Technische Universität Berlin
Electrical Engineering and Computer Science
Institute of Software Engineering and Theoretical Computer Science
Algorithmics and Computational Complexity (AKT)

Algorithmic Complexity of Successive Evacuation in Decaying Temporal Graphs

Louisa Nau

Thesis submitted in fulfillment of the requirements for the degree
"Bachelor of Science" (B. Sc.) in the field of Computer Science

12 2021

Supervisor and first reviewer: Prof. Dr. Rolf Niedermeier
Second reviewer: Dr. Kitty Meeks, University of Glasgow
Co-Supervisors: Dr. Till Fluschnik

I hereby declare that the thesis submitted is my own, unaided work, completed without any unpermitted external help. Only the sources and resources listed were used.

The independent and unaided completion of the thesis is affirmed by affidavit:

Berlin, the _____ _____

Date Signature

Zusammenfassung

In dieser Arbeit behandeln wir die parametrisierte Komplexität von Successive Evacuation auf decaying (temporalen) Graphen. Dieses Problem beschäftigt sich mit dem Finden eines Pfades zwischen zwei Endpunkten zu jedem Zeitschritt, mit der Einschränkung, dass die Anzahl der Knoten in den Pfaden nicht größer als eine Zahl k ist und die Pfade in zwei benachbarten Zeitschritten maximal d gemeinsame Knoten enthalten dürfen. Ein decaying Graph ist ein temporaler Graph mit der Bedingung, dass in jedem Zeitschritt die Kantenmenge eine Teilmenge der Kantenmenge des vorherigen Zeitschritts ist. Wir zeigen, dass sowohl Successive Evacuation, als auch Planar Successive Evacuation (planarer Ursprungsgraph) und Shortest Successive Evacuation (kürzeste Pfade) NP-schwer sind. Des Weiteren beweisen wir, dass Successive Evacuation für den Parameter k fixed-parameter tractable ist und Shortest Successive Evacuation mit $\tau = 2$ und für die Parameter d und ζ in XP ist. Wobei ζ die Kantendistanz zwischen zwei benachbarten "layern" ist. Zuletzt beweisen wir, dass Planar Successive Evacuation (und somit auch Successive Evacuation) keinen Problemkern der Größe polynomiell in $k + d$ annimmt und Shortest Successive Evacuation keinen Problemkern der Größe polynomiell in $d + \zeta$ annimmt (außer NP $\subseteq$ coNP/poly). Alle drei Probleme nehmen jedoch auf "echten" decaying Graphen einen Problemkern polynomiell in n an. Wobei "echte" decaying Graphen diejenigen sind, deren Kantenmenge in jedem Zeitschritt eine echte Teilmenge der Kantenmenge des vorherigen Zeitschritts sind.

Abstract

In this thesis, we study the parameterized complexity of Successive Evacuation in decaying temporal graphs. Successive Evacuation searches for paths between two vertices in each time step, where the length of each path is at most k and the vertex overlap between two paths of neighboring time steps is of size at most d. A decaying temporal graph is a temporal graph where the edge set of each time step has to be a subset of the edge set of the previous time step. We prove that Successive Evacuation, as well as the planar and shortest path version, are NP-hard. We further show that both length-bounded versions of Successive Evacuation are fixed-parameter tractable when parameterized by k and that Shortest Successive Evacuation is in XP when $\tau = 2$ when parameterized by ζ and d. Lastly, we look at the problem kernelization. Planar Successive Evacuation, and consequently Successive Evacuation, does not admit a problem kernel of size polynomial in $k + d$ and Shortest Successive Evacuation does not admit problem kernel of size polynomial in $d + \zeta$ (unless NP $\subseteq$ coNP/poly). However, all three problems admit a problem kernel polynomial in n on proper decaying graphs. In a proper decaying graph, the edge sets are proper subsets of the edge set in the previous time step.

Contents

Chapter 1

Introduction

Climate change is increasingly influencing the earth and thus resulting in a rising probability of the occurrence of natural disasters, such as forest fires, storms, floods, etc. [BKD14]. With these disasters affecting us, our cities and living situations we have to find proper solutions to deal with those situations. One important challenge is to create appropriate evacuation plans to save large parts of the population [GR09]. Overall, we need to make sure that while we are evacuating, not too many roads are occupied at similar times since overcrowding of roads can lead to blockage and therefore result in an overall unsafe evacuation. Furthermore, the duration of the evacuation should be preferably short.

To trying to solve this evacuation problem, especially in cases of flooding, we can attempt to remodel it using the problem Successive Evacuation we introduce below. We work on temporal graphs with the nodes representing certain locations and the edges representing the connecting roads. A temporal graph $G = (V, E_1, \dots, E_\tau)$ contains a fixed node set V as well as multiple edge sets $E_1, \dots, E_\tau$ representing the edge relations for each time step $1, \dots, \tau$. In the case of flooding, we can assume that the water is likely to rise over time. While increasingly more roads in the network will be inaccessible, it is most likely that most of these roads remain inaccessible the whole time. Thus, we limit the temporal graph to be a *decaying* (temporal) graph, meaning that $E_i \supseteq E_{i+1}$ for every $i \in \{1, \dots, \tau - 1\}$. The start and end nodes for each time step can vary as, in evacuation scenarios, residents from various areas may be transported to various safe places. The length and similarity between neighboring paths are important when considering the overall safety of an evacuation. Thus we try to limit the path length to a maximum of k and the maximum overlap of two neighboring paths to d. Formally, we define the Successive Evacuation problem as follows[1]:

Problem 1: Successive Evacuation

Input: A decaying (temporal) graph $G = (V, E_1, \dots, E_\tau)$, τ origin-destination pairs $(a_1, b_i), \dots, (a_\tau, b_\tau)$, and two integers $k, d \in \mathbb{N}$.

Question: Is there an a_i-b_i path P_i in (V, E_i) with $|V(P_i)| \leq k$ for all $i \in \{1, \dots, \tau\}$ such that for all $i \in \{1, \dots, \tau - 1\}$ it holds that $|V(P_i) \mathbin{\overset{\circ}{\cap}} V(P_{i+1})| \leq d$?

[1] Let P_1 and P_2 be two paths with start and end vertex sets $\{s_1, t_1\}$ and $\{s_2, t_2\}$; then $V(P_1) \mathbin{\overset{\circ}{\cap}} V(P_2) \coloneqq (V(P_1) \cap V(P_2)) \setminus \{s_1, s_2, t_1, t_2\}$.

In this context, we will also look at two variations of this problem that may be easier. First, we introduce the planar version of Successive Evacuation. Not every road network contains bridges or tunnels. In those cases, it would be sufficient to solve Successive Evacuation when the input graph is a planar decaying graph. We will define Planar Successive Evacuation as follows:

Problem 2: Planar Successive Evacuation
Input: A planar decaying (temporal) graph $G = (V, E_1, \dots, E_\tau)$, τ origin-destination pairs $(a_1, b_i), \dots, (a_\tau, b_\tau)$, and two integers $k, d \in \mathbb{N}$.
Question: Is there an a_i-b_i path P_i in (V, E_i) with $|V(P_i)| \leq k$ for all $i \in \{1, \dots, \tau\}$ such that for all $i \in \{1, \dots, \tau - 1\}$ it holds that $|V(P_i) \cap V(P_{i+1})| \leq d$?

It could also be that the goal is to evacuate every person as fast as possible, thus not looking for paths shorter than k but for a shortest possible path. We take this into consideration since for other problems, the shortest path version is often easier than the upper-bounded version. Thus we introduce Shortest Successive Evacuation, a variation of Successive Evacuation, with the path length not upper-bounded by a parameter k but by the shortest possible path in each time step.

Problem 3: Shortest Successive Evacuation
Input: A decaying (temporal) graph $G = (V, E_1, \dots, E_\tau)$, τ origin-destination pairs $(a_1, b_i), \dots, (a_\tau, b_\tau)$, and an integer $d \in \mathbb{N}$.
Question: Is there an a_i-b_i path P_i in (V, E_i) with P_i being a shortest path for all $i \in \{1, \dots, \tau\}$ such that for all $i \in \{1, \dots, \tau - 1\}$ it holds that $|V(P_i) \cap V(P_{i+1})| \leq d$?

1.1 Our contributions

We summarize our results for the three problems we studied in Table 1.1. We observed that all three versions of Successive Evacuation are NP-hard, even when d is constant. Successive Evacuation and Shortest Successive Evacuation are NP-hard, even when τ and d are constant. In contrast, for Planar Successive Evacuation it is still open whether it is in FPT for a constant τ. However, both length-bounded versions, Successive Evacuation and Planar Successive Evacuationare fixed-parameter tractable, when parameterized by the length k. It is also interesting to note that when $\tau = 2$ Shortest Successive Evacuation is in XP parameterized by d and the edge distance ζ.

1.2 Related work

Our problems are related to multistage problems, introduced by Eisenstat, Mathieu, and Schabanel [EMS14] and Gupta, Talwar, and Wieder [GTW14] and are often variations of already studied problems on temporal graphs.

Closest to our problem is the vertex-disjoint variation of Multistage s-t Path as studied by Fluschnik et al. [Flu+20]. This problem mainly differs from Successive Evacuation because it only has one origin-destination pair, uses general temporal graphs, and is not restricted to decaying graphs. In their work, they mainly studied

Table 1.1: Summary of our results: The table contains the three problems we studied and their complexity regarding different parameters. We will shorten para-NP-hard to "p-NP-h" and decaying graph to "DG". We will use "PK" to indicate the existence of a problem kernel for this parameter and "noPK" to state that no problem kernel for this parameter exists unless NP $\subseteq$ coNP/poly. [ζ: for all $\{i \in \{1, \dots, \tau - 1\}$ let $\zeta = \max\{|E_i \setminus E_{i+1}|\}$; Δ: maximum degree $\Delta = \max\{deg(v) \mid v \in V\}$]

Parameter	SE	Planar SE	Shortest SE
k	FPT (Thm. 4.3)	FPT (Thm. 4.3)	n/a
d	p-NP-h even if $d = 0$ (Thm. 3.1)	p-NP-h even if $d = 0$ (Thm. 3.3)	p-NP-h even if $d = 0$ (Thm. 3.4)
τ	p-NP-h even if $\tau = 2$ (Thm. 3.1)	*open*	p-NP-h even if $\tau = 2$ (Thm. 3.4)
ζ	p-NP-h even if $\zeta = 0$ (Thm. 3.2)	p-NP-h even if $\zeta = 0$ (Thm. 3.3)	*open*
Δ	p-NP-h even if $\Delta = 4$ (Thm. 3.1)	p-NP-h even if $\Delta = 4$ (Thm. 3.3)	p-NP-h even if $\Delta = 4$ (Thm. 3.4)
$\zeta + \tau + d$	p-NP-h (Thm. 3.1)	*open*	XP if $\tau = 2$ (Lemma 4.2)
$k + d$	noPK (Thm. 5.1)	noPK (Thm. 5.1)	n/a
$d + \zeta$	noPK (p-NP-h) (Thm. 3.1)	noPK (p-NP-h) (Thm. 3.3)	noPK (Thm. 5.3)
n	PK on proper DG (Thm. 5.5)	PK on proper DG (Thm. 5.5)	PK on proper DG (Thm. 5.5)

the parameterized complexity of the problem and showed that Multistage s-t Path is in FPT for a constant path length. We will use this result later to show that Successive Evacuation is in FPT for a constant path length as well.

Successive Evacuation is similar to various problems regarding disjoint paths. Finding length-bounded or shortest pairwise disjoint paths in graphs has been extensively studied [Ben+20; GT11; HP02; LMSL90; Loc21; TV96]. However, the aspect that the input graph can drop any number of edges at each time step does change the complexity of the problem. For instance, while it has been proven that Shortest Disjoint Paths can be solved in $O(kn^{16k\dot{k}!+k+19})$-time for a constant number of pairwise disjoint paths (Lochet [Loc21], Bentert et al. [Ben+20]), Shortest Successive Evacuation is NP-hard even for $\tau = 2$.

Chapter 2

Preliminaries

Table 2.1: Overview of certain terms and symbols used in this work.

term/symbol	
$\mathbb{N}$	natural numbers containing 0
$\mathbb{N}^+$	natural numbers excluding 0
$V(G)$	vertex set of a graph G
$E(G)$	edge set of a graph G
$E_i(G)$	edge set of a temporal graph G at time step i
$N(v)$	a set of the vertices adjacent to v
$deg(v)$	$deg(v) = \lvert N(v)\rvert$
Δ	maximum degree $\Delta = \max\{deg(v) \mid v \in V\}$
$\overset{\circ}{\cap}$	if $\{s_1, t_1\}$ and $\{s_2, t_2\}$ are the endpoints of two paths P_1 and P_2, then $V(P_1) \overset{\circ}{\cap} V(P_2) = (V(P_1) \cap V(P_2)) \setminus \{s_1, s_2, t_1, t_2\}$
ζ	for all $\{i \in \{1, \dots, \tau - 1\}$ let $\zeta = \max\{\lvert E_i \setminus E_{i+1}\rvert\}$

In Table 2.1 we give an overview of our basic notation used in this work.

