}}$. For each influencing individual that voted for the same opinion, an individual receives an additional payoff of σ. We lift this definition to directed graphs.

Definition 67. *On a graph $G = (V, A)$, given parameters $p, \sigma, \upsilon_{\mathcal{I}}$ and $\upsilon_{\bar{\mathcal{I}}}$ as well as an order for the vertices, opinions form as follows:*

a) Initial Opinion: Once a vertex is supposed to vote it chooses its initial/favorite

opinion. This opinion is $\mathcal{Y}$ with probability p and else $\mathcal{N}$ independently of all other vertices.

b) Value: The value of its initial opinion $\mathcal{I} \in \{\mathcal{Y}, \mathcal{N}\}$ is $v_{\mathcal{I}}$, while the value of the opposite opinion $\bar{\mathcal{I}}$ is $v_{\bar{\mathcal{I}}}$. Note that we require $v_{\bar{\mathcal{I}}} < v_{\mathcal{I}}$.

c) Voting: A vertex $v \in V$ decides to vote for its favorite initial opinion $\mathcal{I} \in \{\mathcal{Y}, \mathcal{N}\}$ if $v_{\bar{\mathcal{I}}} + \delta m_{\bar{\mathcal{I}}}(v) > v_{\mathcal{I}} + \delta m_{\mathcal{I}}(v)$, where $m_{\bar{\mathcal{I}}}(v)$ is the number of vertices in $\delta^-(v)$ that already voted for $\bar{\mathcal{I}}$ and $m_{\mathcal{I}}(v)$ is the number vertices in $\delta^-(v)$ that voted for $\mathcal{I}$.

Remark 68. Note, there exists an equivalent formulation of c):

c') Define $c := \lceil \delta^{-1} | v_{\mathcal{I}} - v_{\bar{\mathcal{I}}} | \rceil$, then a vertex i with initial opinion $\mathcal{I} \in \{\mathcal{Y}, \mathcal{N}\}$ votes for $\mathcal{I}$ if $|m_{\mathcal{I}}(v) - m_{\bar{\mathcal{I}}}(v)| < c$ and else chooses the opinion voted by most of the vertices in $\delta^-[v]$.

In the rest of the section we shall make use of this equivalent formulation and stick to directed graphs. Therefore, we replace the input parameters of the process by p and c. For a given vertex v we denote by $\{v \to \mathcal{Y}\}$ and $\{v \to \mathcal{N}\}$ the event that vertex v is manipulated, that is, the vertex is made to vote for $\mathcal{Y}$ or $\mathcal{N}$ due to its neighbors, respectively. Also, we abbreviate the event that a vertex votes for some opinion $\mathcal{I} \in \{\mathcal{Y}, \mathcal{N}\}$ by $\{v \in \mathcal{I}\}$. Even though we do not write $p(n)$, we allow the process probability to be dependent on n.

3.3.3 Our Results

We close the introduction shortly presenting the structure and results of the remaining section.

In Section 3.3.4 we consider non-adaptive orders and our results shall serve as a motivation to consider adaptive orders. We recall some results which were recently given by Hajiaghayi et al. [52] and state one of our main theorems: Given $p < \frac{1}{2}$, expected number of vertices voting for $\mathcal{Y}$ is bounded by pn from above. Additionally, we characterize graphs that admit a non-adaptive order meeting the upper bound using the new notion of degeneracy introduced in Section 2.2.5.

Section 3.3.5 is split into several subsections. We generalize orders to weak orders of width at most k, allowing k vertices simultaneously and independently of each other to take a vote. If the number of simultaneous votes is exactly k, we speak about k-block weak orders. Note, the case of only one vertex taking a vote at a time, i.e. $k = 1$, has already been studied by Chierichetti et al. in [25] and is shortly revisited in Section 3.3.5. However, it is worth mentioning that we obtain slightly better bounds for our presented algorithm. Additionally, our proof makes use of a generalization of

the well known gambler's ruin game which is introduced in Section 3.3.5. The new bounds derived there might be an useful tool for other applications, too.

Weak orders with a fixed as well as a varying number of simultaneously voting vertices shall both be covered in Section 3.3.5. Again, we give tight bounds for the presented algorithms.

Finally, in Section 3.3.6 we analyze the given algorithms computationally on randomized inputs and compare their quality.

3.3.4 Non-adaptive Orders

This subsection shall motivate studying adaptive orders in more detail. Therefore, let us present one of our main results.

Theorem 69. *Given an arbitrary directed n vertex graph G, $0 < p < 1$, and $p \neq \frac{1}{2}$ as well as $c \geq 2$.*
Then,

$$\mathbb{E}[Y_\pi] \begin{cases} \leq pn & \text{if } p \leq \frac{1}{2}, \\ \geq pn & \text{if } p \geq \frac{1}{2} \end{cases}$$

for all non-adaptive weak orders π.

Furthermore, there exists a non-adaptive weak order π of width at most k, with $\mathbb{E}[Y_\pi] = pn$ if and only if G is $(k, c-1)$-degenerate.
Similarly, there exists a non-adaptive weak order π of width at most k with $\mathbb{E}[Y_\pi] = pn$ if and only if G is exact-$(k, c-1)$-degenerate.

In order to prove Theorem 69 we state a lemma, which was given by Hajiaghayi et al. that gave rise to the theorem.

Lemma 70 ([52, Corollary 1.2])**.** *Given an arbitrary graph $G = (V, A)$ and $p < \frac{1}{2}$, then for any vertex $v \in V$ and a given non-adaptive order π we have that $\mathbb{P}[v \in \mathcal{Y}] \leq p$. This immediately implies that $\mathbb{E}[Y_\pi] \leq pn$.*
Similarly, given an arbitrary graph $G = (V, A)$ and $p > \frac{1}{2}$, then for any vertex $v \in V$ and a given non-adaptive order π we have that $\mathbb{P}[v \in \mathcal{Y}] \geq p$. This immediately implies that $\mathbb{E}[Y_\pi] \geq pn$.

Next, we prove Theorem 69.

Proof of Theorem 69. One direction is obvious. If the graph G is $(k, c-1)$-degenerate, then there exists a weak order π on the vertices of width at most k. Voting according to this weak order π gives that $\mathbb{E}[Y_\pi] = pn$ because every vertex votes independently.

For the other direction, assume first that $0 < p < \frac{1}{2}$. Lemma 70 implies that we have $\mathbb{P}[v \in \mathcal{Y}] \leq p$ for all $v \in V$. We require an order π with $\mathbb{E}[Y_\pi] = pn$ to exist, and therefore, we have that $\mathbb{P}[v \in \mathcal{Y}] = p$ for all $v \in V$. We now claim that this is

only possible if the graph is $(k, c-1)$-degenerate, and, actually, the order gives rise to a weak order, that fulfills the conditions of $(k, c-d)$-degeneracy. To prove that, we make use of the fact that each vertex must vote independently as otherwise there was a first vertex v which could be manipulated by its neighbors. Since $p < \frac{1}{2}$, $c \geq 2$, and, as every vertex in the neighborhood of v must have chosen its vote independently, we have that

$$\begin{aligned}\mathbb{P}[v \in \mathcal{Y}] &= p\,(1 - \mathbb{P}[v \to \mathcal{Y}] - \mathbb{P}[v \to \mathcal{N}]) + \mathbb{P}[v \to \mathcal{Y}] \\ &= p + (1-p)\mathbb{P}[v \to \mathcal{Y}] - p\mathbb{P}[v \to \mathcal{N}].\end{aligned}$$

Note,

$$(1-p)\mathbb{P}[v \to \mathcal{Y}] - p\mathbb{P}[v \to \mathcal{N}] \geq 0$$

as

$$\mathbb{P}[v \to \mathcal{Y}] \overset{(*)}{\leq} \left(\frac{p}{1-p}\right)^c \mathbb{P}[v \to \mathcal{N}] < \frac{p}{1-p}\mathbb{P}[v \to \mathcal{N}]$$

where $(*)$ follows by mapping each configuration in the event $v \to \mathcal{Y}$ to an unique configuration in the event $v \to \mathcal{N}$ by flipping each vertex's initial opinion. Since there are at least c more $\mathcal{Y}$ votes than $\mathcal{N}$ votes, the probability of the mapped to configuration is at least $\left(\frac{1-p}{p}\right)^c$ times as large. Additionally, due to $p < \frac{1}{2}$ and $c \geq 2$, we have that $\left(\frac{p}{1-p}\right)^c < \frac{p}{1-p}$. This finally implies that

$$\mathbb{P}[v \in \mathcal{Y}] = p + (1-p)\mathbb{P}[v \to \mathcal{Y}] - p\mathbb{P}[v \to \mathcal{N}] < p \tag{3.3}$$

which contradicts our assumption that $\mathbb{P}[v \in \mathcal{Y}] = p$.

Therefore, each vertex must vote independently, hence, the order of votes indeed gives rise to a weak order fulfilling the conditions of $(k, c-1)$-degeneracy.
This also extends, to $\frac{1}{2} < p < 1$, due to symmetry of N_π and Y_π.
Furthermore, the second part of the theorem follows completely analogously. □

We would like to stress that equivalence in Theorem 69 only holds for $c \geq 2$ as for $c = 1$ any graph admits an order π such that $\mathbb{E}[Y_\pi] = pn$. For example, an order in a breadth first search manner, start with any vertex and subsequently append all new neighbors. Here, the vertex voting first in each connected component determines the votes of the whole component. It is easy to see using linearity of expectation that this order fulfills $\mathbb{E}[Y_\pi] = pn$. Also, it is necessary that $0 < p < 1$ and $p \neq \frac{1}{2}$ holds true, which is due to the fact that for $p \in \{0, \frac{1}{2}, 1\}$, any order π fulfills $\mathbb{E}[Y_\pi] = pn$. Our proof fails for $p = \frac{1}{2}$ because $\frac{p}{1-p} = 1$, and therefore we do not have a strict inequality in (3.3).

