Parameterized Algorithms and Data Reduction for Safe Convoy Routing

René van Bevern
Department of Mechanics and Mathematics, Novosibirsk State University
Ulitsa Pirogova 2, 630090 Novosibirsk, Russian Federation
Sobolev Institute of Mathematics of the Siberian Branch of the Russian Academy of Sciences
Prospekt Akademiaka Koptyuga 4, 630090 Novosibirsk, Russian Federation
rvb@nsu.ru
https://orcid.org/0000-0002-4805-218X

Till Fluschnik
Institut für Softwaretechnik und Theoretische Informatik, TU Berlin
Ernst-Reuter-Platz 7, 10587 Berlin, Germany
till.fluschnik@tu-berlin.de

Oxana Yu. Tsidulko
Sobolev Institute of Mathematics of the Siberian Branch of the Russian Academy of Sciences
Prospekt Akademiaka Koptyuga 4, 630090 Novosibirsk, Russian Federation
Department of Mechanics and Mathematics, Novosibirsk State University
Ulitsa Pirogova 2, 630090 Novosibirsk, Russian Federation
tsidulko@math.nsc.ru

Abstract
We study a problem that models safely routing a convoy through a transportation network, where any vertex adjacent to the travel path of the convoy requires additional precaution: Given a graph \( G = (V, E) \), two vertices \( s, t \in V \), and two integers \( k, \ell \), we search for a simple \( s \)-\( t \)-path with at most \( k \) vertices and at most \( \ell \) neighbors. We study the problem in two types of transportation networks: graphs with small crossing number, as formed by road networks, and tree-like graphs, as formed by waterways. For graphs with constant crossing number, we provide a subexponential \( 2^{O(\sqrt{n})} \)-time algorithm and prove a matching lower bound. We also show a polynomial-time data reduction algorithm that reduces any problem instance to an equivalent instance (a so-called problem kernel) of size polynomial in the vertex cover number of the input graph. In contrast, we show that the problem in general graphs is hard to preprocess. Regarding tree-like graphs, we obtain a \( 2^{O(tw)} \cdot \ell^2 \cdot n \)-time algorithm for graphs of treewidth \( tw \), show that there is no problem kernel with size polynomial in \( tw \), yet show a problem kernel with size polynomial in the feedback edge number of the input graph.

2012 ACM Subject Classification
Mathematics of computing → Graph algorithms

Keywords and phrases
NP-hard problem, fixed-parameter tractability, problem kernelization, shortest path, secluded solution

Digital Object Identifier
10.4230/OASIcs.ATMOS.2018.10

Related Version

Funding
Till Fluschnik was supported by the German Research Foundation, grant NI 369/18.
René van Bevern and Oxana Yu. Tsidulko were supported by the Russian Foundation for Basic Research (RFBR), grant 18-501-12031 NNIO_a, while working on Sections 2.2, 2.3, 3.3, and 3.2.
While working on Sections 2.1 and 3.1, René van Bevern was supported by RFBR grant 16-31-
Acknowledgements
This research was initiated during a research visit of René van Bevern and Oxana Yu. Tsidulko at TU Berlin.

1 Introduction
Finding shortest paths is a fundamental problem in route planning and has extensively been studied with respect to efficient algorithms, including data reduction and preprocessing [1]. In this work, we study the following NP-hard variant of finding shortest s-t-paths.

**Problem 1.1 (Short Secluded Path (SSP)).**

*Input:* An undirected, simple graph \(G = (V, E)\) with two distinct vertices \(s, t \in V\), and two integers \(k \geq 2\) and \(\ell \geq 0\).

*Question:* Is there an s-t-path \(P\) in \(G\) such that \(|V(P)| \leq k\) and \(|N(V(P))| \leq \ell\)?

Herein, \(V(P)\) denotes the set of vertices on path \(P\) and \(N(V(P))\) denotes their set of neighbors (not lying on \(P\)).

The problem can be understood as finding short and safe routes for a convoy through a transportation network: each neighbor of the convoy’s travel path requires additional precaution. Thus, we seek to minimize not only the length of the convoy’s travel path, but also its number of neighbors. In our work, we study the above basic, unweighted variant, as well as a weighted variant of the problem, in which each vertex has two weights: one counts towards the path length, the other models the cost of precaution that has to be taken when the vertex occurs as the neighbor of the travel path.