2.1 Graph theory

A graph $G = (V, E)$ consists of a vertex set V and a set $E = \{\{v, u\} \mid v, u \in V, v \neq u\}$ of undirected edges. A *path* $P = (V_e, E_e)$ consists of a vertex set V_p containing n vertices, such that the vertices can be arranged in a sequence $v_1, \dots, v_n$ and an edge set E_p so that $E_p = \{\{v_i, v_{i+1}\} \mid i \in \{1, \dots, n-1\}\}$. We call v_1 and v_n the start and end vertices and the path a v_1-v_n path. A path P is a path in G if $P \subseteq G$. We call a path P a simple path if every vertex in $V(P)$ is visited exactly once. We call a graph a *planar* graph if it can be embedded in the plane such that no two edges intersect, except possibly in their endpoints.

2.2 Temporal graph theory

A *temporal* graph $G = (V, E_1, \ldots, E_\tau)$ contains a vertex set V and τ many edge sets $E_1, \ldots, E_\tau$. Additionally, a *decaying* graph is a type of temporal graph fulfilling the condition that for all $i, j \in \{1, \ldots, \tau\}$ $E_i \supseteq E_j$, $i < j$. If $E_i \supset E_j$, $i < j$, we call the graph a *proper decaying* graph.

The graph $G_i = (V, E_i)$ is called the i-th layer. We call a decaying graph a *planar* decaying graph if its first layer can be embedded in the plane, such that none of its edges intersect, except possibly in their endpoints.

2.3 Parameterized complexity

Let Σ be a finite alphabet. A problem is a language $L \subset \Sigma^*$, and a parameterized problem is a problem $L \subset \{(x, k) \in \Sigma^* \times \mathbb{N}\}$ with k being the parameter. We say a problem L is in NP if a polynomial sized solution can be verified in polynomial time and it is NP-hard if it is at least as hard as all other problems in NP. A problem is NP-complete if it is in NP and NP-hard. We call a parameterized problem L para-NP-hard if it is NP-hard for a constant parameter k. A parameterized problem L is fixed-parameter tractable (and in FPT) when for a parameter k there is an algorithm that decides any input instance (x, k) in $f(k)\dot{|}x|^{O(1)}$ time, where f is some computable function. A parameterized problem L is in XP when for a parameter k there is an algorithm that decides any input instance (x, k) in $|x|^{f(k)}$ time, where f is some computable function. A problem kernelization for a parameterized problem L is a polynomial-time algorithm that creates an instance (x', k'), that is equivalent to the input instance (x, k) such that $|x'| + k' \leq f(k)$, where f is any computable function. We call a kernelization polynomial if f is a polynomial function.

Chapter 3

NP-hardness

We prove in this chapter that Successive Evacuation and Shortest Successive Evacuation are both NP-hard. All of our variants of Successive Evacuation are contained in the complexity class NP, since we can verify if a path is shorter than k or a shortest path in polynomial time, NP-completeness follows. We prove NP-hardness by providing polynomial-time many-one reductions from 3-SAT to first Successive Evacuation and then to Shortest Successive Evacuation. In Section 3.1.3, we see that Successive Evacuation is NP-hard even on planar decaying graphs.

3.1 NP-hardness of Successive Evacuation

This section focuses on the NP-hardness of length bounded versions of Successive Evacuation. In Section 3.1.1 we prove that Successive Evacuation is NP-complete for $\tau = 2$, a maximum vertex degree of four and $d = 0$. We expand this result in Section 3.1.2 and show that this is even the case when both layers contain identical edge sets by inserting an extra "chain" of vertices to "replace" one of the outgoing edges of the start vertex. We then prove NP-completeness for Planar Successive Evacuation when $d = 0$ and all edge sets are identical.

3.1.1 Two layers of maximum degree four and $d = 0$

We prove that even for a maximum vertex degree of four and two vertex-disjoint paths, Successive Evacuation is still NP-complete.

Theorem 3.1. Successive Evacuation *is NP-complete even if* $d = 0$ *and* $\Delta = 4$.

We give a polynomial-time many-one reduction from the NP-complete 3-SAT problem (inspired by the proof of the NP-completeness for Multistage s-t Path by Fluschnik et al. [Flu+20]).

Problem 4: 3-SAT
Input: A set of n boolean variables $X = \{x_1, \ldots, x_n\}$ and a set of m clauses $C = \{C_1, \ldots, C_m\}$, each containing exactly 3 literals.
Question: Is there a set of variables $X' \subseteq X$ assigned 1 so that in each clause there is at least one literal assigned 1?

The main idea of this reduction is to construct a decaying graph containing two layers to represent the 3-SAT instance. For each variable, we have two "parallel" "chains" of the length of the maximum appearance of any variable in the clauses. One of those "chains" represents the variable when assigned 1 and the other "chain" the variable when assigned 0. The first path contains at least one of these "chains" for each variable and represents X'. The second path represents the clauses. For each clause, we have two representing vertices that are connected by three individual vertices. Those three vertices do each represent one literal in the clause. By removing the edges between the "chains" for the variables, the second path will not contain two connected vertices representing the same variable.

We assume an equal number of variables and clauses w.l.o.g. in our 3-SAT instances in this reduction, as well as in following reductions. We do this since we can adjust every instance of 3-SAT in polynomial time to fulfill this condition by adding variables and clauses that will admit value 1 so that it will not interfere with the solution for the starting instance.

Construction 1. Let $(X = \{x_1, \dots, x_n\}, C = \{C_1, \dots, C_n\})$ be an instance of 3-SAT where the number of variables equals the number of clauses. Let p be the most frequent appearance of a variable in X. We construct a decaying graph $G = (V, E_1, E_2)$ as follows (see Figure 3.1 for an illustration). Let

$$\begin{aligned} V \coloneqq &\{s, t\} \cup \{c_1^i, \dots, c_{2n}^i \mid i \in \{1, 2\}\} \cup \{a_1^i, \dots, a_p^i \mid i \in \{1, \dots, n\}\} \cup \\ &\{b_1^i, \dots, b_p^i \mid i \in \{1, \dots, n\}\}. \end{aligned}$$

Let

$$E_{i,a} \coloneqq \bigcup_{j \in \{1, \dots, p-1\}} \{\{a_j^i, a_{j+1}^i\}\}$$

and let

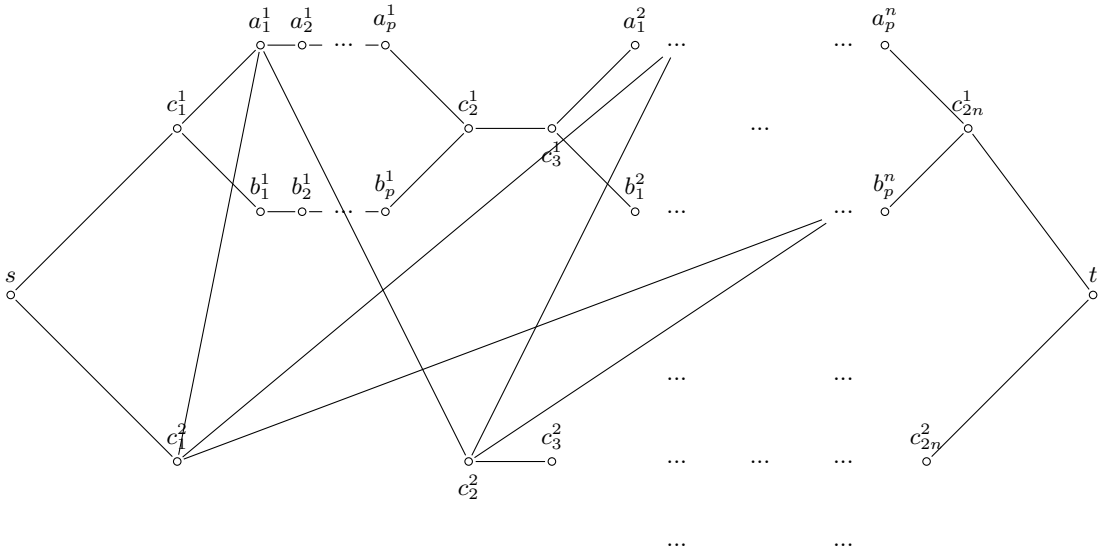
$$E_{i,b} \coloneqq \bigcup_{j \in \{i, \dots, p-1\}} \{\{b_j^i, b_{j+1}^i\}\}.$$

We construct two edge sets $E_{1,1}$ and $E_{1,2}$. Set $E_{1,1}$ contains

- the edges $\{s, c_1^1\}$ and $\{t, c_{2n}^1\}$,
- the edge sets $\bigcup_{i \in \{1, \dots, n\}} \{\{c_{2i-1}^1, a_1^i\}, \{c_{2i-1}^1, b_1^i\}\}$ and $\bigcup_{i \in \{1, \dots, n\}} \{\{c_{2i}^1, a_p^i\}, \{c_{2i}^1, b_p^i\}\}$,
- the edge set $\bigcup_{i \in \{1, \dots, n-1\}} \{\{c_{2i}^1, c_{2i+1}^1\}\}$, and
- the edge sets $\bigcup_{i \in \{1, \dots, n\}} E_{i,a}$ and $\bigcup_{1 \leq i \leq n} E_{i,b}$.

Set $E_{1,2}$ is built in the following way. For each clause $C_q \in C$ we add the following:

- the edges $\{s, c_1^2\}$ and $\{t, c_{2n}^2\}$,
- if C_q contains the j-th appearance of the positive literal x_i add $\{c_{2q-1}^2, a_j^i\}, \{c_{2q}^2, a_j^i\}$,



(a) A snapshot of the first layer of the resulting graphs, $G = (V, E_1)$.

(b) A snapshot of the second layer. $G = (V, E_2)$

Figure 3.1: A decaying graph constructed with Construction 1, exemplified for clause $C_1 = (x_1 \vee x_j \vee \bar{x_i})$, with $i, j \in \{3, \ldots, p\}$ and $i \neq j$.

- if C_q contains the j-th appearance of the negative literal x_i add $\{c_{2q-1}^2, b_j^i\}, \{c_{2q}^2, b_j^i\}$ and
- the edge set $\bigcup_{i \in \{1,\ldots,n-1\}} \{\{c_{2i}^2, c_{2i+1}^2\}\}$.

Let $E_1 \coloneqq E_{1,1} \cup E_{1,2}$ and $E_2 \coloneqq E_{1,2}$. ◊

Proof. Theorem 3.1. Let $I = (X = \{x_1, \ldots, x_n\}, C = \{C_1, \ldots, C_n\})$ be an instance of 3-SAT where the number of variables equals the number of clauses and let $I' = (G = (V, E_1, E_2), (a_1, b_1), (a_2, b_2), k, d)$ be the instance of Successive Evacuation obtained from Construction 1 with $d = 0$, $k = n \cdot p + 2n + 2$, $a_1 = a_2 = s$ and $b_1 = b_2 = t$. We show that I is a YES-instance if and only if I' is a YES-instance.

($\Rightarrow$) Let $X' \subseteq X$ be a solution for I. We construct the paths (P_1, P_2) as follows. For all $i \in \{1, \ldots, n\}$, vertex set $V(P_1)$ contains $\{b_1^i, \ldots, b_p^i\}$ if $x_i \in X'$ and vertex set $V(P_1)$ contains $\{a_1^i, \ldots, a_p^i\}$ if $x_i \notin X'$. Furthermore, $V(P_1)$ contains $\{s, t\} \cup \{c_1^1, \ldots, c_{2n}^1\}$. Let $H = (\bigcup_{1 \leq i \leq n} \{a_1^i, \ldots, a_p^i\} \cup \bigcup_{1 \leq i \leq n} \{b_1^i, \ldots, b_p^i\}) \setminus V(P_1)$ be an auxiliary vertex set. For all

$q \in \{1, \ldots, n\}$, we define $V(C_q) \coloneqq N(c^2_{2q-1}) \cap N(c^2_{2q})$ where $N(v)$ are the neighbor vertices of a vertex v in the second snapshot. Vertex set $V(P_2)$ contains $\{s,t\} \cup \{c^2_1, \ldots, c^2_{2n}\}$ and if $a^i_j \in V(C_q) \cap H$ with i smallest, then $V(P_2)$ contains a^i_j, and if $b^i_j \in V(C_q) \cap H$ with i smallest, then $V(P_2)$ contains b^i_j. We observe, that $|V(P_1)| = n \cdot p + 2n + 2 \leq k$ and $|V(P_2)| = 2n + n + 2 \leq k$. Since $V(P_1) \cap H = \emptyset$ and $(V(P_2) \setminus (\{s,t\} \cup \{c^2_1, \ldots, c^2_{2n}\})) \subseteq H$, $V(P_1) \overset{\circ}{\cap} V(P_2) = \emptyset$ and thus $|V(P_1) \overset{\circ}{\cap} V(P_2)| = 0 \leq d$. Thus (P_1, P_2) is a solution for I'.