Next, we address computational complexity of the decision problem, i.e. deciding whether a graph admits a non-adaptive order π such that $\mathbb{E}[Y_\pi] = pn$. We prove this

decision problem is *NP*-complete.

Lemma 71. *Given an arbitrary graph $G = (V, A)$, $0 < p < 1$ and $p \neq \frac{1}{2}$ as well as $c \geq 3$, then the decision problem whether the graph admits a non-adaptive weak order π of width at most k, for some variable k, such that $\mathbb{E}[Y_\pi] = pn$ is NP-complete.*

Proof. Theorem 23 together with Theorem 69 implies that the decision problem is *NP*-complete for $c \geq 3$. □

But, we may use the approximation of Algorithm 3 to find a weak order π of some width that fulfills $\mathbb{E}[Y_\pi] = pn$. This high number, however, might not be feasible with the underlying application. In particular, this motivates finding better approximation algorithms for k_d.

Additionally, we want to mention that a random order has the following guarantee.

Lemma 72. *Given an arbitrary graph $G = (V, A)$ with maximum in-degree $\Delta^- \geq c - 1$ and $p < \frac{1}{2}$. Let π be an order drawn uniformly at random from the set of all permutations of V. Then, $\mathbb{E}[Y_\pi] \geq \frac{c-1}{\Delta^-} pn$.*

Proof. This bound can easily be obtained by the fact that every vertex i votes independently if not more than $c - 1$ neighbors have been taken a vote already. I.e. if the number of neighbors that received a smaller label than i is less than c. If $\deg^-(i) > 0$, the probability of this event is $\min\{\frac{c-1}{d^-(i)}, 1\}$. As these vertices vote independently with probability p for $\mathcal{Y}$ we have that

$$\mathbb{E}_\pi[\mathbb{E}[Y_\pi]] \geq \sum_{i \in [n]} p \cdot \min\left\{\frac{c-1}{d^-(i)}, 1\right\} \geq \sum_{i \in [n]} p \cdot \frac{c-1}{\Delta^-} = pn \cdot \frac{c-1}{\Delta^-}.$$

The expectation is larger than the claimed value, thus there exists at least one order exceeding the given bound. This finishes the proof. □

Note, this guarantee is independent of the number of vertices voting simultaneously.

Next, we give a lemma on the structure of optimal orders. In particular, the lemma states that optimal orders are not globally optimal, which, to some extend sounds counter-intuitive. More precisely, orders for certain values p are not necessarily optimal for only slightly larger $p' \leq \frac{1}{2}$.

Lemma 73. *Given a graph $G = (V, E)$. An optimal order π_1 for some given probability $p^* < \frac{1}{2}$, is not necessarily optimal for p' with $p^* < p' < \frac{1}{2}$. Formally, there exists a graph, with an order π_1 that maximizes $\mathbb{E}_{p^*}[Y_\pi]$, but $\mathbb{E}_{p'}[Y_{\pi_1}] < \mathbb{E}_{p'}[Y_{\pi_2}]$ for some other order π_2.*

Proof. Let $c = 2$. A graph with such a property is for example K_8 missing and edge. Let the missing edge be $\{1, 2\}$. We now claim, if $p^* < \frac{1}{1000}$, $\pi_1 = (3, 1, 2, 4, 5, 6, 7, 8)$

is an optimal order. But if $p' > 1 - \frac{1}{\sqrt{2}}$, we have that $\pi_2 = (1, 2, 3, 5, 6, 7, 8)$ fulfills $\mathbb{E}_{p'}[Y_{\pi_1}] < \mathbb{E}_{p'}[Y_{\pi_2}]$. The graph is drawn in Figure 3.13.

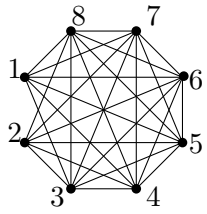


Figure 3.13: K_8 missing an edge.

Let us compute $\mathbb{E}_p[Y_{\pi_1}]$ and $\mathbb{E}_p[Y_{\pi_2}]$ for general p. Consider π_1 first and observe the following:

1. First of all,
$$\mathbb{P}[Y_{\pi_2} = 8] = p^2,$$
since both, vertex 1 and 2 vote independently and manipulate every other vertex to vote for $\mathcal{Y}$ if both of them voted $\mathcal{Y}$.

2. Observe, $Y_{\pi_2} = 7$ implies that 1 and 2 must have opposing preferences and the following two vertices $3, 4$ must prefer $\mathcal{Y}$. Thus,
$$\mathbb{P}[Y_{\pi_2} = 7] = 2p(1-p)p^2 = 2p^3(1-p),$$

3. Also, $Y_{\pi_2} = 6$ implies that 1 and 2 must have opposing votes $\mathcal{Y}$ and so the following two vertices $3, 4$. Consequently, $5, 6$ must prefer $\mathcal{Y}$. Hence,
$$\mathbb{P}[Y_{\pi_2} = 6] = 2p(1-p)2p(1-p)p^2 = 4p^4(1-p)^2,$$

4. Just like before, $Y_{\pi_2} = 5$ implies that 1 and 2 must have preferences and so the following pairs $3, 4$ and $5, 6$. The remaining vertices 7 and 8 must prefer $\mathcal{Y}$. Hence,
$$\mathbb{P}[Y_{\pi_2} = 5] = 2p(1-p)2p(1-p)2p(1-p)p^2 = 8p^5(1-p)^3,$$

5. Same argumentation as above gives,
$$\mathbb{P}[Y_{\pi_2} = 4] = 2p(1-p)2p(1-p)2p(1-p)2p(1-p) = 16p^4(1-p)^4$$

6. Finally, using symmetry of N_{π_2} and Y_{π_2} we obtain, interchanging p and $1-p$ in

above formulas,

$$\mathbb{P}[Y_{\pi_2} = 3] = 8p^3(1-p)^5,$$
$$\mathbb{P}[Y_{\pi_2} = 2] = 4p^2(1-p)^4,$$
$$\mathbb{P}[Y_{\pi_2} = 1] = 2p(1-p)^3.$$

In total, we get that

$$\begin{aligned}\mathbb{E}[Y_{\pi_2}] &= 8(p^2) + 7(2p^3(1-p)) + 6(4p^4(1-p)^2) + 5(8p^5(1-p)^3) + 4(16p^4(1-p)^4) \\ &\quad + 3(8p^3(1-p)^5) + 2(4p^2(1-p)^4) + (2p(1-p)^3) \\ &= 2p + 10p^2 + 12p^3 - 56p^5 + 56p^6 - 16p^7.\end{aligned}$$

In the same way we can compute $\mathbb{E}[Y_{\pi_1}]$:

1. We have that,
$$\mathbb{P}[Y_{\pi_1} = 8] = p^3.$$
Which is due to the fact that all three vertices $1, 2, 3$ must prefer $\mathcal{Y}$. IF they do, all remaining vertices are manipulated to vote for $\mathcal{Y}$.

2. Next,
$$\mathbb{P}[Y_{\pi_1} = 7] = 3p^2(1-p)p = 3p^3(1-p),$$
due to the fact that exactly one vertex in $\{1, 2, 3\}$ must vote for $\mathcal{N}$. Another vote for $\mathcal{N}$ by 4 suffices again to manipulated the remaining vertices.

3. Consider the event that $Y_{\pi_1} = 6$. With analogous arguments, within $1, 2, 3$ there must exist at least one vertex preferring $\mathcal{Y}$ and one vertex preferring $\mathcal{N}$. The next vertex then votes independently and must prefer the opinion of the minority. Otherwise, the remaining vertices would all be manipulated. Thus,
$$\mathbb{P}[Y_{\pi_1} = 6] = 3p^2(1-p)(1-p)p^2 + 3p(1-p)^2p(1-p)p^2 = 6p^4(2-p)^2.$$

4. Similarly,
$$\mathbb{P}[Y_{\pi_1} = 5] = 3p^2(1-p)(1-p)2p(1-p)p^2 + 3p(1-p)^2p2p(1-p)p^2 = 12p^5(1-p)^3.$$

5. Additionally,
$$\begin{aligned}\mathbb{P}[Y_{\pi_1} = 4] &= 3p^2(1-p)(1-p)2p(1-p)2p(1-p) + 3p(1-p)^2p2p(1-p)2p(1-p) \\ &= 24p^4(1-p)^4.\end{aligned}$$

6. Finally, again interchanging p and $1-p$ we obtain

$$\mathbb{P}[Y_{\pi_1} = 3] = 12p^5(1-p)^3,$$
$$\mathbb{P}[Y_{\pi_1} = 2] = 6p^2(1-p)^4,$$
$$\mathbb{P}[Y_{\pi_1} = 1] = 3p(1-p)^3.$$

In total, we get that

$$\begin{aligned}\mathbb{E}[Y_{\pi_1}] &= 8(p^3) + 7(3p^3(1-p)) + 6(6p^4(1-p)^2) + 5(12p^5(1-p)^3) + 4(24p^4(1-p)^4) \\ &\quad + 3(12p^3(1-p)^5) + 2(6p^2(1-p)^4) + (3p(1-p)^3) \\ &= 3p + 3p^2 + 26p^3 - 84p^5 + 84p^6 - 24p^7.\end{aligned}$$