Almost planar and tree-like transportation networks. The focus of our work is two-fold. Firstly, since the problem is NP-hard, we search for efficient algorithms in graphs that are likely to occur as transportation networks: almost planar graphs, which occur as road networks, and tree-like graphs, which arise as waterways (ignoring the few man-made canals, natural river networks form forests [23]). Secondly, given the effect that preprocessing and data reduction had to fundamental routing problems like finding shortest paths [1], we study the possibilities of polynomial-time data reduction with **provable performance guarantees** for SSP.

In order to measure the running time of our algorithms with respect to the “degree of planarity” or the “tree-likeness” of a graph, as well as to analyze the power of data reduction algorithms, we employ parameterized complexity theory, which provides us with the concepts of **fixed-parameter algorithms** and **problem kernelization** [16, 20, 39, 12]. Fixed-parameter algorithms have recently been applied to numerous NP-hard routing problems [30, 28, 29, 27, 3, 42, 41, 4, 15, 5, 26]. In particular, they led to subexponential-time algorithms for fundamental NP-hard routing problems in planar graphs [33] and to algorithms for hard routing problems that work efficiently on real-world data [3].

**Fixed-parameter algorithms.** The main idea of fixed-parameter algorithms is to accept the exponential running time seemingly inherent to solving NP-hard problems, yet to restrict the combinatorial explosion to a parameter of the problem, which can be small in applications. We call a problem **fixed-parameter tractable** if it can be solved in \(f(k) \cdot n^{O(1)}\) time on inputs of length \(n\) and some function \(f\) depending only on some parameter \(k\). In contrast to an algorithm that merely runs in polynomial time for fixed \(k\), fixed-parameter algorithms can solve NP-hard problems quickly if \(k\) is small.
Table 1 Overview of our results. Herein, $n$, $tw$, $vc$, $fes$, $cr$, and $\Delta$ denote the number of vertices, treewidth, vertex cover number, feedback edge number, the crossing number, and maximum degree of the input graph, respectively. “const.” abbreviates “constant”.

<table>
<thead>
<tr>
<th></th>
<th>On almost planar graphs (Sec. 2)</th>
<th>on tree-like graphs (Sec. 3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>exact solution</td>
<td>$2^{O(\sqrt{n})}$ time in graphs with const. $cr$ (Thm. 2.1)</td>
<td>$2^{O(tw)} \cdot \ell^2 \cdot n$ time (Thm. 3.2)</td>
</tr>
<tr>
<td>problem kernel</td>
<td>size $vc^{O(r)}$ in $K_{r,r}$-free graphs (Thm. 2.5)</td>
<td>size $fes^{O(1)}$ (Thm. 3.13)</td>
</tr>
<tr>
<td>lower bounds</td>
<td>No kernel with size poly($vc + r$) in $K_{r,r}$-free graphs and WK[1]-hard when parameterized by $vc + r$ (Thm. 2.14)</td>
<td>No kernel with size poly($tw + k + \ell$) even in planar graphs with const. $\Delta$ (Thm. 3.10)</td>
</tr>
</tbody>
</table>

Provably effective polynomial-time data reduction. Kernelization allows for provably effective polynomial-time data reduction. Note that a result of the form “our polynomial-time data reduction algorithm reduces the input size by at least one bit, preserving optimality of solutions” is impossible for NP-hard problems unless $P = NP$. In contrast, a kernelization algorithm reduces a problem instance into an equivalent one (the problem kernel) whose size depends only (ideally polynomially) on some problem parameter. Problem kernelization has been successfully applied to obtain effective polynomial-time data reduction algorithms for many NP-hard problems [25, 34] and also led to techniques for proving the limits of polynomial-time data reduction [7, 38, 9].