($\Leftarrow$) Let (P_1, P_2) be a solution for I'. We construct a solution X' for I as follows. If $V(P_1)$ contains the vertex set $\{b^i_1, \ldots, b^i_p\}$, then $x_i \in X'$, and if $V(P_1)$ contains the vertex set $\{a^i_1, \ldots, a^i_p\}$, then $x_i \notin X'$, for all $i \in \{1, \ldots, n\}$. For each clause $C_q \in C$, $q \in \{1, \ldots, n\}$, there exists a vertex pair $\{c^2_{2q-1}, c^2_{2q}\} \in V(P_2)$ and a vertex $v \in V(P_2)$ that connects this pair. The vertex v is either a^i_j, if x_i is a positive literal in C_q, or b^i_j, if x_i is a negative literal in C_q. Thus, $x_i \in X'$ if $v = a^i_j$, since $V(P_1) \overset{\circ}{\cap} V(P_2) = \emptyset$. Thus $X' \subseteq X$ is a solution for I. □

3.1.2 Two identical layers of maximum degree four and $d = 0$

We prove that Successive Evacuation is NP-complete for two disjoint paths, the maximum underlying vertex-degree being four, $\tau = 2$, and the edge distance $\zeta = 0$.

Theorem 3.2. Successive Evacuation *is NP-complete even for* $\tau = 2$, $d = 0$, $\zeta = 0$ *and* $\Delta G = 4$.

We give a polynomial-time many-one reduction from the NP-complete 3-SAT problem (inspired by the reduction of Li, McCormick, and Simchi-Levi [LMSL90] for showing NP-hardness of Two length-bounded Vertex-Disjoint Paths). This reduction is similar to Construction 1 because both paths are built very similar. However, in Construction 2 both layers of the graph have the same edge sets such that $\zeta = 0$. So to prevent the paths from containing vertices that are not intended, we include an extra chain of vertices between the starting vertex s and the c^2_1. This way, the path using c^2_1 can only use the vertices used in the second path of Construction 1, or else the path would be too long. The other path will be identical to the first path of Construction 1.

Construction 2. Let $(X = \{x_1, \ldots, x_n\}, C = \{C_1, \ldots, C_n\})$ be an instance of 3-SAT where the number of variables equals the number of clauses. Let p be the most frequent appearance of a variable in X. We construct a decaying graph $G = (V, E_1, E_2)$ as follows (see Figure 3.2 for an illustration). Let

$$\begin{aligned} V \coloneqq &\{s,t\} \cup \{c^i_1, \ldots, c^i_{2n} \mid i \in \{1,2\}\} \cup \{a^i_1, \ldots, a^i_p \mid x_i \in X\} \cup \{b^i_1, \ldots, b^i_p \mid x_i \in X\} \cup \\ &\{w_1, \ldots, w_{3n^2(p-1)+pn-n}\} \cup \{a^i_{j,1}, \ldots, a^i_{j,3n} \mid x_i \in X, j \in \{1, \ldots, p-1\}\} \cup \\ &\{b^i_{j,1}, \ldots, b^i_{j,3n} \mid x_i \in X, j \in \{1, \ldots, p-1\}\}. \end{aligned}$$

Let

$$E_{i,a} \coloneqq \bigcup_{j \in \{1,\ldots,p-1\}} \{\{a^i_j, a^i_{j,1}\}, \{a^i_{j+1}, a^i_{j,3n-1}\}\} \cup \bigcup_{g \in \{1,\ldots,3n-1\}} \{\{a^i_{j,g}, a^i_{j,g+1}\}\},$$

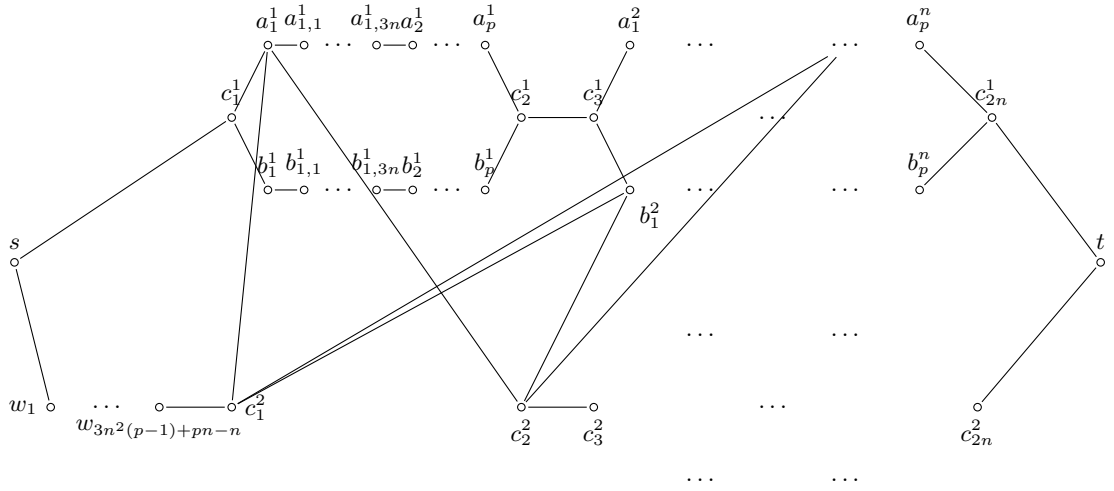


Figure 3.2: Snapshots we obtained from Construction 2, exemplified for clause $C_1 = (x_1 \vee \bar{x_2}) \vee x_i)$, with $i \in \{3, \ldots, p\}$. In this graph, both edge sets are identical. Thus we can represent it as a static graph.

let

$$E_{i,b} \coloneqq \bigcup_{j \in \{1,\ldots,p-1\}} \{\{b_j^i, b_{j,1}^i\}, \{b_{j+1}^i, b_{j,3n-1}^i\}\} \cup \bigcup_{g \in \{1,\ldots,3n\}} \{\{a_{j,g}^i, a_{j,g+1}^i\}\},$$

and let

$$E_w \coloneqq \bigcup_{j \in \{1,\ldots,3n^2(p-1)+pn-n-1\}} \{\{w_j, w_{j+1}\}\}.$$

Then E_1 contains

- the edges $\{s, c_1^1\}$, $\{s, w_1\}$, $\{t, c_{2n}^1\}$ and $\{t, c_{2n}^2\}$,
- the edge $\{w_{3n^2(p-1)+pn-n}, c_1^2\}$,
- the edge sets $\bigcup_{i \in \{1,\ldots,n\}} \{\{c_{2i-1}^1, a_1^i\}, \{c_{2i-1}^1, b_1^i\}\}$ and $\bigcup_{i \in \{1,\ldots,n\}} \{\{c_{2i}^1, a_p^i\}, \{c_{2i}^1, b_p^i\}\}$,
- if C_q contains the j-th appearance of the positive literal x_i add $\{c_{2q-1}^2, a_j^i\}, \{c_{2q}^2, a_j^i\}$,
- if C_q contains the j-th appearance of the negative literal x_i add $\{c_{2q-1}^2, b_j^i\}, \{c_{2q}^2, b_j^i\}$,
- the edge sets $\bigcup_{i \in \{1,\ldots,n-1\}} \{\{c_{2i}^1, c_{2i+1}^1\}\}$ and $\bigcup_{i \in \{1,\ldots,n-1\}} \{\{c_{2i}^2, c_{2i+1}^2\}\}$ and
- the edge sets $\bigcup_{i \in \{1,\ldots,n\}} E_{i,a}$, $\bigcup_{1 \le i \le n} E_{i,b}$, and E_w.

Let $E_2 \coloneqq E_1$. ◊

Proof. Theorem 3.2. Let $I = (X = \{x_1, \ldots, x_n\}, C = \{C_1, \ldots, C_n\})$ be an instance of 3-SAT where the number of variables equals the number of clauses. Let $I' = (G = (V, E_1, E_2), (a_1, b_1), (a_2, b_2), k, d)$ be an instance of Successive Evacuation obtained from Construction 2 with $d = 0$, $k = 3n^2(p-1) + (p+2)n + 2$, $a_1 = a_2 = s$, and $b_1 = b_2 = t$. We show that I is a YES-instance if and only if I' is a YES-instance.

($\Rightarrow$) Let $X' \subseteq X$ be a solution for I. We construct the paths (P_1, P_2) as follows. For all $i \in \{1, \ldots, n\}$, vertex set $V(P_1)$ contains $\{b_1^i, \ldots, b_p^i\}$ if $x_i \in X'$ and $V(P_1)$ contains $\{a_1^i, \ldots, a_p^i\}$ if $x_i \notin X'$. Furthermore, $V(P_1)$ contains $\{s, t\} \cup \{c_1^1, \ldots, c_{2n}^1\}$. Let $H = (\bigcup_{1 \leq i \leq n} \{a_1^i, \ldots, a_p^i\} \cup \bigcup_{1 \leq i \leq n} \{b_1^i, \ldots, b_p^i\}) \setminus V(P_1)$ be an auxiliary vertex set. For each $q \in \{1, \ldots, n\}$, we define $V(C_q) \coloneqq N(c_{2q-1}^2) \cap N(c_{2q})^2$ where $N(v)$ are the neighbors vertices of a vertex v. Vertex set $V(P_2)$ contains $\{s, t\} \cup \{c_1^2, \ldots, c_{2n}^2\} \cup E_w$ and if $a_j^i \in V(C_q) \cap H$ with i smallest, then $V(P_2)$ contains a_j^i, and if $b_j^i \in V(C_q) \cap H$ with i smallest, then $V(P_2)$ contains b_j^i. Since $V(P_1) \cap H = \emptyset$ and $(V(P_2) \setminus (\{s, t\} \cup \{c_1^2, \ldots, c_{2n}^2\})) \subseteq H$, $V(P_1) \overset{\circ}{\cap} V(P_2) = \emptyset$ and thus $|V(P_1) \overset{\circ}{\cap} V(P_2)| = 0 \leq d$. It is easy to see that $|V(P_1)| = n \cdot p + 2n + 2 \leq k$ and $|V(P_2)| = 3n^2(p-1) + (p+2)n + 2 \leq k$. Thus, (P_1, P_2) is a solution for I'.

($\Leftarrow$) Let (P_1, P_2) be a solution for I'. We construct a solution $X' \subseteq X$ for I as follows. If $V(P_1)$ contains the set $\{b_1^i, \ldots, b_p^i\}$, then $x_i \in X'$, and if $V(P_1)$ contains the set $\{a_1^i, \ldots, a_p^i\}$, then $x_i \notin X'$, for all $i \in \{1, \ldots, n\}$. For each clause $C_q \in C$, $q \in \{1, \ldots, n\}$, there exists a vertex pair $\{c_{2q-1}^2, c_{2q}^2\} \in V(P_2)$ and a vertex $v \in V(P_2)$ that connects this pair. The vertex v is either a_j^i if x_i is a positive literal in C_q, or b_j^i if x_i is a negative literal in C_q. Thus, if $v = a_j^i$ then $x_i \in X'$, since $V(P_1) \overset{\circ}{\cap} V(P_2) = \emptyset$. $\square$

3.1.3 On planar decaying graphs

We now prove that NP-hardness Successive Evacuation even holds when the input graph is a planar decaying graph, the maximum vertex degree is four, the edge sets of all layers are identical, and $d = 0$.

Theorem 3.3. Planar Successive Evacuation *is NP-complete even for* $d = 0$, $\zeta = 0$, *and* $\Delta = 4$.

We give a polynomial-time many-one reduction from the NP-complete 3-SAT problem (inspired by the proof for NP-hardness for length-bounded disjoint paths on planar graphs from Holst and Pina [HP02]). The main idea of this reduction is to construct various zigzag paths, two representing each variable and one for each clause.