Therefore,

$$\mathbb{E}[Y_{\pi_1}] - \mathbb{E}[Y_{\pi_2}] = p - 7p^2 + 14p^3 - 28p^5 + 28p^6 - 8p^7,$$

which implies that $\mathbb{E}[Y_{\pi_1}] < \mathbb{E}[Y_{\pi_2}]$ if $\frac{1}{\sqrt{2}} < p < \frac{1}{2}$.
Last but not least, we need to verify that π_1 is indeed an optimal order for small enough p. By linearity of expectation, it immediately follows that $\mathbb{E}[Y_{\pi_1}] \geq 3p$. Any other order π (non-isomorphic to π_1, π_2) begins with two vertices, that are both adjacent to all other vertices. Overestimating $\mathbb{E}[Y_\pi]$, assuming that two of three votes within the first three vertices suffices to manipulate every other vertex to $\mathcal{Y}$, yields,

$$\mathbb{E}[Y_{\pi_1}] \leq 8p^2 + 7(2p(1-p))p = 22p^2 - 14p^3 \overset{p \text{ small}}{\leq} 3p = \mathbb{E}[Y_{\pi_1}].$$

□

As we have seen in Lemma 70 there is no non-adaptive order that yields in expectation more than pn vertices voting for $\mathcal{Y}$ if $p < \frac{1}{2}$. Also, optimal orders may change with slight increases of p. From an application point of view, especially if p is not known exactly, this lack of knowledge might have a big impact on the order that should be chosen. This also motivates the following subsection where we study adaptive orders that theoretically allow obtaining an expected number of pn vertices voting for $\mathcal{Y}$ if $p < \frac{1}{2}$.

3.3.5 Adaptive Orders

In this subsection, we address adaptive (weak) orders aiming to maximize the expected number of vertices voting for $\mathcal{Y}$. In particular, we study the case when more than one vertex can take a vote at a time. Here, vertices that vote simultaneously vote

independently of each other, and, are influenced by vertices that voted previously. One might think that increasing the number of simultaneous votes should increase the expected number of $\mathcal{Y}$ votes. But as it turns out, this not necessarily the case. In particular, if the number at each time step may not vary but is fixed to k, it is easy to see that increasing this number can result in a decreasing expected number of votes for $\mathcal{Y}$. For an easy example let $r = 1$, $p < \frac{1}{2}$ and G be a graph consisting of two disjoint edges. If $k = 2$, we achieve an expected number of $\mathcal{Y}$ votes of $4p$. However, if $k = 3$, an optimal order can achieve at most $3p + (1 - p) < 4p$.

Warm-up: Adaptive Orders — One Vertex at a Time

First, we give an adaptive algorithm that maximizes the expected number of vertices voting for $\mathcal{Y}$ if only one vertex votes at a time. As mentioned before this question has been studied by Chierichetti et al. [25]. The authors come up with an algorithm that computes an adaptive order on an undirected graph G, which easily can be extended to directed graphs.
The algorithm by Chierichetti et al. in [25] is given below.

Algorithm 5: Adaptive Order [Chierichetti et al. [25]]

Input: Graph $G = (V, E)$
Output: Adaptive order of V

Let $W \subseteq V$ be any set inducing a maximal $(c-1)$-degenerate graph in G;
Let $v_1, \ldots, v_{|W|}$ be an Erdős-Hanjal sequence of the nodes in W;
for $i = 1, \ldots, |W|$ **do**
    Append v_i to π;
    while *there exists* $v \in V \backslash W$ *with exactly* c *neighbors in* W*, all of which have chosen* $\mathcal{Y}$ **do**
        Append v to π;
    end
end
Append the remaining vertices to π in arbitrary order;

Given a graph on n vertices, process probability p, and threshold value c, then the expected number $\mathbb{E}[Y_\pi]$ of vertices voting for $\mathcal{Y}$ using the adaptive order π produced by Algorithm 5 fulfills $\mathbb{E}[Y_\pi] \geq p^c n$, c.f. [25, Theorem 2.1.].

In the following, we present a simpler algorithm that performs slightly better than Algorithm 5 in terms of the expected number of vertices voting for $\mathcal{Y}$ for all values $p > 0$. Let us first introduce some notation. A vertex v that has not been appended to the order and is currently not manipulated by its neighbors, i.e. currently $|m_I(v) - m_{\bar{I}}(v)| < c$, is called to be *active*. If such a vertex is momentarily manipulated by its neighbors, i.e. currently $|m_I(v) - m_{\bar{I}}(v)| \geq c$, it is called to be *passive*. Note, every vertex is either active, passive or has already been appended to the order. In the beginning, every vertex is active.

Algorithm 6: Improved Adaptive Order

Input: Graph $G = (V, A)$
Output: Adaptive order π of V

```
while there exists an active node v ∈ V do
    Schedule v;
    while there exists a passive node v ∈ V with m_𝒴(v) − m_𝒩(v) ≥ c do
        Append v to π;
    end
end
Append the remaining vertices to π in arbitrary order;
```

Just like in Algorithm 5 one could preprocess the vertex set and compute a maximal $(c-1)$-degenerate graph first. But as it turns out, this preprocessing does not give any increases in terms of global bounds for general graphs. Also, our algorithm can be implemented such that the algorithm runs in $\mathcal{O}(n^2 \log(n))$ time using simple data structures.

The following theorem gives a bound on the expected value of Y after processing Algorithm 6 on any graph.

Theorem 74. *Given an arbitrary directed n vertex graph, process probability p and threshold c. Then, the expected number of vertices $\mathbb{E}[Y_\pi]$ that vote $\mathcal{Y}$ after proceeding Algorithm 6 fulfills*

$$\mathbb{E}[Y_\pi] \geq \begin{cases} \frac{1}{1+\left(\frac{1-p}{p}\right)^c} \cdot n & \text{for } p < \frac{1}{2}, \\ p \cdot n & \text{for } p \geq \frac{1}{2}. \end{cases}$$

The bounds in both ranges of p are sharp for general graphs. On the one hand, for K_n, the bound is best possible for $p < \frac{1}{2}$ and asymptotically in n. On the other hand, consider the stable set on n vertices. Then, any (even non-adaptive) order will return in expectation $p \cdot n$ many $\mathcal{Y}$ votes.
We want to mention that Algorithm 7 presented in the next subsection is a generalization of Algorithm 6 and Theorem 76 implies the aforementioned Theorem 74. Therefore, we omit a prove of Theorem 74 at this point. Finally, observe that for all $p > 0$ we have that

$$p^c n \leq \begin{cases} \frac{1}{1+\left(\frac{1-p}{p}\right)^c} \cdot n & \text{for } p < \frac{1}{2} \\ p \cdot n & \text{for } p \geq \frac{1}{2}. \end{cases}$$

More than one Vertex at a Time

In this subsection, we turn our attention to weak orders where more than one vertex may take a vote at a time. We differ between k-block weak orders where the number

remains constant, i.e. constantly $k \geq 1$, for all but one time step and weak orders, where the number may vary but must not exceed some constant $k \geq 1$. We refer to weak orders of the second kind as variable orders.

But first, before going to positive results and presenting our algorithms, we give a bound on the maximum number of $\mathcal{Y}$'s in terms of the independence number $\alpha(G)$ as well as c, k and p for both, k-block and variable blocksize weak orders. Recall, an independent set on a directed graph is a set of vertices where no vertex has an arc towards any other vertex in the set.

Lemma 75. *Given an arbitrary graph G on n vertices, with independence number α, and $p < \frac{1}{(3c+k-3)\alpha}$. Then, any adaptive order π of width at most k, fulfills*

$$\mathbb{E}[Y_\pi] \leq n \cdot \frac{((3c+k-3)\alpha \cdot p)^c}{1-(3c+k-3)\alpha \cdot p} + (c-1).$$

Proof. We extend the idea of proof of [25, Lemma 4.1].
Consider any weak order π' of the vertices, then, if at most $c-1$ vertices choose $\mathcal{Y}$ during the process, every vertex that is adjacent to[7] at least $3c-2$ vertices that already voted, chooses deterministically $\mathcal{N}$. Let S denote the set of vertices that are influenced by at most $3c-3$ vertices that took a vote at the moment they are supposed to vote according to π'. Then $G[S]$ is $(3c+k-3)$-colorable as by construction of the set S, we have that $G[S]$ is $(k, 3c-2)$-degenerate, and hence, using Lemma 26 the claimed bound on the coloring number follows. As the independence number of G is α, we can bound the size of S by $|S| \leq (3c+k-3)\alpha$. If at most $c-1$ vertices in S choose to vote for $\mathcal{Y}$, we get that all vertices outside S deterministically choose $\mathcal{N}$, and hence we have $Y_{\pi'} \leq c-1$. The probability that this is not the case, namely that at least c vertices in S vote for $\mathcal{Y}$, can be bounded by

$$\begin{aligned}\sum_{i=c}^{|S|} \binom{|S|}{i} p^i (1-p)^{|S|-i} &\leq \sum_{i=c}^{|S|} (|S| \cdot p)^i \leq \sum_{i=c}^{|S|} ((3c+k-3)\alpha \cdot p)^i \\ &= \frac{((3c+k-3)\alpha \cdot p)^c - ((3c+k-3)\alpha \cdot p)^{(3c+k-3)\alpha}}{1-(3c+k-3)\alpha \cdot p} \\ &\leq \frac{((3c+k-3)\alpha \cdot p)^c}{1-(3c+k-3)\alpha \cdot p}.\end{aligned}$$