1.1 Our contributions

We study SSP (and a weighted variant) in two main classes of graphs: almost planar graphs and tree-like graphs. We refer to Table 1 for an overview on our main results. Regarding almost planar graphs, in graphs of constant crossing number, we show that (even the weighted version of) SSP is solvable in subexponential $2^{O(\sqrt{n})}$-time. Moreover, we prove that SSP is not solvable in $2^{o(\sqrt{n})}$-time in planar graphs unless the Exponential Time Hypothesis fails. In $K_{r,r}$-free graphs, which comprise the graphs with crossing number $O(r^3)$ [40], we show a problem kernel for SSP with size $vc^{O(r)}$, where $vc$ is the vertex cover number of the input graph. We prove that, unless the polynomial-time hierarchy collapses, there is no problem kernel of size polynomial in $vc + r$. Moreover, we prove that, unless the classes FPT and WK[1] coincide, SSP does not even allow for Turing kernels with size polynomial in $vc + r$; that is, we could not solve SSP in polynomial time even if we precomputed all answers to subproblems of size polynomial in $vc + r$ and could look them up in constant time. Regarding tree-like graphs, we prove that SSP is solvable in $2^{O(tw)} \cdot \ell^2 \cdot n$ time in graphs of treewidth $tw$ and that there is no problem kernel with size polynomial in $tw$. Instead, we show a problem kernel of size $fes^{O(1)}$, where $fes$ is the feedback edge number of the input graph.

Due to space constraints, results marked with $(\star)$ are deferred to a full version of the paper.

1.2 Related work

Several classical graph optimization problems have been studied in the “secluded” (small closed neighborhood) and the “small secluded” (small set with small open neighborhood) variants [2]. Luckow and Fluschnik [37] first defined SSP and analyzed its parameterized complexity with respect to the parameters $k$ and $\ell$. In contrast, we study problem parameters that describe the structure of the input graphs and are small in transportation networks. Chechik et al. [11] introduced the SECLUDED PATH problem, that, given an undirected
graph $G = (V, E)$ with two designated vertices $s, t \in V$, vertex-weights $w : V \to \mathbb{N}$, and two integers $k, C \in \mathbb{N}$, asks whether there is an $s$-$t$-path $P$ such that the size of the closed neighborhood $|N[V(P)]| \leq k$ and the weight of the closed neighborhood $w(N[V(P)]) \leq C$. Fomin et al. [21], in particular, prove that Secluded Path does not admit problem kernels with size polynomial in the vertex cover number $vc$. Our negative results on kernelization for SSP are significantly stronger: not only do we show that there is no problem kernel of size polynomial in $vc + r$ even in bipartite $K_{r,r}$-free graphs, we also show that SSP is W[1]-hard parameterized by $vc + r$. Golovach et al. [24] studied the “small secluded” scenario for finding connected induced subgraphs parameterized by the size $\ell$ of the open neighborhood. Their results obviously does not generalize to SSP, since SSP is NP-hard even for $\ell = 0$ [37].

1.3 Preliminaries

**Graph Theory.** We use basic notation from graph theory [14]. We study simple, finite, undirected graphs $G = (V, E)$. We denote by $V(G) := V$ the set of vertices of $G$ and by $E(G) := E$ the set of edges of $G$. We denote $n := |V|$ and $m := |E|$. For any subset $U \subseteq V$ of vertices, we denote by $N_G(U) = \{v \in V \setminus U \mid \exists v \in U : \{v, w\} \in E\}$ the open neighborhood of $U$ in $G$. When the graph $G$ is clear from the context, we drop the subscript $G$. A set $U \subseteq V$ of vertices is a vertex cover if every edge in $E$ has an endpoint in $U$. The size of a minimum vertex cover is called vertex cover number $vc(G)$ of $G$. A set $F \subseteq E$ of edges is a feedback edge set if the graph $(G, E \setminus F)$ is a forest. The minimum size of a feedback edge set in a connected graph is $m - n + 1$ and is called the feedback edge number $fes(G)$ of $G$. The crossing number $cr(G)$ of $G$ is the minimum number of crossings in any drawing of $G$ into the plane (where only two edges are allowed to cross in each point). A path $P = (V, E)$ is a graph with vertex set $V = \{x_0, x_1, \ldots, x_p\}$ and edge set $E = \{(x_i, x_{i+1}) \mid 0 \leq i < p\}$. We say that $P$ is an $x_0$-$x_p$-path of length $p$. We also refer to $x_0, x_p$ as the end points of $P$, and to all vertices $V \setminus \{x_0, x_p\}$ as the inner vertices of $P$. A $K_{r,r}$ is a complete bipartite graph $G = (U \sqcup V, E)$ with $|U| = |V| = r$. We say that a graph is $K_{r,r}$-free if it does not contain $K_{r,r}$ as a subgraph.