Construction 3. Let $(X = \{x_1, \ldots, x_n\}, C = \{C_1, \ldots, C_m\})$ be an instance of 3-SAT where n equals the number of variables and m equals the number of clauses. We construct a decaying graph $G = (V, E_1, \ldots, E_{2n+m})$ as follows (see Figure 3.3 for an illustration). Let

$$\begin{aligned} V \coloneqq &\{x_{i,1}^1, \ldots, x_{i,4}^1 \mid i \in \{1, \ldots, n\}\} \cup \{x_{i,1}^2, \ldots, x_{i,4}^2 \mid i \in \{1, \ldots, n\}\} \cup \\ &\{c_1^i, c_2^i, c_3^i \mid i \in \{1, \ldots, m\}\} \cup \\ &\{(h, q) \mid 0 \leq h \leq 4n, 0 \leq q \leq 4n + 2m, (h + q) \mod 2 = 1\}. \end{aligned}$$

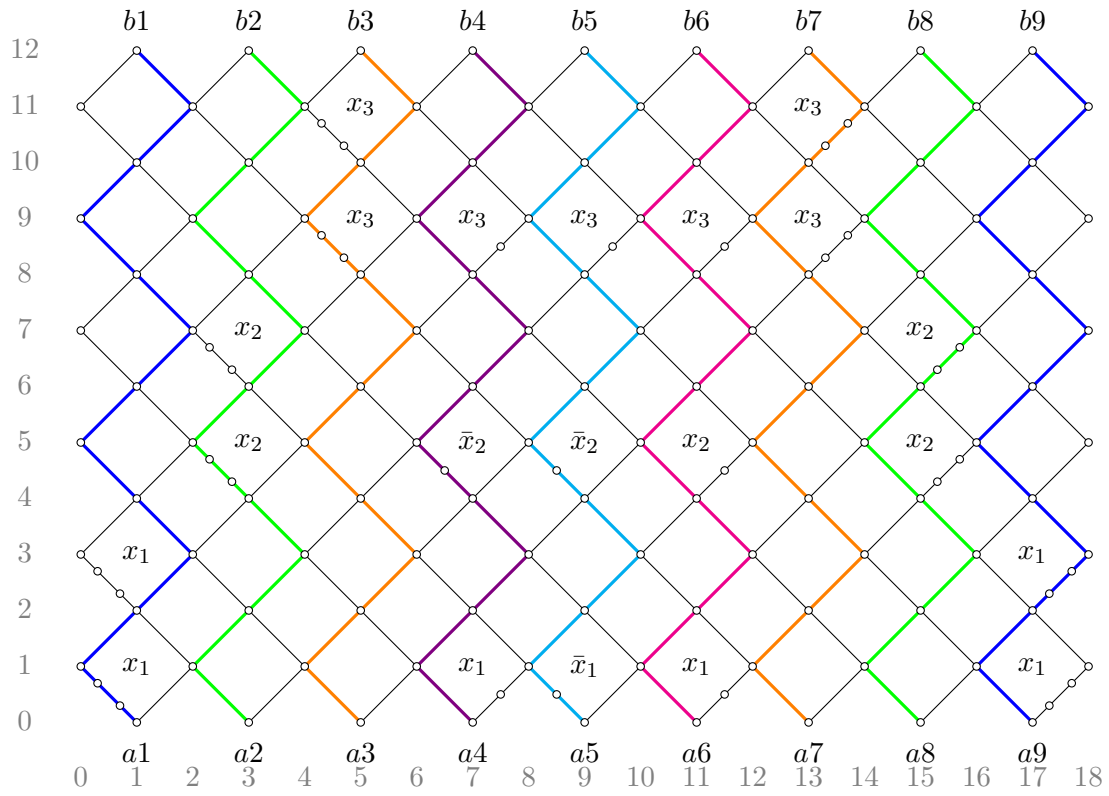


Figure 3.3: Graph resulting from construction Construction 3 using the following 3-SAT instance: $C = \{\{x_1, \bar{x}_2, x_3\}, \{\bar{x}_1, \bar{x}_2, x_3\}, \{x_1, x_2, x_3\}\}$. The colored paths in this figure represent a solution where every variable is assigned value 1. Again, all layers of our decaying graph are identical, thus one snapshot is able to represent the whole graph.

For all $i \in \{1, \ldots, n\}$, let

$$\begin{aligned} E_{x_i} \coloneqq &\{\{(4i-4, 2i-1), x^1_{i,1}\}, \{(4i-3, 2i-2), x^1_{i,2}\}, \{(4i-2, 2i-1), x^1_{i,3}\}, \\ &\{(4i-1, 2i-2), x^1_{i,4}\}, \{4i-4, 4n+2m-2i+1), x^2_{i,1}\}, \\ &\{(4i-3, 4n+2m-2i+2), x^2_{i,2}\}, \{(4i-2, 4n+2m-2i+1), x^2_{i,3}\}, \\ &\{(4i-1, 4n+2m-2i+2), x^2_{i,4}\}\}. \end{aligned}$$

Let $E_{1,1} \coloneqq \{\{(h,o),(i,l)\} \mid \max\{|h-i|, |o-l|\} = 1\}, h, i \in \{0, \ldots, 4n+2m\}, o, l \in \{0, \ldots, 4n\}\}$. The edge set E_{not} (these are the edges we need to "remove" from $E_{1,1}$, since we want to add extra vertices in their place) contains

- the edge set

  $$\begin{aligned} &\{\{\{(4i-4, 2i-1), (4i-3, 2i-2)\}, \{(4i-2, 2i-1), (4i-1, 2i-2)\}, \\ &\{(4i-4, 4n+2m-2i+1), (4i-3, 4n+2m-2i+2)\}, \\ &\{(4i-2, 4n+2m-2i+1), (4i-1, 4n+2m-2i+2)\}\} \mid i \in \{1, \ldots, n\}\}, \end{aligned}$$

  and

- for $i \in \{1, \ldots, m\}$ and $j \in \{1, \ldots, n\}$ the edge $\{(4i-4, 2(n+j)-1), (4i-3, 2(n+j))\}$, if C_i contains the positive literal x_j and the edges $\{(4i-4, 2(n+j)-1), (4i-3, 2(n+j)-2)\}$, if C_i contains the negative literal x_j.

Then the edge set E contains

- the edge sets $\bigcup_{i\in\{1,\ldots,n\}}\{\{x^1_{i,1}, x^1_{i,2}\}\}$ and $\bigcup_{i\in\{1,\ldots,n\}}\{\{x^1_{i,3}, x^1_{i,4}\}\}$,
- the edge sets $\bigcup_{i\in\{1,\ldots,n\}}\{\{x^2_{i,1}, x^2_{i,2}\}\}$ and $\bigcup_{i\in\{1,\ldots,n\}}\{\{x^2_{i,3}, x^2_{i,4}\}\}$,
- the edge sets $\bigcup_{i\in\{1,\ldots,n\}} E_{x_i}$ and $E_{1,1} \setminus E_{\text{not}}$, and
- for each $i \in \{1, \ldots, m\}$, $r \in \{1, 2, 3\}$, and $j \in \{1, \ldots, n\}$, the edges $\{\{(4i-4, 2(n+j)-1), c^i_r\}, \{(4i-3, 2(n+j)), c^i_r\}\}$, if C_i contains the positive literal x_j at position r, and the edges $\{\{(4i-4, 2(n+j)-1), c^i_r\}, \{(4i-3, 2(n+j)-2), c^i_r\}\}$, if C_i contains the negative literal x_j at position r.

Let $E_1 = \cdots = E_{2n+m} \coloneqq E$. ◊

Proof. *Theorem 3.3.* Let $(X = \{x_1, \ldots, x_n\}, C = \{C_1, \ldots, C_m\})$ be an instance of 3-SAT where n equals the number of variables and m equals the number of clauses. Let $I' = (G = (V, E_1, \ldots, E_{2n+m}), (a_1, b_1), \ldots, (a_{2n+m}, b_{2n+m}), k, d)$ be the instance of Planar Successive Evacuation obtained from the Construction 3 with $d = 0$, $k = 4n + 2$, and $a_i = (0, 2i-1)$ and $b_i = (4n, 2i-1)$ for all $i \in \{1, \ldots, 2n+m\}$. We show that I is a YES-instance if and only if I' is a YES-instance.
($\Rightarrow$) Let $X' \subseteq X$ be a solution for I. We construct the paths $(P_1, \ldots, P_{2n+m})$ as follows. For every $i \in \{1, \ldots, 2n+m\}$, $V(P_i)$ has to contain the vertex set $\{(j, 2i-1) \mid j \in \{0, \ldots, 4n\}, j \mod 2 = 0\}$. The paths P_j and P_{n+m+j}, $j \in \{1, \ldots, n\}$, represent the variable $x_j \in X$ in I. Then, for all $j \in \{1, \ldots, n\}$ and $i \in \{1, \ldots, 2n+m\}$, we do the following:

if $x_j \in X'$
add the vertices $\{x^1_{j,1}, x^1_{j,2}\}$ to $V(P_j)$,
$\{x^2_{j,3}, x^2_{j,4}\}$ to $V(P_{n+m+j})$, and
$\{(4j-3, 2i-2), (4j-1, 2i)\}$ to $V(P_i)$.

if $x_j \notin X'$
add the vertices $\{x^1_{j,3}, x^1_{j,4}\}$ to $V(P_j)$,
$\{x^2_{j,1}, x^2_{j,2}\}$ to $V(P_{n+m+j})$, and
$\{(4j-3, 2i), (4j-1, 2i-2)\}$ to $V(P_i)$.

The paths $\{P_1, \ldots, P_n\}$ and $\{P_{n+m}, \ldots, P_{2n+m}\}$ have a maximum length of $k = 4n + 2$. This is because the minimum length is $4n$ and we add exactly one vertex pair depending on wether $x_j \in X'$ or $x_j \notin X'$. For every $j \in \{1, \ldots, m\}$, the path P_{n+j} represents the clause $C_j \in C$. For every $i \in \{n+1, \ldots, n+m\}$ and $j \in \{1, 2, 3\}$, if $|N(c^i_j) \cap V(P_i)| = 2$ add vertex c^i_j to $V(P_{n+j})$. For every $j \in \{1, \ldots, m\}$, an extra vertex is added to P_{n+j} for each literal in C_j that admits the value false. This is because we can represent that way that at most two literals in each clause can admit the value false, or else our path would

be too long. We know that at least one literal in each clause C_i, $i \in \{1, \dots, m\}$, has the value true, thus a maximum of two extra vertices will be added and the maximum length of the paths $P_{n+1}, \dots, P_{n+m}$ is $k = 4n + 2$. It is easy to see that every vertex is contained in at most one path, thus $d = 0$. Thus, $(P_1, \dots, P_{2n+m})$ is a solution for I'.

($\Leftarrow$) Let $(P_1, \dots, P_{2n+m})$ be a solution for I'. We construct a solution $X' \subseteq X$ for instance I as follows. For all $i \in \{1, \dots, n\}$, the path P_i represents the variable $x_i \in X$ and the vertices $x^1_{i,1}, \dots, x^1_{i,4}$ represent the value of a variable x_i. We know that only one of two pairs $(x^1_{i,1}, x^1_{i,2})$ and $(x^1_{i,3}, x^1_{i,4})$, $i \in \{1, \dots, n\}$, can be contained in $V(P_i)$, since otherwise $|V(P_i)| > 4n + 2$. If $x^1_{i,1}, x^1_{i,2} \in V(P_i)$ we add x_i to X'. For all $i \in \{1, \dots, n\}$ the vertices c^i_1, c^i_2 and c^i_3 represent the three literals in each clause C_i in I and since $k = 4n + 2$ only two of those vertices can be contained in P_i to meet this condition. Thus, we know that at least one literal in each clause $C_1, \dots, C_m$ admits the value true and X' is a solution for I. □

3.2 NP-Hardness of Shortest Successive Evacuation

Shortest Successive Evacuation is NP-complete even for two disjoint paths and a maximum vertex degree of four.

Theorem 3.4. Shortest Successive Evacuation *is NP-complete even for* $\tau = 2$, $d = 0$, *and* $\Delta = 4$.

We give a polynomial-time many-one reduction from the NP-complete 3-SAT problem. The Construction 4 we use for this reduction is similar to Construction 1. The main difference being that we insert $2pn$ extra vertices between the vertices representing the clauses and the vertices representing the literals. This is because we want to "force" our first path to be the representative path for the variables $x \in X'$. We then remove the same edges as in Construction 1. While the second path is noticeably longer, it can only be of one certain length if it is a simple path.