Here, we made use of $p < \frac{1}{(3c+k-3)\alpha}$ and that every vertex in S votes independently of all other vertices in S. Thus, no matter how the adaptive order π is chosen, the probability that there are more than $c-1$ vertices voting for $\mathcal{Y}$, is bounded from above

[7] is influenced by, respectively

by

$$\frac{((3c+k-3)\alpha \cdot p)^c}{1-(3c+k-3)\alpha \cdot p}.$$

Finally, observe that Y is always bounded from above by n. So we get that

$$\mathbb{E}[Y_\pi] \leq n \cdot \frac{((3c+k-3)\alpha \cdot p)^c}{1-(3c+k-3)\alpha \cdot p} + (c-1).$$

□

A Fixed Number of Vertices at a Time

Let $k \geq 1$ be the fixed number of vertices that take a vote simultaneously. Note that we would have to assume that $k \mid n$, but to keep it general if $k \nmid n$ we allow $n \mod k$ vertices to vote in the last step. The first important observation is, given an optimal order π, increasing k must not increase $\mathbb{E}[Y_\pi]$. Even more, increasing k might decrease the value of an optimal order π', that is $\mathbb{E}[Y_\pi] > \mathbb{E}[Y_{\pi'}]$ as we have seen at the beginning of Section 3.3.5. Next, we present an algorithm that computes an adaptive order when k vertices must vote simultaneously. We again use the notation as for Algorithm 6.

Algorithm 7: k-block Weak Orders

Parameter: c
Input: $G = (V, A)$
Output: Weak order π of V with width $\leq k$

Let $Q = V$ and $P = \emptyset$;
while *there exits* $v \in Q \cup P$ **do**
  if $|P| < k$ **then**
    Take some set $S' \subseteq Q$ with $|S'| = k - |P|$;
    Let $S = P \cup S'$;
  else
    let $S = P$;
  end
  if $|S| > k$ **then**
    Remove arbitrary vertices from S such that $|S| = k$;
  end
  if $|S| < k$ **then**
    Add arbitrary vertices from $Q \cup P$ to S such that $|S| = k$
  end
  Append S to π ;
  Compute the set of active vertices Q;
  Compute the set of passive vertices P, with $m_{\mathcal{Y}}(v) - m_{\mathcal{N}}(v) \geq c$;
end
Append the remaining nodes to π

Note, this algorithm can be implemented in such a way that it has running time $\Theta(n^2 \log(n))$. The implementation follows the implementation of Algorithm 6 by setting the initial difference $(m_{\mathcal{Y}} - m_{\mathcal{N}})$ counter for each vertex to 0. Now append the k vertices with the largest counters and update the counters of the vertices influenced by these. These are at most n, but as k might be some parameter in n it takes k operations to compute the new values. After that, we have to compute a new order. Due to the fact that each vertex votes exactly once, we have found an implementation taking $\Theta((n/k) \cdot k \cdot n \cdot \log(n))$ time.

The following theorem gives a bound on $\mathbb{E}[Y_\pi]$ when processing Algorithm 7.

Theorem 76. *Given an arbitrary directed n vertex graph, process probability p, threshold c and parameter k. The weak order π produced by Algorithm 7 fulfills*

$$\mathbb{E}[Y_\pi] \geq \begin{cases} \frac{1-\left(\frac{1-p}{p}\right)^{c+k-1}}{1-\left(\frac{1-p}{p}\right)^{2(c+k-1)}} \cdot n & \text{for } p < \frac{1}{2} \\ p \cdot n & \text{for } p \geq \frac{1}{2}. \end{cases}$$

Observe, for $k = 1$ the statement corresponds to Theorem 74. Moreover, it is easy to see that for $k = 1$ the orders produced by Algorithm 6 and Algorithm 7 agree.

For the analysis of the algorithm and the proof of the Theorem 76 let us first consider a slightly altered gambler's ruin game. The game, more or less, resembles votes taking place in the neighborhood of a single vertex before it gets to vote itself. We shall use the analysis of winning/bankruptcy probabilities to bound manipulation probabilities later.

Definition 77 (Generalized Gambler's Ruin Game)**.** *Two Players start with $A_0^k = a$ and $B_0^k = b$ coins each. They repeatedly play k games simultaneously where player A wins each game independently with probability $0 \leq p \leq 1$ and else player B wins the game. Let α_i be the number of games player A won in the ith round of games, and $\beta_i := k - \alpha_i$ be the number of games player B won, respectively. Then, we define the amount of coins the players have after the ith round of games to be $A_i^k = A_{i-1}^k + \alpha_i - \beta_i$ and $B_i^k = B_{i-1}^k + \beta_i - \alpha_i$. Once either $A_i^k \leq 0$ or $B_i^k \leq 0$ the game stops and the corresponding player is bankrupt. For consistency reasons, once the game stops, we set $A_j^k = A_i^k$ and $B_j^k = B_i^k$ for all $j > i$.*

Next, we prove a lemma on the probabilities of bankruptcy of the players for the altered gambler's ruin game. This lemma shall prove useful in the proof of Theorem 76.

Lemma 78. *In the symmetric altered gambler's ruin game, i.e. for $a = b = c$, the probability that player A is bankrupt until the mth round of k simultaneous played*

games fulfills

$$\frac{\mathbb{P}[\exists 0 < i \leq m : A_i^k = 0]}{\mathbb{P}[\exists 0 < i \leq m : B_i^k = 0]} \leq \begin{cases} \left(\frac{1-p}{p}\right)^c & \text{if} \quad p \geq \frac{1}{2}, \\ \left(\frac{1-p}{p}\right)^{c+k-1} & \text{if} \quad p < \frac{1}{2}. \end{cases}$$

for all m such that $mk \geq c$, and hence

$$\frac{\mathbb{P}[\exists i > 0 : A_i^k = 0]}{\mathbb{P}[\exists i > 0 : B_i^k = 0]} \leq \begin{cases} \left(\frac{1-p}{p}\right)^c & \text{if} \quad p \geq \frac{1}{2}, \\ \left(\frac{1-p}{p}\right)^{c+k-1} & \text{if} \quad p < \frac{1}{2}. \end{cases}$$

Proof. We prove the lemma by induction on the number m of rounds of games. First observe that if $k \cdot m \geq c$, simply due to symmetry every sequence of $m \cdot k$ games such that $B_m^k \leq 0$ and $B_{m-1}^k > 0$ has an uniquely corresponding sequence of $m \cdot k$ games such that $A_m^k \leq 0$ and $B_{m-1}^k > 0$. Dividing the two probabilities of each sequence gives

$$\frac{(1-p)^\alpha p^\beta}{p^\alpha (1-p)^\beta} = (1-p)^{\alpha-\beta} p^{\beta-\alpha},$$

where $c \leq \alpha - \beta \leq c + k - 1$, and hence

$$\frac{\mathbb{P}[A_m^k \leq 0, A_{m-1}^k > 0]}{\mathbb{P}[B_m^k \leq 0, B_{m-1}^k > 0]} \leq \begin{cases} \left(\frac{1-p}{p}\right)^c & \text{if} \quad p \geq \frac{1}{2}, \\ \left(\frac{1-p}{p}\right)^{c+k-1} & \text{if} \quad p < \frac{1}{2}. \end{cases} \tag{3.4}$$

So for the smallest m such that $m \cdot k \geq c$, the statement holds, since

$$\frac{\mathbb{P}[\exists 0 < i \leq 1 : A_i^k \leq 0]}{\mathbb{P}[\exists 0 < i \leq 1 : B_i^k \leq 0]} = \frac{\mathbb{P}[A_1^k \leq 0, A_0^k > 0]}{\mathbb{P}[B_1^k \leq 0, B_0^k > 0]} \leq \begin{cases} \left(\frac{1-p}{p}\right)^c & \text{if} \quad p \geq \frac{1}{2}, \\ \left(\frac{1-p}{p}\right)^{c+k-1} & \text{if} \quad p < \frac{1}{2}. \end{cases}$$

Let the statement hold for m, then if $p \geq \frac{1}{2}$

$$\begin{aligned} \frac{\mathbb{P}[\exists 0 < i \leq m+1 : A_i^k \leq 0]}{\mathbb{P}[\exists 0 < i \leq m+1 : B_i^k \leq 0]} &= \frac{\mathbb{P}[\exists 0 < i \leq m : A_i^k \leq 0] + \mathbb{P}[A_{m+1}^k \leq 0,\ A_m^k > 0]}{\mathbb{P}[\exists 0 < i \leq m : B_i^k \leq 0] + \mathbb{P}[B_{m+1}^k \leq 0,\ B_m^k > 0]} \\ &\overset{(3.4)}{\leq} \frac{\mathbb{P}[\exists 0 < i \leq m : A_i^k \leq 0] + \left(\frac{1-p}{p}\right)^c \mathbb{P}[B_{m+1}^k \leq 0,\ B_m^k > 0]}{\mathbb{P}[\exists 0 < i \leq m : B_i^k \leq 0] + \mathbb{P}[B_{m+1}^k \leq 0,\ B_m^k > 0]} \\ &\overset{\text{Ind.}}{\leq} \frac{\left(\frac{1-p}{p}\right)^c \left(\mathbb{P}[\exists 0 < i \leq m : B_i^k \leq 0] + \mathbb{P}[B_{m+1}^k \leq 0,\ B_m^k > 0]\right)}{\mathbb{P}[\exists 0 < i \leq m : B_i^k \leq 0] + \mathbb{P}[B_{m+1}^k \leq 0,\ B_m^k > 0]} = \left(\frac{1-p}{p}\right)^c. \end{aligned}$$