**Parameterized Complexity Theory.** For more details on parameterized complexity, we refer to the text books [16, 20, 39, 12]. Let $\Sigma$ be a finite alphabet. A parameterized problem $L$ is a subset $L \subseteq \Sigma^* \times \mathbb{N}$. An instance $(x, k) \in \Sigma^* \times \mathbb{N}$ is a yes-instance for $L$ if and only if $(x, k) \in L$. We call $x$ the input and $k$ the parameter.

**Definition 1.2** (fixed-parameter tractability, FPT). A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is fixed-parameter tractable if there is a fixed-parameter algorithm deciding $(x, k) \in L$ in time $f(k) \cdot |x|^{O(1)}$. The complexity class FPT consists of all fixed-parameter tractable problems.

**Definition 1.3** (kernelization). Let $L \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. A kernelization is an algorithm that maps any instance $(x, k) \in \Sigma^* \times \mathbb{N}$ to an instance $(x', k') \in \Sigma^* \times \mathbb{N}$ in $\text{poly}(|x| + k)$ time such that:

(i) $(x, k) \in L \iff (x', k') \in L'$, and

(ii) $|x'| + k' \leq f(k)$ for some computable function $f$.

We call $(x', k')$ the problem kernel and $f$ its size.

**Basic observations.** We may assume our input graph to be connected due to the following obviously correct and linear-time executable data reduction rule.

**Reduction Rule 1.4.** If $G$ has more than one connected component, then delete all but the component containing both $s$ and $t$ or return no if such a component does not exist.
2 Almost planar graphs

Many transportation networks such as rail and street networks are planar or at least have a small crossing number – the minimum number of edge crossings in a plane drawing of a graph. Unfortunately, SSP remains NP-hard even in planar graphs with maximum degree four and $\ell = 0$.

In this section, we present algorithms for SSP in graphs with constant crossing number. These, in particular, apply to planar graphs. First, in Section 2.1, we present a subexponential-time algorithm and a matching lower bound. Second, in Section 2.2, we present a provably effective data reduction algorithm. Finally, in Section 2.3, we show the limits of data reduction algorithms for SSP in graphs with small but non-constant crossing number.

2.1 A subexponential-time algorithm

In this section, we describe how to solve SSP in subexponential time in graphs with constant crossing number.

Theorem 2.1. Short Secluded Path is solvable in $2^{O(\sqrt{n})}$ time on graphs with constant crossing number.

We will also see a matching lower bound. To prove Theorem 2.1, we exploit that graphs with constant crossing number are $H$-minor free for some graph $H$.

Definition 2.2 (graph minor). A graph $H$ is a minor of a graph $G$ if $H$ can be obtained from $G$ by a sequence of vertex deletions, edge deletions, and edge contractions. If a graph $G$ does not contain $H$ as a minor, then $G$ is said to be $H$-minor free.

Bokal et al. [10] showed that, if a graph $G$ contains $K_{r,r}$ as a minor, then the crossing number of $G$ is $\text{cr}(G) \geq \frac{1}{2}(r - 2)^2$. Thus, any graph $G$ is $K_{r,r}$-minor free for $r > \sqrt{2\text{cr}(G)} + 2$, which goes in line with the well-known fact that planar graphs are $K_{3,3}$-minor free [43]. Demaine and Hajiaghayi [13] showed that, for any graph $H$, all $H$-minor free graphs have treewidth $tw \in O(\sqrt{n})$. To prove Theorem 2.1, it thus remains to show that SSP is solvable in $2^{O(tw)} \cdot \text{poly}(n)$ time, which is the main technical work deferred to Section 3.1.

Complementing Theorem 2.1, we can show a matching lower bound using the Exponential Time Hypothesis (ETH).

Conjecture 2.3 (Exponential Time Hypothesis (ETH), Impagliazzo et al. [32]). There is a constant $c$ such that $n$-variable 3-SAT cannot be solved in $2^{c(n+m)}$ time.