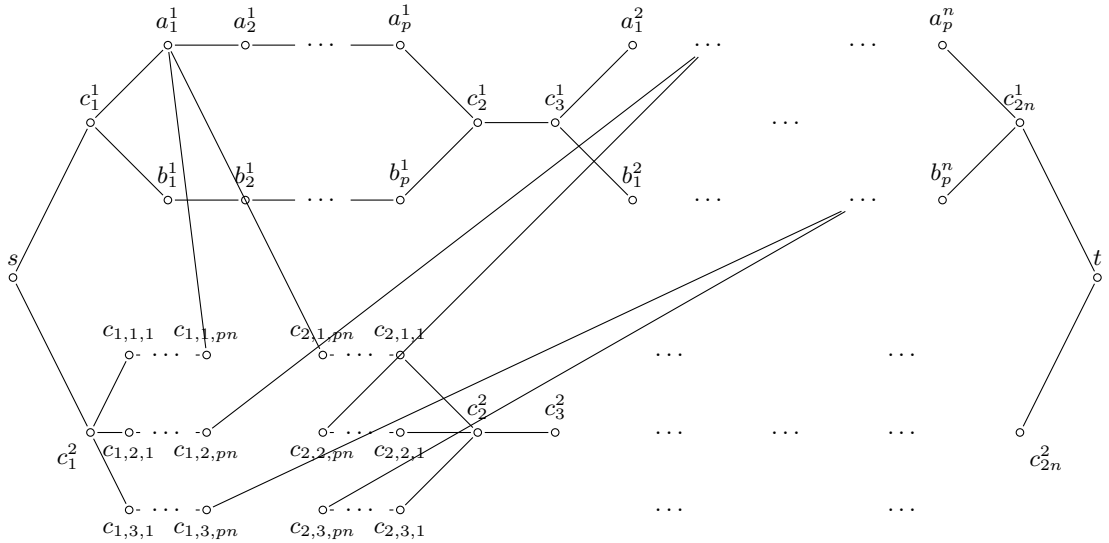
Construction 4. Let $(X = \{x_1, \dots, x_n\}, C = \{C_1, \dots, C_n\})$ be an instance of 3-SAT where the number of variables equals the number of clauses. Let $p \geq 2$ be the most frequent appearance of a variable in X. We construct a decaying graph $G = (V, E_1, E_2)$ as follows (see Figure 3.4 for an illustration).

Let

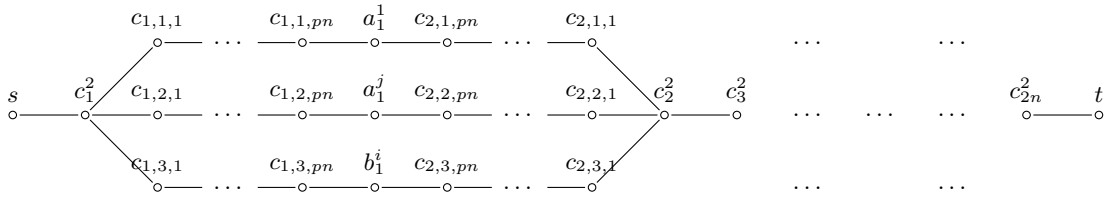
$$V \coloneqq \{s, t\} \cup \{c^i_1, \dots, c^i_{2n} \mid i \in \{1, 2\}\} \cup \{a^i_1, \dots, a^i_p \mid x_i \in X\} \cup \{b^i_1, \dots, b^i_p \mid x_i \in X\} \cup \{c_{i,j,1}, \dots, c_{i,j,pn} \mid i \in \{1, \dots, 2n\}, j \in \{1, 2, 3\}\}.$$

Let $E_{i,a} \coloneqq \bigcup_{j \in \{1,\dots,p-1\}} \{\{a^i_j, a^i_{j+1}\}\}$, let $E_{i,b} \coloneqq \bigcup_{j \in \{1,\dots,p-1\}} \{\{b^i_j, b^i_{j+1}\}\}$. Next we construct the edge sets $E_{1,1}$ and E_2. Set $E_{1,1}$ contains

- the edges $\{s, c^1_1\}$ and $\{t, c^1_{2n}\}$,
- the edge sets $\bigcup_{i \in \{1,\dots,n\}} \{\{c^1_{2i-1}, a^i_1\}, \{c^1_{2i-1}, b^i_1\}\}$ and $\bigcup_{i \in \{1,\dots,n\}} \{\{c^1_{2i}, a^i_p\}, \{c^1_{2i}, b^i_p\}\}$,
- the edge set $\bigcup_{i \in \{1,\dots,n-1\}} \{\{c^1_{2i}, c^1_{2i+1}\}\}$, and



(a) Snapshot of the first layer. $G = (V, E_1)$



(b) Snapshot of the second layer. $G = (V, E_2)$

Figure 3.4: Snapshots of the two layers of the decaying graph resulting from Construction 4, exemplified for clause $C_1 = (x_1 \vee x_j \vee \bar{x_i})$, with $i, j \in \{3, \ldots, p\}$ and $i \neq j$.

- the edge sets $\bigcup_{i \in \{1,\ldots,n\}} E_{i,a}$ and $\bigcup_{1 \leq i \leq n} E_{i,b}$.

Let $E_{w,h,c} \coloneqq \bigcup_{1 \leq i < pn-1} \{\{c_{w,h,i}, c_{w,h,i+1}\}\}$. Set E_2 is built in the following way. For each clause $C_q \in C$ we add the following:

- the edges $\{s, c_1^2\}$ and $\{t, c_{2n}^2\}$,
- the edge sets $\bigcup_{w \in \{1,\ldots,2n\}} \bigcup_{1 \leq h \leq 3} E_{w,h,c}$,
- the edge set $\bigcup_{i \in \{1,\ldots,2n\}} \{\{c_i^2, c_{i,1,1}\}, \{c_i^2, c_{i,2,1}\}, \{c_i^2, c_{i,3,1}\}\}$,
- for all $o \in \{1, 2, 3\}$ and $i \in \{1, \ldots, n\}$, if the literal at position o in C_q contains the j-th appearance, $j \in \{1, \ldots, p\}$, of the positive literal x_i add $\{c_{2q-1,o,pn}, a_j^i\}$, $\{c_{2q,o,pn}, a_j^i\}$,
- for all $o \in \{1, 2, 3\}$ and $i \in \{1, \ldots, n\}$, if the literal at position o in C_q contains the j-th appearance, $j \in \{1, \ldots, p\}$, of the negative literal x_i add $\{c_{2q-1,o,pn}, b_j^i\}$, $\{c_{2q,o,pn}, b_j^i\}$ and
- the edge set $\bigcup_{i \in \{1,\ldots,n\}} \{\{c_{2i}^2, c_{2i+1}^2\}\}$.

Let $E_1 \coloneqq E_{1,1} \cup E_2$. ◊

Proof. Theorem 3.4. Let $I = (X = \{x_1, \ldots, x_n\}, C = \{C_1, \ldots, C_n\})$ be an instance of 3-SAT where the number of variables equals the number of clauses. Let $I' = (G = (V, E_1, E_2), (a_1, b_1), (a_2, b_2), d)$ be the instance of Shortest Successive Evacuation obtained from Construction 4 with $d = 0$, $a_1 = a_2 = s$ and $b_1 = b_2 = t$. We show that I is a YES-instance if and only if I' is a YES-instance.

($\Rightarrow$) Let $X' \subseteq X$ be a solution for I. We construct the paths (P_1, P_2) as follows. For all $i \in \{1, \ldots, n\}$, vertex set $V(P_1)$ contains $\{b_1^i, \ldots, b_p^i\}$ if $x_i \in X'$ and $V(P_1)$ contains $\{a_1^i, \ldots, a_p^i\}$ if $x_i \notin X'$. Furthermore, $V(P_1)$ contains $\{s, t\} \cup \{c_1^1, \ldots, c_{2n}^1\}$. Let $H = (\bigcup_{i\{1,\ldots,n\}} \{a_1^i, \ldots, a_p^i\} \cup \bigcup_{i \in \{1,\ldots,n\}} \{b_1^i, \ldots, b_p^i\}) \setminus V(P_1)$ be an auxiliary vertex set. For each $q \in \{1, \ldots, n\}$ we define $V(C_q) \coloneqq N(c_{2q-1}^2) \cap N(c_{2q}^2)$ where $N(v)$ are the neighboring vertices of v. Vertex set $V(P_2)$ contains $\{s, t\} \cup \{c_1^2, \ldots, c_{2n}^2\}$ and if $a_j^i \in V(C_q) \cap H$ with i smallest, then $V(P_2)$ contains a_j^i, and if $b_j^i \in V(C_q) \cap H$ with i smallest, then $V(P_2)$ contains b_j^i, as well as their neighbor vertices $\{c_{2q,o,pn}, c_{2q-1,o,pn}\}$, $\{o \in \{1, 2, 3\}$, and the vertex sets $\bigcup_{1 \leq i < pn} \{c_{2q,o,i}\}$ and $\bigcup_{i \in \{1,\ldots,pn-1} \{c_{2q-1,o,i}\}$. Since $V(P_1) \cap H = \emptyset$ and $(V(P_2) \setminus (\{s, t\} \cup \{c_1^2, \ldots, c_{2n}^2\} \cup \{c_{i,j,1}, \ldots c_{i,j,pn} \mid i \in \{1, \ldots, 2n\}, j \in \{1, 2, 3\}, \}\})) \subseteq H$, $V(P_1) \cap V(P_2) = \emptyset$ and thus $|V(P_1) \cap V(P_2)| = 0 \leq d$. We know that $|V(P_1)| = 2 + 2n + pn$ and every other possible P_1 would have at least the length of size $7 + 2pn$ and it is clear that there is no other option than $|V(P_2)| = 2 + 3n + 2pn^2$. Thus, both paths are shortest paths.

($\Leftarrow$) Let (P_1, P_2) be a solution for I'. We construct a solution $X' \subset X$ for I as follows. If $V(P_1)$ contains the set $\{b_1^i, \ldots, b_j^i\}$, then $x_i \in X'$, and if $V(P_1)$ contains the set $\{a_1^i, \ldots, a_j^i\}$, then $x_i \notin X'$, for all $i \in \{1, \ldots, n\}$. For each clause $C_q \in C$, $q \in \{1, \ldots, n\}$, there exists a vertex pair $\{c_{2q-1}^2, c_{2q}^2\} \in V(P_2)$ and a vertex set $V_v \in V(P_1)$ that connects this pair. The vertex set V_v either contains a_j^i if x_i is a positive literal in C_q, or b_j^i if x_i is a negative literal in C_q. Thus, $x_i \in X'$ if $a_j^i \in V_v$, since $V(P_1) \cap V(P_2) = \emptyset$. Since this is the case for every clause in I and at least one literal is satisfied in each clause, I is a YES-instance. □

Chapter 4

XP and fixed-parameter tractability

In this chapter, will investigate whether Successive Evacuation is fixed-parameter tractable for parameter k and Shortest Successive Evacuation is in XP for a constant edge distance ζ and d. First, we give an algorithm that shows that Shortest Successive Evacuation is in XP when parameterized by ζ and d if $\tau = 2$. Then, in Section 4.2, we show that Successive Evacuation is fixed-parameter tractable for parameter k by using results of Fluschnik et al. [Flu+20] for Multistage s-t path.

4.1 Shortest Successive Evacuation parameterized by ζ, d, and $\tau = 2$

In this section, we give an XP-algorithm for Shortest Successive Evacuation for $\tau = 2$ and constant vertex overlap d and edge distance ζ. First, we introduce an algorithm for disjoint paths. Then we show how we can use Algorithm 1 to decide if the input instance is a YES-instance by duplicating vertex sets $S \subseteq V$ of size d.

4.1.1 For $d = 0$

We give an XP-algorithm for Shortest Successive Evacuation when $\tau = 2$, $d = 0$ and constant d. We start Algorithm 1 on a simple graph $G = (V, E_2)$. We then iterate over each edge $e \in E_1 \setminus E_2$ and represent this edge in our simple graph with an extra vertex v_e. This vertex v_e is used as a stopover for path P_1. We compute each an a_1-v_e and v_e-b_1 shortest path and use their combined length for comparison. By calling the algorithm recursively, we add one edge after another to our "stopover chain". By doing this, we replicate every possible combination of the missing edges in E_2.

Let $I = (G = (V, E_1, E_2), (a_1, b_1), (a_2, b_2), k, d)$ be an instance of Shortest Successive Evacuation with $d = 0$ and let $H = E_1 \setminus E_2$. We will not consider our end and start vertices in the path length, resulting in the path length being $|V(P)| - 2$.