While, if $p < \frac{1}{2}$

$$\begin{aligned}
\frac{\mathbb{P}[\exists 0 < i \leq m+1 : A_i^k \leq 0]}{\mathbb{P}[\exists 0 < i \leq m+1 : B_i^k \leq 0]} &= \frac{\mathbb{P}[\exists 0 < i \leq m : A_i^k \leq 0] + \mathbb{P}[A_{m+1}^k \leq 0,\ A_m^k > 0]}{\mathbb{P}[\exists 0 < i \leq m : B_i^k \leq 0] + \mathbb{P}[B_{m+1}^k \leq 0,\ B_m^k > 0]} \\
&\overset{(3.4)}{\leq} \frac{\mathbb{P}[\exists 0 < i \leq m : A_i^k \leq 0] + \left(\frac{1-p}{p}\right)^{c+k-1} \mathbb{P}[B_{m+1}^k \leq 0,\ B_m^k > 0]}{\mathbb{P}[\exists 0 < i \leq m : B_i^k \leq 0] + \mathbb{P}[B_{m+1}^k \leq 0,\ B_m^k > 0]} \\
&\overset{\text{Ind.}}{\leq} \frac{\left(\frac{1-p}{p}\right)^{c+k-1} \left(\mathbb{P}[\exists 0 < i \leq m : B_i^k \leq 0] + \mathbb{P}[B_{m+1}^k \leq 0,\ B_m^k > 0]\right)}{\mathbb{P}[\exists 0 < i \leq m : B_i^k \leq 0] + \mathbb{P}[B_{m+1}^k \leq 0,\ B_m^k > 0]} \\
&= \left(\frac{1-p}{p}\right)^{c+k-1}.
\end{aligned}$$

As this holds for all values of m, it also holds in the limit, which proves the lemma. □

Corollary 79. *If $p \leq \frac{1}{2}$, Lemma 78 directly implies that*

$$\mathbb{P}[\exists i > 0 : A_i^k = 0] \leq \frac{\left(\frac{1-p}{p}\right)^{c+k-1}}{1 + \left(\frac{1-p}{p}\right)^{c+k-1}}.$$

Next, we prove Theorem 76.

Proof of Theorem 76. Similarly to the proof of Theorem 74, as long as there are more than k vertices in $A \cup P$, that is, there are at least k active or passive vertices that are currently manipulated towards $\mathcal{Y}$, every vertex appended to π in line 15 of Algorithm 7 votes for $\mathcal{Y}$ with probability at least p. We now interpret every repetition of line 15 in which k vertices vote simultaneously as a round the introduced altered gambler's ruin game. Even though the votes are not necessarily independent, the gambler's ruin problem dominates our model. We bound the expected number of vertices voting for $\mathcal{N}$ in the following way. Every vertex, once it is about to vote according to the order, is either active or passive. If it is active, it votes for $\mathcal{N}$ with probability $(1-p)$. The probability of being passive can be split into the probabilities $\rho_i = \mathbb{P}[m_{\mathcal{N}}(v) - m_{\mathcal{Y}}v) \geq c]$ and $\sigma_i = \mathbb{P}[m_{\mathcal{Y}}(v) - m_{\mathcal{N}}(v) \geq c]$. Next, observe that

$$\rho_i \leq \mathbb{P}[\exists i > 0 : A_i^k \leq 0]$$

which is due to the fact that active vertices vote independently according to the Bernoulli law, and, in any other case, only vertices get to take a vote if they are manipulated towards $\mathcal{Y}$ due to their neighbors.

Now, we can use Lemma 78 to bound ρ_i in terms of σ_i:

$$\begin{aligned}\rho_i &\leq \mathbb{P}[\exists 0 < i \leq m : A_i = 0] \\ &= \begin{cases} \mathbb{P}[\exists 0 < i \leq m : B_i = 0]\left(\frac{1-p}{p}\right)^c & \text{if} \quad p \geq \frac{1}{2}, \\ \mathbb{P}[\exists 0 < i \leq m : B_i = 0]\left(\frac{1-p}{p}\right)^{c+k-1} & \text{if} \quad p < \frac{1}{2} \end{cases} \\ &\leq \begin{cases} \sigma_i\left(\frac{1-p}{p}\right)^c & \text{if} \quad p \geq \frac{1}{2}, \\ \sigma_i\left(\frac{1-p}{p}\right)^{c+k-1} & \text{if} \quad p < \frac{1}{2} \end{cases} \end{aligned} \tag{3.5}$$

Next, we bound the expected value of N_π if $p \geq \frac{1}{2}$ with

$$\begin{aligned}\mathbb{E}[N_\pi] &\leq \sum_{i=1}^{n} (\rho_i + (1 - \rho_i - \sigma_i)(1-p)) \\ &\overset{(3.5)}{\leq} \sum_{i=1}^{n} \left(\rho_i + \left(1 - \rho_i - \left(\frac{p}{1-p}\right)^c \rho_i\right)(1-p)\right) \\ &= \sum_{i=1}^{n} \left(1 - p + \left(\frac{(1-p)^{c-1} - p^{c-1}}{(1-p)^{c-1}}\right) p\rho_i\right) \\ &\overset{p \geq 1-p}{\leq} (1-p)n.\end{aligned}$$

On the other hand if $p < \frac{1}{2}$, again using (3.5), yields

$$\begin{aligned}\mathbb{E}[N_\pi] &\leq \sum_{i=1}^{n} (\rho_i + (1 - \rho_i - \sigma_i)(1-p)) \\ &\overset{(3.5)}{\leq} \sum_{i=1}^{n} \left(\rho_i + \left(1 - \rho_i - \left(\frac{p}{1-p}\right)^{c+k-1} \rho_i\right)(1-p)\right) \\ &= \sum_{i=1}^{n} \left(1 - p - (1-p)\rho_i \left(\frac{p}{1-p}\right)^{c+k-1} + p\rho_i\right) \\ &= \sum_{i=1}^{n} \left(1 - p + \left(1 - \underbrace{\frac{p^{c+k-2}}{(1-p)^{c+k-2}}}_{<1}\right) p\rho_i\right).\end{aligned}$$

As this function in ρ_i becomes maximal if ρ_i is chosen to be maximal, i.e. if we let

$\rho_i = \mathbb{P}[\exists i > 0 : A_i^k \leq 0]$, we get

$$\begin{aligned}\mathbb{E}[N_\pi] &\leq n\left(1 - p + \left(1 - \frac{p^{c+k-2}}{(1-p)^{c+k-2}}\right) p\mathbb{P}[\exists i > 0 : A_i^k \leq 0]\right) \\ &= n\left(1 - p\left(1 - \left(1 - \frac{p^{c+k-2}}{(1-p)^{c+k-2}}\right)\mathbb{P}[\exists i > 0 : A_i^k \leq 0]\right)\right).\end{aligned}$$

We finish the proof with using the bound on $\mathbb{P}[\exists i > 0 : A_i^k \leq 0]$ given in Corollary 79 and the fact that $N_\pi + Y_\pi = n$:

$$\begin{aligned}\mathbb{E}[Y_\pi] &\geq np\left(1 - \left(1 - \frac{p^{c+k-2}}{(1-p)^{c+k-2}}\right)\mathbb{P}[\exists i > 0 : A_i^k \leq 0]\right) \\ &\geq np\left(1 - \left(1 - \frac{p^{c+k-2}}{(1-p)^{c+k-2}}\right)\frac{\left(\frac{1-p}{p}\right)^{c+k-1}}{1 + \left(\frac{1-p}{p}\right)^{c+k-1}}\right) \\ &= np\left(1 - \frac{\left(\frac{1-p}{p}\right)^{c+k-1} - \frac{1-p}{p}}{1 + \left(\frac{1-p}{p}\right)^{c+k-1}}\right) \\ &= np \cdot \frac{1 + \left(\frac{1-p}{p}\right)^{c+k-1} - \left(\frac{1-p}{p}\right)^{c+k-1} + \frac{1-p}{p}}{1 + \left(\frac{1-p}{p}\right)^{c+k-1}} \\ &= \frac{n}{1 + \left(\frac{1-p}{p}\right)^{c+k-1}} = \frac{1 - \left(\frac{1-p}{p}\right)^{c+k-1}}{1 - \left(\frac{1-p}{p}\right)^{2(c+k-1)}} \cdot n.\end{aligned}$$

□

Note, one could slightly improve the bounds if one takes into account that ρ_i depends on the vertex's degree.

A Varying Number of Vertices at a Time

We now allow the number of voting vertices to vary each time step. Here, by k we denote the maximal number of vertices that may take a vote simultaneously. Let Y_{π^v} be the number of $\mathcal{Y}$ voting vertices according to a weak order π^v. Observe that the expected number of vertices voting for $\mathcal{Y}$ in an optimal order is monotone in k.

Next, we present a randomized algorithm which is quite similar to Algorithm 7. The algorithm presented below computes an adaptive weak order of width k for a given parameter k. However, we cannot prove (better) performance guarantees, but we included this algorithm in our experimental results in Section 3.3.6.