Algorithm 1 ShortestSuccessiveEvacuationNonOverlapping

Input: $G = (V, E_1, E_2)$, $\{a_1, b_1\}$ and $\{a_2, b_2\}$ with $a_1, a_2, b_1, b_2 \in V$.

Output: A boolean value true or false, indicating if there exist two disjoint paths between vertex pairs $\{a_1, b_1\}$ and $\{a_2, b_2\}$ on G.

1: **function** ShortestSuccessiveEvacuationNonOverlapping($G = (V, E_1, E_2), \{a_1, b_1\}, \{a_2, b_2\}$)
2: $length1 \leftarrow$ ShortestPathLength($G = (V, E_1), \{a_1, b_1\}$)
3: $length2 \leftarrow$ ShortestPathLength($G = (V, E_2), \{a_2, b_2\}$)
4: $length \leftarrow length1 + length2$
5: **return** ShortestSuccessiveEvacuation($G = (V, E_2), \{\{a_1, b_1\}, \{a_2, b_2\}\}, E_1 \setminus E_2, length$)
6: **end function**

Algorithm 2 ShortestSuccessiveEvacuation

Input: $G = (V, E)$, a sorted set of vertex pairs $S = \{\{a_1, b_1\}, \{a_2, b_2\}\}$, an edge set H, and an integer $length$.

Output: A boolean value true or false, indicating if there exist two disjoint paths between vertex pairs $\{a_1, b_1\}$ and $\{a_2, b_2\}$ on G with the help of the auxiliary vertex set H.

function ShortestSuccessiveEvacuation($G = (V, E), S, H, length$)
2: $lengthAll \leftarrow 0$
for each $s \in S$ **do**
4: $lengthAll \leftarrow lengthAll +$ ShortestPathLength(G, s)
end for
6: **if** $lengthAll = length$ **then**
$l \leftarrow$ DisjointShortestPaths(G, S)
8: **if** $l = true$ **then**
return true
10: **end if**
end if
12: **for** each $e \in H$ **do**
$last \leftarrow$ last element $\{last_1, last_2\} \in S$
14: $\{v_1, v_2\} \leftarrow e$
$V_{\text{new}} \leftarrow V \cup v_e$.
16: $E_{\text{new}} \leftarrow E \setminus e \cup \{\{v_e, v_1\}, \{v_e, v_2\}\}$
$S \leftarrow (S \setminus last) \cup \{\{last_1, v_e\}, \{v_e, last_2\}\}$ ▷ Sorted union
18: $sol \leftarrow$ Algorithm 1 $(G = (V_{\text{new}}, E_{\text{new}}), S, H \setminus \{e\}, length)$
if $sol =$ **true then**
20: **return true**
end if
22: **end for**
return false
24: **end function**

Algorithm 3 ShortestPathsLength

Input: $G = (V, E)$ and a vertex pair $\{a, b\}$.
Output: Length of the shortest paths between the vertices a and b without a and b.

Algorithm 4 DisjointShortestPaths

Input: $G = (V, E)$ and a set S of vertex pairs
Output: A boolean value true or false. Deciding if vertex disjoint paths exist between the vertex pairs in S. Using the algorithm introduced by Bentert et al. [Ben+20].

Proof. (Correctness of Algorithm 1). ($\Rightarrow$) If there exists a solution for I, then Algorithm 1 will return true. It is easy to see that Algorithm 1 will return a solution if two disjoint paths exist on the edge set E_2. If this is not the case, the edges $e \in H$ are added to P_1 one at a time by representing them with vertex v_e. We then use the vertex v_e as a "layover" for the last path in S. This replicates the edge being contained in P_1. Since we do not consider the start and end vertices in our path length but only the "inner" vertices of the path, we can compare the sum of all paths to the initial length of the shortest path. Since S is a sorted set and in each iteration we "split" the last element in, by connecting the two endpoints with a different edge e, we compute each possible sequence. To compute the various disjoint paths, we use Algorithm 4 introduced by Bentert et al. [Ben+20], which proved correct. We require even the auxiliary paths to be disjoint. If they contained the same vertices, the resulting path a_1-b_1 would not be a simple path and therefore not a shortest path. Thus if there exists a solution Algorithm 1 returns true.

($\Leftarrow$) If Algorithm 1 returns true, then there is a valid solution for I. P_1 and P_2 are vertex disjoint, as P_1 is a combination of paths that all are pairwise vertex disjoint with P_2. Since the total length is at maximum the length of the combined length of two shortest paths between (a_1, b_1) and (a_2, b_2), computed in advance, the resulting paths are two shortest paths. □

Lemma 4.1. SHORTEST SUCCESSIVE EVACUATION *when parameterized by* ζ *and* $\tau = 2$ *is in XP.*

Proof. For any fixed $|E_1 \setminus E_2| = \zeta$, Algorithm 1 is a polynomial-time algorithm. In each recursion step, we call the function at most a total of $|H| - 1$ times, resulting in a total number of $\zeta!$ recursion steps. Since Bentert et al. [Ben+20] showed that k DISJOINT SHORTEST PATHS can be solved in $O(kn^{16k \cdot k! + k + 1})$ -time, the worst-case running time of our algorithm is in $O(\zeta! \cdot (\zeta + 1)n^{16(\zeta+1)\cdot(\zeta+1)! + (\zeta+1)+1})$. □

4.1.2 For constant d

We give an XP-algorithm for SHORTEST SUCCESSIVE EVACUATION for $\tau = 2$ and any constant $d > 0$. We do this by computing every possible vertex set $S \subseteq V$ with $|S| = d$ and creating a new graph $G_S = (V_S, E_1, E_{2_S})$ by duplicating all vertices in S and connecting the duplicate vertex to the neighbors of the original vertex. When we call Algorithm 1, we can visit each of these vertices twice. We do not take E_1 into account

Algorithm 5 DuplicateVertices

Input: $G = (V, E_1, E_2)$ and $H \subset V$
Output: $G = (V, E_1, E_2)$ with updated V and E_2

```
function DUPLICATE(G, H)
    for each i ∈ {1, ..., |H|} do
        V ← V ∪ v_i^2
        for each v ∈ N_{E_2}(v_i) do
            E_2 ← E_2 ∪ {{v_i^2, v}}
        end for
    end for
    return G
end function
```

Algorithm 6 ShortestSuccessiveEvacuationOverlapping

Input: $G = (V, E_1, E_2)$, $\{a_1, b_1\}$, $\{a_2, b_2\}$ and $d \in \mathbb{N}$
Output: wether there exist two paths $\{a_1, b_1\}$, $\{a_2, b_2\}$ with $P(\{a_1, b_1\}) \cap P(\{a_2, b_2\} \leq d$

```
function SHORTESTSUCCESSIVEEVACUATIONOVERLAPPING(G, H)
    if |V| ≤ d then
        return true
    end if
    for each S ⊂ V; |S| = d do
        G_new ← DuplicateVertices(G, S)
        a ← ShortestSuccessiveEvacuationNonOverlapping(G_new, {{a_1, b_1}, {a_2, b_2}})
        if a = true then
            return true
        end if
    end for
    return false
end function
```

when adding the edges for the duplicate vertices. We do this since the edges existing in E_1 and not in E_2 can only be contained in P_1. If P_1 contains the original vertex, P_2 can still contain the duplicate vertex.
Let $I = (G = (V, E_1, E_2), (a_1, b_1), (a_2, b_2), k, d)$ be an instance of SHORTEST SUCCESSIVE EVACUATION. For all $v \in V$ let $N_{E_2}(v)$ be the set of all adjacent vertices of v_i in the second layer of G.

Algorithm 5 will create a new graph $G' = (V', E_1, E_2')$ with certain duplicated vertices and a new edge set $E_2' := E_2 \cup \bigcup_{v_i \in H}(\bigcup_{v \in N(v_i)}\{\{v_i^2, v\}\}$.

Proof. (Correctness of Algorithm 6). ($\Rightarrow$) If there exists a solution for I, then Algorithm 6 will return true. It is easy to see that Algorithm 6 will return true if $d \geq |V|$. The other case is that $d < |V|$. We compute every possible subset $S \subseteq V$ with $|S| = d$ and an alternate graph G' using Algorithm 5. The extra vertices and edges in G' simulate the overlapping vertices since now it is possible for $V(P_1)$ to contain $v_i \in V$ and for $V(P_2)$ to contain v_i^2. Since the vertex set S has size d, we ensure that this vertex overlap

is at most the size of d. By not adding the edges between the duplicate vertex of v and its neighbors in E_1, we ensure that the size of the auxiliary vertex set $H = E_1 \setminus E_2$ in Algorithm 2 does not contain any "duplicate" edges. Since those could then occur in the same vertex-pair sequence and worsen the running time of Algorithm 2. As shown above, Algorithm 1, and consequently Algorithm 6, will return true if a solution exists.

($\Leftarrow$) If Algorithm 6 returns true, it is a valid solution for I. We showed that Algorithm 1 always found a valid solution if it returns true. Every vertex set $S \subseteq V$, used to compute G', is of size $|S| = d$. Thus at most d vertices can be contained in more than one path computed by Algorithm 4, so when Algorithm 6 returns true, there exists a solution for I. □

Lemma 4.2. SHORTEST SUCCESSIVE EVACUATION *for* $\tau = 2$ *and when parameterized by* ζ *is in XP.*

Proof. For any fixed d and $|E_1 \setminus E_2| = \zeta$, Algorithm 6 is a polynomial-time algorithm. The number of different subsets $S_i \subset V$ with $|S_i| = d$ is $\binom{n}{d}$. Therefore SHORTEST SUCCESSIVE EVACUATION can be solved in $O(\binom{n}{d}\zeta! \cdot (\zeta+1)n^{16(\zeta+1)\cdot(\zeta+1)!+(\zeta+1)+1})$-time for $\tau = 2$. □

4.2 Successive Evacuation parameterized with k

We prove that SUCCESSIVE EVACUATION is fixed-parameter tractable when parameterized by k. We do this by constructing a parameterized reduction to MULTISTAGE s-t PATH (as described below), proven to be fixed-parameter tractable when parameterized by k by Fluschnik et al. [Flu+20].

Theorem 4.3. SUCCESSIVE EVACUATION *is fixed-parameter tractable when parameterized by* k.

Problem 5: MULTISTAGE s-t PATH
Input: A temporal graph $G = (V, E_1, \ldots, E_\tau)$, origin-destination vertices s and t, and two integers $k, d \in \mathbb{N}$.
Question: Is there an s-t path P_i in (V, E_i) with $|V(P_i)| \leq k$ for all $i \in \{1, \ldots, \tau\}$ such that for all $j \in \{1, \ldots, \tau - 1\}$ it holds that $|V(P_j) \cap V(P_{j+1})| \leq d$?

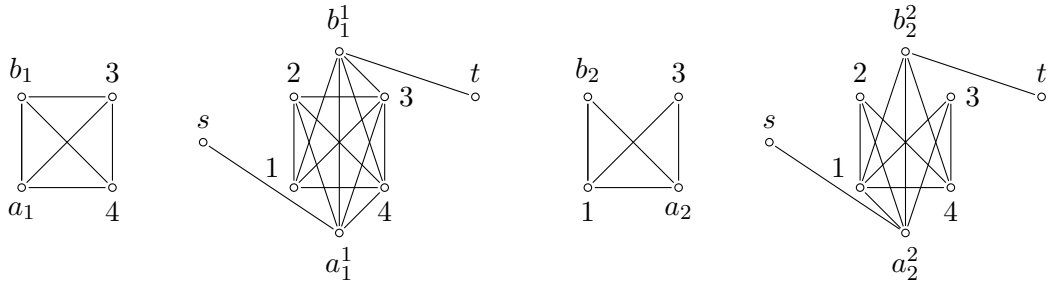
Construction 5. Let $I = (G = (V, E_1, \ldots, E_\tau), (a_1, b_1), \ldots, (a_\tau, b_\tau), k, d)$ an instance of SUCCESSIVE EVACUATION. We construct a temporal graph $G' = (V', E'_1, \ldots, E'_\tau)$ as follows (see Figure 4.1 for an illustration). Let $N(v)$ be the set of the neighboring vertices of $v \in V'$. Then the vertex set V' contains

- the vertex set V, the vertices s and t and
- for all $i \in \{1, \ldots, \tau\}$ the vertex a_i^i and $i \in \{1, \ldots, \tau\}$ the vertex b_i^i.

For every $i \in \{1, \ldots, \tau\}$ let

$$E'_i := E_i \cup \bigcup_{j \in N(a_i)} \{\{a_i^i, j\}\} \cup \{\{s, a_i^i\}\} \cup \bigcup_{j \in N(b_i)} \{\{b_i^i, j\}\} \cup \{\{t, b_i^i\}\}.$$

◊



(a) left: $G = (V, E_1)$, right: $G' = (V', E'_1)$ (b) left: $G = (V, E_2)$, right: $G' = (V', E'_2)$

Figure 4.1: Snapshots of two layers, (a) layer one and (b) layer two, of a temporal graph shows how the layers changed after applying Construction 5.

Proof. *Theorem 4.3.* Let $I = (G = (V, E_1, \ldots, E_\tau), (a_1, b_1), \ldots, (a_\tau, b_\tau), k, d)$ be an instance of Successive Evacuation and let $I' = (G' = (V', E'_1, \ldots, E'_\tau), s, t, k', d)$ be an instance of Multistage s-t Path obtained from Construction 5 with $k' = k + 2$. We show that I is a YES-instance if and only if I' is a YES-instance.

($\Rightarrow$) Let $(P_1, \ldots, P_\tau)$ be a solution for I. We construct the paths $(P'_1, \ldots, P'_\tau)$ as follows. For all $i \in \{1, \ldots, \tau\}$, let $V(P'_i) = (V(P_i) \cup \{a_i^i, b_i^i, s, t\}) \setminus \{a_i, b_i\}$ since a_i^i, b_i^i and a_i, b_i have the same outgoing edges (except for the edges to s and t) this path connects the start and end vertices. The length of P_i is at most k and we add exactly two additional vertices to $V(P'_i)$, thus $|V(P'_i)| \leq k + 2$. For all $j \in \{1, \ldots, \tau - 1\}$ we know that $|V(P_j) \overset{\circ}{\cap} V(P_{j+1})| \leq d$, thus $(V(P'_j) \overset{\circ}{\cap} V(P'_{j+1}) \leq d$. We added the extra vertices $a_j^j, a_{j+1}^{j+1}, b_j^j$ and b_{j+1}^{j+1} thus if $a_j, b_j, a_{j+1}, b_{j+1} \in V(P_j) \cap V(P_{j+1})$ it means that $a_j, b_j, a_{j+1}, b_{j+1} \notin V(P'_j) \cap V(P'_{j+1})$ and $|V(P'_j) \overset{\circ}{\cap} V(P'_j + 1)| \leq d$. Thus, $(P'_1, \ldots, P'_\tau)$ is a solution for I'.

($\Leftarrow$) Let $(P'_1, \ldots, P'_\tau)$ be a solution for I'. We construct the paths $(P_1, \ldots, P_\tau)$ as follows. For all $i \in \{1, \ldots, \tau\}$ let $V(P_i) = (V(P'_i) \cup \{a_i, b_i\}) \setminus \{s, t, a_i^i, b_i^i\}$. P'_1 has to contain the two corresponding vertices a_i^i and b_i^i to a_i and b_i, thus P_i has to be a a_i-b_i path if P'_i is an s-t path. It is easy to see that for all $i \in \{1, \ldots, \tau - 1\}$, $V(P_i) \overset{\circ}{\cap} V(P_{i+1}) \subseteq V(P'_i) \overset{\circ}{\cap} V(P'_{i+1})$, thus $|V(P_j) \overset{\circ}{\cap} V(P_{j+1})| \leq d$. Thus, $(P_1, \ldots, P_\tau)$ is a solution for I. □

Chapter 5

Problem Kernels

In this chapter, we study problem kernelization for Successive Evacuation and Shortest Successive Evacuation. We prove that both problems admit a problem kernel of size polynomial in the size of the vertex set $|V| = n$ if the input graph is a proper decaying graph.

5.1 No problem kernel of size polynomial in $k+d$ for Planar Successive Evacuation

To prove that Planar Successive Evacuation admits no problem kernel of size polynomial in $k + d$, we use an AND-cross-composition and a polynomial equivalence relation $\mathcal{R}$ as defined by Bodlaender, Jansen, and Kratsch [BJK10].

Let Σ be a finite alphabet. We call an equivalence relation $\mathcal{R}$ on $\Sigma*$ a polynomial equivalence relation $\mathcal{R}$ if there is an algorithm that can decide in polynomial time if two instances x and y are in the same equivalence class and if for any finite set $S \in \Sigma^*$ the number of classes that $\mathcal{R}$ divides S in is polynomially bounded by the largest element in S. Let $L \subseteq \Sigma^*$ be a language, $\mathcal{R}$ a polynomial equivalence relation on Σ^*, and let $\mathcal{Q} \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. An AND-composition is a polynomial-time algorithm that maps p different $\mathcal{R}$-equivalent instances $x_1, \ldots, x_p \in \Sigma^*$ of L on an instance $(x, k) \in \Sigma^* \times \mathbb{N}$ so that the parameter value k is polynomially bounded in $\max_i |x_i| + \log p$ and (x, k) is a YES-instance for $\mathcal{Q}$ if and only if every instance $x_1, \ldots, x_p$ is a YES-instance for L.

Bodlaender, Jansen, and Kratsch [BJK10] showed that if an NP-hard language L AND-cross-composes into the parameterized problem $\mathcal{Q}$, then $\mathcal{Q}$ does not admit a polynomial kernelization unless NP $\subseteq$ coNP/poly and the polynomial hierarchy collapses. We will use this going forward.

Theorem 5.1. *Unless NP $\subseteq$ coNP/poly,* Planar Successive Evacuation *does not admit a problem kernel of size polynomial in $k + d$.*

Two instances of Planar Successive Evacuation $I = (G, \{a_1, b_1\}, \ldots, \{a_\tau, b_\tau\}, k, d)$, $I' = (G', \{a'_1, b'_1\}, \ldots, \{a'_\tau, b'_\tau\}, k', d')$ are $\mathcal{R}$-equivalent if $|V(G)| = |V'(G)|$, $k = k'$ and $d = d'$. It is decidable in polynomial time if two instances I and I' are $\mathcal{R}$-equivalent. Thus, $\mathcal{R}$ is a polynomial equivalence relation.

Proposition 5.2. *There is an algorithm that computes in polynomial time an instance I of* Planar Successive Evacuation *if given p $\mathcal{R}$-equivalent different instances $I_1, \ldots, I_p$ of* Planar Successive Evacuation*, such that I is a YES-instance if and only if each of $I_1, \ldots, I_p$ is a YES-instance.*

Construction 6. Let $I_1 = (G_1, \{a_1, b_1\}, \ldots, \{a_{\tau_1}, b_{\tau_1}\}, k, d), \ldots, I_p = (G_p, \{a_{\sum_{i=1}^{p} \tau_i - (\tau_p - 1)}, b_{\tau' - (\sum_{i=1}^{p} \tau_i)}\}, \ldots, \{a_{\tau'}, b_{\tau'}\}, k, d)$ be p $\mathcal{R}$-equivalent instances of Planar Successive Evacuation and for all $j \in \{1, \ldots, p\}$, $G_j = (V_j, E_{\sum_{i=1}^{j} \tau_i - \tau_j 1}, \ldots, E_{\sum_{i=1}^{j} \tau_i})$, $d = 0$, and all vertex sets $\{V_1, \ldots, V_p\}$ are pairwise disjoint. We construct $I = (G', \{a_1, b_1\}, \ldots, \{a_{\tau'}, b_{\tau'}\}, k, d)$ with $G' = (V', E'_1, \ldots, E'_{\tau'})$ as follows. Let $V' = \bigcup_{1 \leq i \leq p} V_i$ and let $\tau' = \sum_{i=1}^{p} \tau_i$. For all $j \in \{1, \ldots, \tau'\}$, let $E'_j = \bigcup_{j \leq i \leq \tau'} E_i$. ◊

Proof. Theorem 5.1. Let $I_1 = (G_1, \{a_1, b_1\}, \ldots, \{a_{\tau_1}, b_{\tau_1}\}, k, d), \ldots, I_p = (G_p, \{a_{\sum_{i=1}^{p} \tau_i - (\tau_p - 1)}, b_{\tau' - (\sum_{i=1}^{p} \tau_i)}\}, \ldots, \{a_{\tau'}, b_{\tau'}\}, k, d)$ be p $\mathcal{R}$-equivalent instances of Planar Successive Evacuation and for all $j \in \{1, \ldots, p\}$, $G_j = (V_j, E_{\sum_{i=1}^{j} \tau_i - \tau_j 1}, \ldots, E_{\sum_{i=1}^{j} \tau_i})$, $d = 0$, and all vertex sets $\{V_1, \ldots, V_p\}$ are pairwise disjoint. We obtain the instance $I = (G', \{a_1, b_1\}, \ldots, \{a_{\tau'}, b_{\tau'}\}, k, d)$ with $\tau' = \sum_{i=1}^{p} \tau_i$ from Construction 6.

($\Rightarrow$) Let $(P_1, \ldots, P_{\tau'})$ be a solution for I. For each instance I_i with $i \in \{1, \ldots, p\}$, we construct the paths $(P_1^i, \ldots, P_{\tau_i}^i)$ as follows. Let $P_1^i = P_{\sum_{j=1}^{i} \tau_j - (\tau_i - 1)}, \ldots, P_{\tau_i}^i = P_{\sum_{j=1}^{i} \tau_j}$ for every $i \in \{1, \ldots, p\}$. The instances $\{I_1, \ldots, I_p\}$ are represented by pairwise disjoint vertex sets, thus two representative paths for two different instances will not influence each other. It is easy to see that each subset of paths $\{P_i, \ldots, P_j\} \in \{P_1, \ldots, P_{\tau'}\}$, $i, j \in \{1, \ldots, \tau'\}$ and $i \leq j$ fulfills the requirements of Planar Successive Evacuation, thus for all $i \in \{1, \ldots, p\}$ the paths $(P_1^i, \ldots, P_{\tau_i}^i)$ form a solution for I_i.

($\Leftarrow$) For all $i \in \{1, \ldots, p\}$, let $(P_1^i, \ldots, P_{\tau_i}^i)$ be a solution for I_i. We construct the paths $(P_1, \ldots, P_{\tau'})$ as follows. For all $i \in \{1, \ldots, p\}$ and $j \in \{1, \ldots, \tau_i\}$, let $P_{\sum_{q=1}^{i} \tau_q - j + 1} = P_{\tau_i - j + 1}^i$. We know that the paths on I resulting from an instance I_i, $i \in \{1, \ldots, p\}$ are always on a vertex subset $V_i \subseteq V'$ and all of those subsets are pairwise disjoint. Thus no two paths resulting from two separate instances will be interfering with each other. Since the sequence of origin-destination pairs in I is the same as in the starting instances the paths will also be disjoint, thus $(P_1, \ldots, P_{\tau'})$ is a solution for I. □

Planar Successive Evacuation is a special case of Successive Evacuation. Thus, Successive Evacuation also does not admit a problem kernel of size polynomial in $k + d$. Planar Successive Evacuation does also not admit a problem kernel of size polynomial in $k + d$ on proper decaying graphs since it is possible to add another component with at least τ edges to the graph and remove one each time step. This can be done in polynomial time. And since there are no edges connecting the additional component to the original graph, the result for this instance will not change.

5.2 No problem kernel of size polynomial in $d+\zeta$ for Shortest Successive Evacuation

In this section, we show that Shortest Successive Evacuation admits no problem kernel of size polynomial in $d + \zeta$ by using an AND-cross-composition.

Theorem 5.3. *Unless NP $\subseteq$ coNP/poly,* Shortest Successive Evacuation *does not admit a problem kernel of size polynomial in $d + \zeta$.*

We call two instances $I = (G, \{a_1, b_1\}, \ldots, \{a_\tau, b_\tau\}, d)$, $I' = (G', \{a'_1, b'_1\}, \ldots, \{a'_\tau, b'_\tau\}, d')$ $\mathcal{R}$-equivalent if $|V(G)| = |V'(G)|$, $\tau(G) = \tau(G')$ and $d = d'$.