Algorithm 8: Variable k-block Adaptive Weak Order

Parameter: c
Input: $G = (V, E)$
Output: Weak order π of V

Let $Q_1 = \emptyset$, $Q_2 = V(G)$ and $P = \emptyset$;
while *there exits* $v \in Q_1 \cup Q_2 \cup P$ **do**
  if $P = \emptyset$ **then**
    if $Q_1 = \emptyset$ **then**
      Let $S = \{v\}$ for an arbitrary $v \in Q_2$
    else
      let $S = Q_1 \backslash Q_2$
    end
  else
    let $S = P$
  end
  if $|S| > k$ **then**
    Remove arbitrary vertices from S such that $|S| = k$
  end
  Append S to π;
  Compute the set of passive vertices P, with $m_{\mathcal{Y}}(v) - m_{\mathcal{N}}(v) \geq c$;
  Compute the set of active vertices Q_1, with $-c + p \cdot 2c \geq m_{\mathcal{Y}}(v) - m_{\mathcal{N}}(v) \geq -c$;
  Compute the set of active vertices Q_2, with $c > m_{\mathcal{Y}}(v) - m_{\mathcal{N}}(v) > -c + p \cdot 2c$;
end
Append the remaining vertices to π arbitrarily;

Compared to the other algorithms, Algorithm 8 prioritizes vertices that are about to be manipulated towards $\mathcal{N}$. Intuitively this is what should be done. If there are several vertices which tend to be influenced to $\mathcal{N}$, the algorithm calls as many as possible to vote simultaneously. But if there is no such vertex, it only asks for one active vertex to vote at a time. The concrete rule, which vertices belong to A_1 or to A_2, is arbitrary but performed best in experimental testing.

3.3.6 Computational Results

As opinion formation plays an important role especially in social networks we give a short survey on the quality of our algorithms on inhomogeneous random graphs, where the degree of the vertices follows a power-law distribution. If the parameter of the power-law distribution is β and $2 < \beta < 3$ those graphs model several kinds of real networks appropriately. See for example results by Réka and Barabási in [2]. Hence,

we focus on that specific regime for β in our experiments. The general inhomogeneous random graph model was introduced by Söderberg in [81], but we consider a special case of these graphs, which was studied first by Chung and Lu in [26]. For a graph $G = G(w)$ on n vertices we are given a weight sequence $w(n) = (w_1, \dots, w_n)$ where w_i is a random variable that follows a power-law with exponent $1 < \beta$ for all $i \in [n]$, that is, given some minimal weight x_0, $\mathbb{P}[w_i > x] = \gamma x^{1-\beta}$ for some normalizing constant γ for all vertices independently. The probability of two vertices i and j being adjacent is, again independently of all other vertices, $p_{ij} = \min\{\frac{w_i w_j}{W_{[n]}}, 1\}$, where $W_{[n]} := \sum_{k\in[n]} w_k$. One can think of the weight w_i for some vertex $i \in [n]$ as the expected degree, given we drop the minimum in the calculation of the probabilities and allow loops in the graph, since

$$\mathbb{E}[\deg(i)|w_i] \leq \sum_{j\in[n]} \frac{w_i w_j}{W_{[n]}} = w_i \sum_{j\in[n]} \frac{w_j}{W_{[n]}} = w_i.$$

Lemma 80. *In any inhomogeneous random graph with power-law distributed weights with parameter $1 < \beta$ and $x_0 < c$, any order π of the vertices results in $\mathbb{E}[Y_\pi] \geq \varepsilon pn$ for some fixed $\varepsilon > 0$.*

Proof. This is simply due to the fact that there is a constant fraction of vertices that have degree less than c, and hence decide independently of all other vertices.
The probability that a vertex v with weight w_v has degree less than c is bounded by

$$\mathbb{P}[\deg(v) \geq c \mid w_v] \leq \frac{\mathbb{E}[\deg(v) \mid w_v]}{c} \leq \frac{w_v}{c}, \tag{3.6}$$

which follows from Markov's inequality. Observe, this is bounded by a constant less than 1 if e.g. $w_v < \frac{c+x_0}{2}$. Hence, the expected number of vertices having degree less than c is bounded by

$$\mathbb{E}[\sum_{i\in[n]} \mathbb{1}_{\deg(i)<c}] = \sum_{i\in[n]} \mathbb{E}[\mathbb{1}_{\deg(i)<c}] \overset{(3.6)}{\geq} \sum_{i\in[n]} \left(1 - \frac{c+x_0}{2c}\right) \mathbb{P}\left[w_i \leq \frac{c+x_0}{2}\right] \tag{3.7}$$

$$= \left(1 - \frac{c+x_0}{2c}\right)\left(1 - \gamma\left(\frac{c+x_0}{2}\right)^{1-\beta}\right) n =: \varepsilon n. \tag{3.8}$$

This gives $\mathbb{E}[Y_\pi] \geq \varepsilon pn$. □

Next, we test run our presented algorithms on several example graphs. We also compare the given algorithms to random orders. Additionally, we plot the fraction of vertices voting for $\mathcal{Y}$ under the deterministic bootstrap percolation process, which yields as an upper bound for an optimal (also adaptive) order[8].

[8] As mentioned above, bootstrap percolation is an optimal strategy and corresponds to complete information of initial opinions.

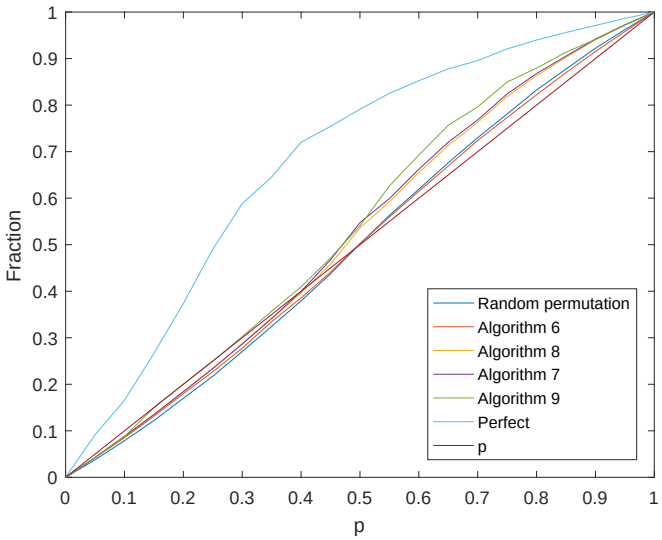


Figure 3.14: Comparison of Algorithms 6-9 on inhomogeneous random graphs with power-law distributed weights, where $n = 3000$, $\beta = 2.3$, $x_0 = 6$, $c = 5$ and $k = 50$.

First of all, we consider the algorithms on inhomogeneous random graphs where the weights follow a power-law distribution. We choose $\beta = 2.3$, $x_0 = 6$ and $n = 3000$. For each value of

$$p \in \{q \in [0,1] : q = i \cdot 0.05 \text{ for } i \in [20] \cup \{0\}\} =: M$$

we computed 500 sample graphs and executed all presented algorithms on them[9]. In Figure 3.14, we plot the average fraction of $\mathcal{Y}$ voting vertices as a function of p after processing the algorithms. It is not surprising that Algorithm 5 performs worse than a random order of vertices for large values of p in our computational experiments. Also, there seems to be no big difference in numbers between the produced orders of Algorithm 6 and Algorithm 7, while Algorithm 8 performed best on almost all instances.

The plot only slightly changes when altering the parameters of the graphs as well as the process threshold c. For increasing c the curves converge to the bisection p, while for decreasing c the functions become bulgier.

Next, we consider the algorithms on the random graph $G_{n,q}$ where $n = 3000$ and $q = 0.01$. Just like before, for each value of $p \in M$ we compute 500 sample graphs and

[9]For each of the values of $p \in M$ we used the same initial opinions for the respective graphs for each of the four algorithms.

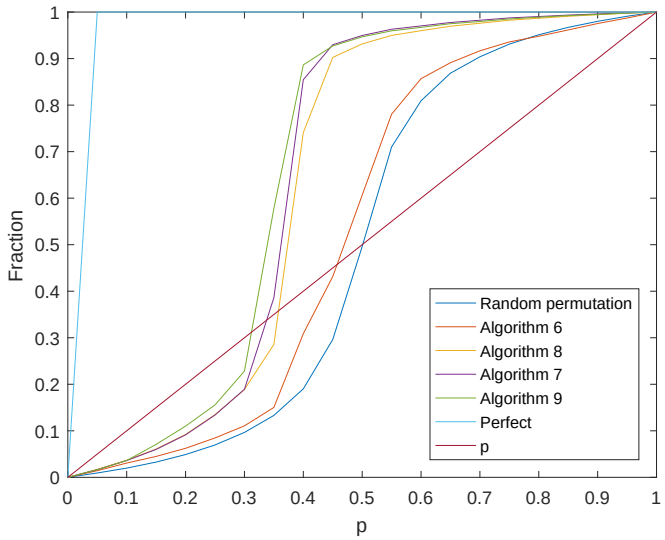


Figure 3.15: Comparison of Algorithms 6-9 on $G(n, q)$ with $n = 3000$, $q = 0.01$, $c = 5$ and $k = 50$.

execute all four algorithms on them. Figure 3.15 shows the fraction of $\mathcal{Y}$ voting vertices as a function of p. Again, we have that Algorithm 8 performs best. Additionally, just like before, Algorithm 5 is worse than a random order for large p. Also, note that there naturally is a large gap between bootstrap percolation — representing an optimum order with perfect information — and our algorithms.

3.3.7 Open Problems

We gave an adaptive algorithm (Algorithm 8) computing variable k-block weak order not making (much) use of the underlying graph structure. In particular, since the adaptive setting allowing for varying number of votes each round gives more flexibility for increasing k we expect better approximation guarantees. For our worst case examples, i.e. cliques, there cannot be any improvement. However, for general graphs, one should be able to exploit the graph structure and derive better approximation algorithms in terms of graph parameters, e.g. the independence number. But also particularly, with applications in social networks in mind, exploiting special structural properties of relevant graph classes such as inhomogeneous random graphs in the algorithms could be a starting point for further research and yield better performance guarantees.

Chapter 4

Conclusion

In the previous chapter of the thesis, we studied processes on graphs partly motivated from physics like ferromagnetic processes at zero temperature. All studied processes share the property that update rules only depend on the current states of the vertex itself and its neighboring vertices. We derived improved bounds on sizes of important sets and resolved some open questions. More precisely, we studied sets of vertices that infect the entire vertex set on degenerate graphs and grids in bootstrap percolation. Potential arguments as well as recursive constructions were used to prove our bounds.

Furthermore, even though the second infection process lacks monotonicity that is crucial in most analyses of such processes, we were able to prove existence of a phase transition on an infinite graph. Key for the analysis was a composed auxiliary Markov chain that allowed us to derive different drift properties for different values of the process parameter, finally proving a phase transition.

Moreover, we analyzed a third process resembling a voting procedure on two options. For this process, we gave arguably simpler algorithms computing an order of votes with better performance guarantees over existing algorithms from the literature. The analysis of the adaptive algorithm relied on modeling the problem as a generalization of the Gambler's ruin game, and deriving new bounds through dominance arguments.

Motivated from the analysis of these processes, we also studied new concepts in algorithmic graph theory such as new classes of cut-degree problems, both in a symmetric as well as an asymmetric version (including a new notion of graph degeneracy). These concepts proved to be interesting themselves and rich in applications, e.g. in data analysis (clustering) or in game theory (anti-coordination games). We derived APX-hardness results for all *NP*-hard problems and gave polynomial time algorithms for the remaining ones. For the Max-Min-Cut-Degree Problem we actually proved that an existing approximation algorithm for a related problem already has a best possible approximation ratio. Additionally to cut-degree problems, we introduced and analyzed a new class of cut-related optimization problems with updated cost on edges in the cut set. We gave polynomial time algorithms for several special instances when the minimization objective is the sum of path lengths.

Finally, in several places of the analysis of computational complexity of some of the cut problems, we came across disjoint path problems. We resolved an open question on the complexity of 2 disjoint shortest paths with non-negative edge weights and gave a polynomial time algorithm for undirected graphs. This algorithm transfers the input to an instance of the k disjoint paths problem on weakly acyclic graphs for constant k, for which we gave a polynomial time algorithm in the preceding subsection.

At the end of each section, we state open problems that could be the starting points for future research direction. In particular, see Section 2.1.6 for several open questions regarding disjoint paths, Section 2.2.6 for cut-degree related problems, Section 2.3.4 for open questions on the complexity of restricted instances, Section 3.1.4 for a conjecture on the maximum size of minimal percolating sets in bootstrap percolation, Section 3.2.6 for a conjecture of the three-state contact process, and Section 3.3.7 for suggestions on further research directions for the opinion formation process.

Additionally, the reader is pointed to the publications and preprints [50, 46, 47] that did not fit the scope of this thesis.

Bibliography

[1] Susanne Albers and Marinus Gottschau, *Opinion formation*, Working paper, 2017.

[2] R. Albert and A. Barabási, *Statistical mechanics of complex networks*, Reviews of Modern Physics **74** (2002), no. 1, 47.

[3] N. Anari, S. Ehsani, M. Ghodsi, N. Haghpanah, N. Immorlica, H. Mahini, and V. S. Mirrokni, *Equilibrium pricing with positive externalities (extended abstract)*, pp. 424–431, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[4] S. Arora, S. Rao, and U. Vazirani, *Expander flows, geometric embeddings and graph partitioning*, Journal of the ACM **56** (2009), no. 2, 1–37.

[5] W. B. Arthur, *Competing technologies, increasing returns, and lock-in by historical events*, The Economic Journal **99** (1989), no. 394, 116–131.

[6] J. Balogh, B. Bollobás, H. Duminil-Copin, and R. Morris, *The sharp threshold for bootstrap percolation in all dimensions*, Transactions of the American Mathematical Society **364** (2012), no. 5, 2667–2701.

[7] J. Balogh, Y. Peres, and G. Pete, *Bootstrap percolation on infinite trees and non-amenable groups*, Combinatorics, Probability and Computing **15** (2006), no. 5, 715–730.

[8] J. Bang-Jensen and S. Bessy, *Degree-constrained 2-partitions of graphs*, Theoretical Computer Science **776** (2019), 64–74.

[9] C. Bazgan, Z. Tuza, and D. Vanderpooten, *Complexity and approximation of satisfactory partition problems*, Computing and Combinatorics (Berlin, Heidelberg) (Lusheng Wang, ed.), Springer Berlin Heidelberg, 2005, pp. 829–838.

[10] J. E. Beasley and N. Christofides, *An algorithm for the resource constrained shortest path problem*, Networks **19** (1989), no. 4, 379–394.

[11] R. Bellman, *On a routing problem*, Quarterly of Applied Mathematics **16** (1958), no. 1, 87–90.

[12] F. Benevides and M. Przykucki, *Maximum percolation time in two-dimensional bootstrap percolation*, SIAM Journal on Discrete Mathematics **29** (2015), no. 1, 224–251.

[13] K. Bérczi and Y. Kobayashi, *The directed disjoint shortest paths problem*, LIPIcs-Leibniz International Proceedings in Informatics, vol. 87, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[14] J. van den Berg, J. E. Björnberg, and M. Heydenreich, *Sharpness versus robustness of the percolation transition in 2d contact processes.*, Stochastic Processes and their Applications **125** (2015), no. 2, 513–537 (English).

[15] A. Björklund and T. Husfeldt, *Shortest two disjoint paths in polynomial time*, Automata, Languages, and Programming (Berlin, Heidelberg) (Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, eds.), Springer Berlin Heidelberg, 2014, pp. 211–222.

[16] O. Blondel, N. Cancrini, F. Martinelli, C. Roberto, and C. Toninelli, *Fredrickson-Andersen one spin facilitated model out of equilibrium*, Markov Processes and Related Fields **19** (2013), no. 3, 383–406. MR 3156958

[17] B. Bollobás, *Modern graph theory*, vol. 184, Springer Science & Business Media, 2013.

[18] B. Bollobás, K. Gunderson, C. Holmgren, S. Janson, and M. Przykucki, *Bootstrap percolation on Galton-Watson trees*, Electronic Journal of Probability **19** (2014).

[19] B. Bollobás, C. Holmgren, P. Smith, and A. J. Uzzell, *The time of bootstrap percolation with dense initial sets*, The Annals of Probability **42** (2014), no. 4, 1337–1373.

[20] M. Bradonjić and I. Saniee, *Bootstrap percolation on periodic trees*, 2015 Proceedings of the Twelfth Workshop on Analytic Algorithmics and Combinatorics (ANALCO), SIAM, 2014, pp. 89–96.

[21] E. I. Broman, *Stochastic domination for a hidden Markov chain with applications to the contact process in a randomly evolving environment*, The Annals of Probability **35** (2007), no. 6, 2263–2293. MR 2353388 (2009a:60118)

[22] D. Brown and N. Hayes, *Influencer marketing: Who really influences your customers?*, Elsevier/Butterworth-Heinemann, 2008.

[23] J. Chalupa, P. L. Leath, and G. R. Reich, *Bootstrap percolation on a Bethe lattice*, Journal of Physics C: Solid State Physics **12** (1979), no. 1, L31.

[24] M. Charikar, N. Gupta, and R. Schwartz, *Local guarantees in graph cuts and clustering*, Integer Programming and Combinatorial Optimization (Cham) (Friedrich Eisenbrand and Jochen Koenemann, eds.), Springer International Publishing, 2017, pp. 136–147.

[25] F. Chierichetti, J. Kleinberg, and A. Panconesi, *How to schedule a cascade in an arbitrary graph*, SIAM Journal on Computing **43** (2014), no. 6, 1906–1920.

[26] F. Chung and L. Lu, *The average distance in a random graph with given expected degrees*, Internet Mathematics **1** (2003), no. 1, 91–113.

[27] L. J. Cowen, R. H. Cowen, and D. R. Woodall, *Defective colorings of graphs in surfaces: Partitions into subgraphs of bounded valency*, Journal of Graph Theory **10** (1986), no. 2, 187–195.

[28] J. T. Cox and R. B Schinazi, *Survival and coexistence for a multitype contact process*, The Annals of Probability **37** (2009), no. 3, 853–876.

[29] R. Durrett, *Lecture notes on particle systems and percolation*, The Wadsworth & Brooks/Cole Statistics/Probability Series, Wadsworth & Brooks/Cole Advanced Books & Software, Pacific Grove, CA, 1988. MR 940469

[30] R. Durrett and C. Neuhauser, *Coexistence results for some competition models*, The Annals of Applied Probability **7** (1997), no. 1, 10–45. MR 1428748 (98g:60178)

[31] R. Durrett and G. Swindle, *Are there bushes in a forest?*, Stochastic Processes and their Applications **37** (1991), no. 1, 19–31.