Proposition 5.4. *There is an algorithm that computes in polynomial time an instance I of* Shortest Successive Evacuation *if given p $\mathcal{R}$-equivalent different instances $I_1, \ldots, I_p$ of* Shortest Successive Evacuation*, such that I is a YES-instance if and only if each of $I_1, \ldots, I_p$ is a YES-instance.*

Construction 7. Let $I_1 = (G_1, \{a_1, b_1\}, \{a_2, b_2\}, d), \ldots, I_p = (G_p, \{a_{2p-1}, b_{2p-1}\}, \{a_{2p}, b_{2p}\}, d)$ be p $\mathcal{R}$-equivalent instances of Shortest Successive Evacuation and for all $j \in \{1, \ldots, p\}$, $G_j = (V_j, E_{2j-1}, E_{2j})$,$d = 0$, $a_{2j-1} = a_{2j}$, $b_{2j-1} = b_{2j}$ and let the vertex sets $\{V_1, \ldots, V_p\}$ be pairwise disjoint. We construct $I = (G', \{s, t\}, \ldots, \{s, t\}, d)$ with $G' = (V', E'_1, \ldots, E'_{\tau'})$ as follows. Let

$$V^{j,1} := \bigcup_{1 \leq i \leq |V(G_1)| \cdot (j-1)} \{v_i^j\},$$

$$V^{j,2} := \bigcup_{|V(G_1)| \cdot (j-1) < i \leq 2|V(G_1)| \cdot (j-1)} \{v_i^j\},$$

$$V^j := V^{j,1} \cup V^{j,2},$$

and

$$\begin{aligned} E^j := &\{\{s, v_1^j\}, \{v^j_{|V(G_1)| \cdot (j-1)}, a_{2j-1}\}, \{v^j_{2|V(G_1)| \cdot (j-1)}, a_{2j}\}\} \cup \\ &\bigcup_{1 \leq i \leq |V(G_1)| \cdot (j-1) - 1} \{v_i^j, v_{i+1}^j\} \cup \\ &\bigcup_{|V(G_1)| \cdot (j-1) + 1 \leq i \leq 2|V(G_1)| \cdot (j-1) - 1} \{v_i^j, v_{i+1}^j\} \end{aligned}$$

for every $j \in \{1, \ldots, p\}$.

Let $V' := \bigcup_{1 \leq i \leq p} (V_i \cup V^{j,1} \cup V^{j,2})$ and let $\tau' = 2p$. For all $j \in \{1, \ldots, p\}$ let $E'_{2j-1} := \bigcup_{j \leq i \leq p} (E_{2i-1} \cup E^i)$ and $E'_{2j} := E_{2j} \cup (E^j \setminus \{v^j_{|V(G_1)| \cdot (j-1)}, a_{2j-1}\}) \cup \bigcup_{j+1 \leq i \leq p} (E_{2i-1} \cup E^i)$. $\Diamond$

Proof of Proposition 5.4. Let $I_1 = (G_1, \{a_1, b_1\}, \{a_2, b_2\}, d), \ldots, I_p = (G_p, \{a_{2p-1}, b_{2p-1}\}, \{a_{2p}, b_{2p}\}, d)$ be p $\mathcal{R}$-equivalent instances of Shortest Successive Evacuation and for all $j \in \{1, \ldots, p\}$, $G_j = (V_j, E_{2j-1}, E_{2j})$, $d = 0$, $a_{2j-1} = a_{2j}$, $b_{2j-1} = b_{2j}$ and

let the vertex sets $\{V_1, \ldots, V_p\}$ be pairwise disjoint. We obtain the instance $I = (G', \{s,t\}, \ldots, \{s,t\}, d')$ with $\tau' = 2p$ and $d' = 2$ from Construction 7.

($\Rightarrow$) Let $(P_1, \ldots, P_{\tau'})$ be a solution for I. For each instance I_i, with $i \in \{1, \ldots, p\}$ we construct the paths (P_1^i, P_2^i) as follows. Let the two paths $P_1^i = P_{2i-1} \setminus (\{s,t\} \cup V^{j,1})$ and $P_2^i = P_{2i} \setminus (\{s,t\} \cup V^{j,2})$ for every $i \in \{1, \ldots, p\}$. If we remove the two vertices s and t the instances $\{I_1, \ldots, I_p\}$ would be represented as different components in I. This ensures that paths of different starting instances will not interfere with one another. The instance I can have a vertex overlap of $d' = 2$. However, since for each of our starting instances the two start and end vertices are each the same vertex, this overlap is required for those two vertices and thus no overlap in P_1^i and P_2^i will occur. Through adding the vertex sets $V^1, \ldots, V^p$ we ensure that the two paths $V(P_{2i-1})$ and $V(P_{2i})$, representing P_1^i and P_2^i, are neighboring paths and consequently P_1^i and P_2^i are disjoint, for every $i \in \{1, \ldots, p-1\}$. They are neighboring paths because for all $i \in \{1, \ldots, p-1\}$, the size of $|V(P_{2i-1})|$ and $|V(P_{2i})|$ can be at most $|V(G_i)|\dot{(}j-1) + |V(G_i)|$ and the "chains" $V^{i+1,1}, V^{i+1,2}, \ldots, V^{p,1}, V^{p,2}$, are of size at least $|V(G_i)|\dot{(}j-1) + |V(G_i)|$. Therefore the size of the following paths $|V(P_{2i+1})|, \ldots, |V(P_{2p})|$ has to be at least $|V(G_i)|\dot{(}j-1) + |V(G_i)| + 2 > |V(G_i)|\dot{(}j-1) + |V(G_i)|$. Thus, for every $i \in \{1, \ldots, p\}$, (P_1^i, P_2^i) is a solution for I_i.

($\Leftarrow$) For all $i \in \{1, \ldots, p\}$ let (P_1^i, P_2^i) be a solution for the instance I_i. We construct the paths $(P_1, \ldots, P_{\tau'})$ as follows. Let $P_{2i-1} = P_1^i \cup \{s,t\} \cup V^{j,1}$ and $P_{2i} = P_2^i \cup \{s,t\} \cup V^{j,2}$ for every $i \in \{1, \ldots, p\}$. The vertex sets $V^{j,1}$ and $V^{j,2}$ do not have a connection to the representing component of I_j after a path used it. Each path of P_1^i and P_2^i can contain at most $|V(G_i)|$ vertices and $|V^{i+1,1}| = |V^{i,1}| + |V(G_i)|$ and therefore the path through the instance I_i at time step $2i-1$ and $2i$ is always shortest, $i \in \{1, \ldots, p\}$. Thus $(P_1, \ldots, P_{\tau'})$ is a solution for I.

□

5.3 Problem kernel of proper decaying graphs

We prove the following theorem:

Theorem 5.5. Successive Evacuation *and* Shortest Successive Evacuation *do admit a problem kernel of size polynomial in the number of vertices n if the input graph is a proper decaying graph.*

We know that every input instance trivially admits a problem kernel of size polynomial in $n + \tau$ from Fluschnik et al. [Flu+20].

Proof of Theorem 5.5. Let $G_j = (V_j, E_1, \ldots, E_\tau)$ and $\tau > \binom{n}{2}$. An undirected graph can have a maximum of $\binom{n}{2}$ edges. Per definition, a proper decaying graph has at least one edge removed in each time step. Thus, G can not be a proper decaying graph if $\tau > \binom{n}{2}$.

Since τ can admit a maximal size of $\binom{n}{2}$ it is easy to see that Successive Evacuation, Planar Successive Evacuation, and Shortest Successive Evacuation admit a problem kernel of size $n + \binom{n}{2}$. □

Chapter 6

Conclusion

We studied three problems: Successive Evacuation, Planar Successive Evacuation and Shortest Successive Evacuation. We found that all of those three problems are NP-hard. For example, NP-hardness for Successive Evacuation holds even if $\tau = 2$, both edge sets are identical, the vertex overlap is zero, and the maximum vertex degree is four. However, considering that Li, McCormick, and Simchi-Levi [LMSL90] proved that Two length-bounded Vertex-Disjoint Paths on undirected paths is NP-complete even for a vertex degree of three, we introduce the following corollary:

Corollary 6.1. Successive Evacuation *is NP-complete even if* $d = 0$, $\zeta = 0$, $\tau = 2$ *and* $\Delta = 3$.

We say this because in the reduction from Max 2-SAT to Two length-bounded Vertex-Disjoint Paths as described by Li, McCormick, and Simchi-Levi [LMSL90], all the edge weights are polynomial in the size of the input instance. Thus we could replace the weighted edges in the reduction by vertex chains where the vertex number equals the edge weight with it still being a polynomial-time many-one reduction. When considering a planar decaying graph, we do not prove NP-hardness for a constant parameter τ. This raises the question of whether the problem may be fixed-parameter tractable regarding parameter τ on planar decaying graphs. We further showed that Successive Evacuation is fixed-parameter tractable regarding parameter k via a parameterized reduction from Successive Evacuation to Multistage s-t Path.

For Shortest Successive Evacuation we found that it is NP-complete even when $d = 0$, $\tau = 2$ and $\Delta = 4$. We then showed that it is in XP for $\tau = 2$ when parameterized by ζ and d. It would be interesting to further research if Shortest Successive Evacuation with these parameters is still in XP for $\tau > 2$ or if a constant parameter τ is even required and if we can improve this result.

Our results show that unless NP $\subseteq$ coNP/poly, there exists no problem kernel of size polynomial in $k + d$ for Successive Evacuation and no problem kernel of size polynomial in $d + \zeta$ for Shortest Successive Evacuation.

Lastly, we proved that all problems have a problem kernel of size polynomial in n if the input graph is a proper decaying graph. However, it remains open if there is a problem kernel of size polynomial in n if the input graph is not a proper decaying graph.

In future work, research on how planarity influences the hardness of our problems on decaying graphs can be considered. By removing the edges that are required for the

second path in the first layer, we could show NP-hardness for Successive Evacuation and Shortest Successive Evacuation on planar temporal graphs (that are not decaying graphs) for the constant values $d = 0$, $\zeta = 0$, $\tau = 2$ and $\Delta = 4$. However, on decaying graphs, it remains open if the planar version of Shortest Successive Evacuation is NP-hard even if no parameter is constant. Further research on the complexity of Planar Successive Evacuation and what happens if Planar Successive Evacuation is parameterized by τ would be especially interesting since many road networks can be compared to planar graphs. Therefore, the planar versions seem particularly applicable regarding floods.

Literature

[Ben+20] Matthias Bentert, André Nichterlein, Malte Renken, and Philipp Zschoche. *Using a geometric lens to find k disjoint shortest paths*. In: *arXiv preprint arXiv:2007.12502* (2020) (cit. on pp. 11, 29).

[BJK10] Hans L Bodlaender, Bart MP Jansen, and Stefan Kratsch. *Cross-composition: A new technique for kernelization lower bounds*. In: *arXiv preprint arXiv:1011.4224* (2010) (cit. on p. 33).

[BKD14] Sandra Banholzer, James Kossin, and Simon Donner. *The impact of climate change on natural disasters*. In: *Reducing disaster: Early warning systems for climate change*. Springer, 2014, pp. 21–49 (cit. on p. 9).

[EMS14] David Eisenstat, Claire Mathieu, and Nicolas Schabanel. *Facility location in evolving metrics*. In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2014, pp. 459–470 (cit. on p. 10).

[Flu+20] Till Fluschnik, Rolf Niedermeier, Carsten Schubert, and Philipp Zschoche. *Multistage st path: Confronting similarity with dissimilarity in temporal graphs*. In: *31st International Symposium on Algorithms and Computation (ISAAC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik. 2020 (cit. on pp. 10, 15, 27, 31, 36).

[GR09] Brian J Gerber and Scott E Robinson. *Local government performance and the challenges of regional preparedness for disasters*. In: *Public Performance & Management Review* 32.3 (2009), pp. 345–371 (cit. on p. 9).

[GT11] Petr A Golovach and Dimitrios M Thilikos. *Paths of bounded length and their cuts: Parameterized complexity and algorithms*. In: *Discrete Optimization* 8.1 (2011), pp. 72–86 (cit. on p. 11).

[GTW14] Anupam Gupta, Kunal Talwar, and Udi Wieder. *Changing bases: Multistage optimization for matroids and matchings*. In: *International Colloquium on Automata, Languages, and Programming*. Springer. 2014, pp. 563–575 (cit. on p. 10).

[HP02] Hein van der Holst and José Coelho de Pina. *Length-bounded disjoint paths in planar graphs*. In: *Discrete Applied Mathematics* 120.1-3 (2002), pp. 251–261 (cit. on pp. 11, 20).

[LMSL90] Chung-Lun Li, S Thomas McCormick, and David Simchi-Levi. *The complexity of finding two disjoint paths with min-max objective function*. In: *Discrete Applied Mathematics* 26.1 (1990), pp. 105–115 (cit. on pp. 11, 18, 37).

[Loc21] Willian Lochet. *A Polynomial Time Algorithm for the k-Disjoint Shortest Paths Problem*. In: *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2021, pp. 169–178 (cit. on p. 11).

[TV96] Spyros Tragoudas and Yaakov L Varol. *Computing disjoint paths with length constraints*. In: *International Workshop on Graph-Theoretic Concepts in Computer Science*. Springer. 1996, pp. 375–389 (cit. on p. 11).