[32] J. Edmonds, *Paths, trees, and flowers*, Canadian Journal of Mathematics **17** (1965), 449–467.

[33] J. Edmonds and R. M. Karp, *Theoretical improvements in algorithmic efficiency for network flow problems*, Journal of the ACM **19** (1972), no. 2, 248–264.

[34] T. Eilam-Tzoreff, *The disjoint shortest paths problem*, Discrete Applied Mathematics **85** (1998), no. 2, 113–138.

[35] P. Elias, A. Feinstein, and C. Shannon, *A note on the maximum flow through a network*, IRE Transactions on Information Theory **2** (1956), no. 4, 117–119.

[36] S. Even, A. Itai, and A. Shamir, *On the complexity of timetable and multicommodity flow problems*, SIAM Journal on Computing **5** (1976), no. 4, 691–703.

[37] G. Flake, R. Tarjan, and K. Tsioutsiouliklis, *Graph clustering and minimum cut trees*, Internet Mathematics **1** (2003), 385–408.

[38] L. R. Ford and D. R. Fulkerson, *Maximal flow through a network*, Canadian Journal of Mathematics **8** (1956), 399–404.

[39] S. Fortune, J. Hopcroft, and J. Wyllie, *The directed subgraph homeomorphism problem*, Theoretical Computer Science **10** (1980), no. 2, 111–121.

[40] G. H. Fredrickson and H. C. Andersen, *Kinetic Ising model of the glass transition*, Physical Review Letters **53** (1984), 1244–1247.

[41] ———, *Facilitated kinetic Ising models and the glass transition*, The Journal of Chemical Physics **83** (1985), no. 11, 5822–5831.

[42] M. R. Garey and D. S. Johnson, *Computers and intractability: A guide to the theory of np-completeness*, 1979.

[43] M. U. Gerber and D. Kobler, *Classes of graphs that can be partitioned to satisfy all their vertices*, Australasian Journal of Combinatorics **29** (2004), 201–214.

[44] A. V. Goldberg and R. E. Tarjan, *Finding minimum-cost circulations by canceling negative cycles*, Journal of the ACM **36** (1989), no. 4, 873–886.

[45] M. Gottschau, *Bootstrap percolation on degenerate graphs*, Operations Research Proceedings 2017 (Cham) (Natalia Kliewer, Jan Fabian Ehmke, and Ralf Borndörfer, eds.), Springer International Publishing, 2018, pp. 303–308.

[46] M. Gottschau, F. Happach, M. Kaiser, and C. Waldmann, *Budget minimization with precedence constraints*, arXiv preprint arXiv:1905.13740 (2019).

[47] M. Gottschau, H. Haverkort, and K. Matzke, *Reptilings and space-filling curves for acute triangles*, Discrete & Computational Geometry **60** (2018), no. 1, 170–199.

[48] M. Gottschau, M. Heydenreich, K. Matzke, and C. Toninelli, *Phase transition for a non-attractive infection process in heterogeneous environment*, Markov Processes and Related Fields **24** (2018), no. 1, 39–56.

[49] M. Gottschau, M. Kaiser, and C. Waldmann, *The undirected two disjoint shortest paths problem*, Operations Research Letters **47** (2019), no. 1, 70–75.

[50] M. Gottschau and M. Leichter, *The minimum hitting set of bundles problem: Computational complexity and approximability*, under review (2019).

[51] M. Gottschau and J. Matuschke, *Mitigating the negative impact of security inspections on regular journeyers*, Working paper, 2020.

[52] M. Hajiaghayi, H. Mahini, and D. Malec, *The polarizing effect of network influences*, Proceedings of the Fifteenth ACM Conference on Economics and Computation (New York, NY, USA), EC '14, ACM, 2014, pp. 131–148.

[53] M. Hajiaghayi, H. Mahini, and A. Sawant, *Scheduling a cascade with opposing influences*, International Symposium on Algorithmic Game Theory, Springer, 2013, pp. 195–206.

[54] G. Y. Handler and I. Zang, *A dual algorithm for the constrained shortest path problem*, Networks **10** (1980), no. 4, 293–309.

[55] T. E. Harris, *Contact interactions on a lattice*, The Annals of Probability **2** (1974), 969–988. MR 0356292

[56] A. E. Holroyd, *Sharp metastability threshold for two-dimensional bootstrap percolation*, Probability Theory and Related Fields **125** (2003), no. 2, 195–224.

[57] S. Janson, T. Łuczak, T. Turova, and T. Vallier, *Bootstrap percolation on the random graph $G_{n,p}$*, The Annals of Applied Probability **22** (2012), no. 5, 1989–2047.

[58] R. M. Karp, *On the computational complexity of combinatorial problems*, Networks **5** (1975), no. 1, 45–68.

[59] D. Kempe, J. Kleinberg, and É. Tardos, *Maximizing the spread of influence through a social network*, Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003, pp. 137–146.

[60] S. Khot, G. Kindler, E. Mossel, and R. O'Donnell, *Optimal inapproximability results for max-cut and other 2-variable csps?*, SIAM Journal on Computing **37** (2007), no. 1, 319–357.

[61] Y. Kobayashi and R. Sako, *Two disjoint shortest paths problem with non-negative edge length*, Operations Research Letters **47** (2019), no. 1, 66–69.

[62] N. Konno, R. B. Schinazi, and H. Tanemura, *Coexistence results for a spatial stochastic epidemic model*, Markov Processes and Related Fields **10** (2004), no. 2, 367–376. MR 2082579 (2005h:60298)

[63] R. Krishnamurti and D. R. Gaur, *LP rounding and extensions*, Handbook of Approximation Algorithms and Metaheuristics (Teofilo F. Gonzalez, ed.), Chapman and Hall/CRC, 2007.

[64] J. Kun, B. Powers, and L. Reyzin, *Anti-coordination games and stable graph colorings*, Algorithmic Game Theory (Berlin, Heidelberg) (Berthold Vöcking, ed.), Springer Berlin Heidelberg, 2013, pp. 122–133.

[65] A. D. Levin, M. J. Łuczak, and Y. Peres, *Glauber dynamics for the mean-field ising model: cut-off, critical power law, and metastability*, Probability Theory and Related Fields **146** (2010), no. 1-2, 223–265.

[66] T. M. Liggett, *Interacting particle systems*, Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], vol. 276, Springer-Verlag, New York, 1985. MR 776231

[67] ———, *Stochastic interacting systems: contact, voter and exclusion processes*, Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], vol. 324, Springer-Verlag, Berlin, 1999. MR 1717346

[68] D. W. Matula and L. L. Beck, *Smallest-last ordering and clustering and graph coloring algorithms*, Journal of the ACM **30** (1983), no. 3, 417–427.

[69] K. Mehlhorn and M. Ziegelmann, *Resource constrained shortest paths*, Algorithms - ESA 2000 (Berlin, Heidelberg) (M. S. Paterson, ed.), Springer Berlin Heidelberg, 2000, pp. 326–337.

[70] K. Menger, *Zur allgemeinen kurventheorie*, Fundamenta Mathematicae **10** (1927), no. 1, 96–115.

[71] R. Morris, *Minimal percolating sets in bootstrap percolation*, Electronic Journal of Combinatorics **16** (2009), no. 1 R, 1–20.

[72] C. Neuhauser, *Ergodic theorems for the multitype contact process*, Probability Theory and Related Fields **91** (1992), no. 3-4, 467–506. MR 1151806 (93c:60162)

[73] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: algorithms and complexity*, Courier Corporation, 1998.

[74] C. A. Phillips, *The network inhibition problem*, Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing (New York, NY, USA), STOC '93, Association for Computing Machinery, 1993, pp. 776–785.

[75] M. Przykucki, *Maximal percolation time in hypercubes under 2-bootstrap percolation*, The Electronic Journal of Combinatorics **19** (2012), 1–13.

[76] D. Remenik, *The contact process in a dynamic random environment*, The Annals of Applied Probability **18** (2008), no. 6, 2392–2420. MR 2474541 (2010d:60224)

[77] E. Riedl, *Largest minimal percolating sets in hypercubes under 2-bootstrap percolation*, The Electronic Journal of Combinatorics **17** (2010), no. "R80", 1–13.

[78] ———, *Largest and smallest minimal percolating sets in trees*, The Electronic Journal of Combinatorics **19** (2012), no. "P64", 1–18.

[79] N. Robertson and P. D. Seymour, *Graph minors .xiii. the disjoint paths problem*, Journal of Combinatorial Theory, Series B **63** (1995), no. 1, 65–110.

[80] J. M. S. Simões-Pereira, *Joins of n-degenerate graphs and uniquely (m, n)-partitionable graphs*, Journal of Combinatorial Theory, Series B **21** (1976), no. 1, 21–29.

[81] B. Söderberg, *General formalism for inhomogeneous random graphs*, Phyical Review E **66** (2002), 066121.

[82] D. Stirzaker and D. Grimmett, *Probability and random processes*, Clarendon Press, 1992.

[83] J. W. Suurballe, *Disjoint paths in a network*, Networks **4** (1974), no. 2, 125–145.

[84] C. Zhang and H. Nagamochi, *The next-to-shortest path in undirected graphs with nonnegative weights*, Proceedings of the Eighteenth Computing: The Australasian Theory Symposium - Volume 128 (Darlinghurst, Australia, Australia), CATS '12, Australian Computer Society, Inc., 2012, pp. 13–20.