The ETH was introduced by Impagliazzo et al. [32] and since then has been used to prove running time lower bounds for various NP-hard problems (we refer to Cygan et al. [12, Chapter 14] for an overview). We use it to prove that Theorem 2.1 can be neither significantly improved in planar graphs nor generalized to general graphs.

Theorem 2.4. Unless the Exponential Time Hypothesis fails, Short Secluded Path has no $2^{o(\sqrt{n})}$-time algorithm in planar graphs and no $2^{o(n+m)}$-time algorithm in general.
As the first step towards proving Theorem 2.5, we will show that the following data reduction
where
weight functions
with constant crossing number. There, we exploited the fact that graphs with crossing
Lemma 2.9
Definition 2.7
In the previous section, we have shown a subexponential-time algorithm for
Reduction Rule 2.8
Note that an instance of
Problem 2.6
▶
Theorem 2.5.
▶
Proof. Assume that there is a \(2^{o(\sqrt{n})}\)-time algorithm for SSP in planar graphs and a \(2^{o(n)}\)-time algorithm for SSP in general graphs. Luckow and Fluschnik [37] give a polynomial-time many-one reduction from HAMILTONIAN CYCLE to SSP that maintains planarity and increases the number of vertices and edges by at most a constant. Thus, we get a \(2^{o(\sqrt{n})}\)-time algorithm for HAMILTONIAN CYCLE in planar graphs and a \(2^{o(n+m)}\)-time algorithm in general graphs. This contradicts ETH [12, Theorems 14.6 and 14.9].

2.2 Effective data reduction

In the previous section, we have shown a subexponential-time algorithm for SSP in graphs with constant crossing number. There, we exploited the fact that graphs with crossing number \(cr\) are \(K_{r,r}\)-minor free for \(r > \sqrt{2cr} + 2\). Of course, this means that they neither contain \(K_{r,r}\) as subgraph (indeed, one can show this even for \(r \geq 3.145 \cdot \sqrt{cr}\) using bounds from Pach et al. [40]).

In this section, we show how to reduce any instance of SSP in \(K_{r,r}\)-free graphs to an equivalent instance with size polynomial in the vertex cover number of the input graph. In the next section, we prove that this does not generalize to general graphs.

▶ Theorem 2.5. For each constant \(r \in \mathbb{N}\), SHORT SECLUDED PATH in \(K_{r,r}\)-free graphs admits a problem kernel with size polynomial in the vertex cover number of the input graph.

The proof of Theorem 2.5 consists of three steps. First, in linear time, we transform an \(n\)-vertex instance of SSP into an equivalent instance of an auxiliary vertex-weighted version of SSP with \(O(|vcr|)\) vertices. Second, using a theorem of Frank and Tardos [22], in polynomial time, we reduce the vertex weights to \(O(|vc^3|)\) so that the total instances size (in bits) becomes \(O(|vc^3|)\). Finally, since SSP is NP-complete in planar, and, hence, in \(K_{3,3}\)-free graphs, we can, in polynomial time, reduce the shrunk instance back to an instance of the unweighted SSP in \(K_{r,r}\)-free graphs. Due to the polynomial running time of the reduction, there is at most a polynomial blow-up of the instance size.

Our auxiliary variant of SSP allows each vertex to have two weights: one weight counts towards the length of the path, the other counts towards the number of neighbors:

▶ Problem 2.6 (VERTEX-WEIGHTED SHORT SECLUDED PATH (VW-SSP)).

Input: An undirected, simple graph \(G = (V, E)\) with two distinct vertices \(s, t \in V\), two integers \(k \geq 2\) and \(\ell \geq 0\), and vertex weights \(\kappa : V \to \mathbb{N}\) and \(\lambda : V \to \mathbb{N}\).

Question: Does \(G\) have an \(s\)-\(t\)-path \(P\) with \(\sum_{v \in V(P)} \kappa(v) \leq k\) and \(\sum_{v \in N(V(P))} \lambda(v) \leq \ell\)?

Note that an instance of SSP can be considered to be an instance of VW-SSP with unit weight functions \(\kappa\) and \(\lambda\). Our data reduction will be based on removing twins.

▶ Definition 2.7 (twins). Two vertices \(u\) and \(v\) are called (false) twins if \(N(u) = N(v)\).

As the first step towards proving Theorem 2.5, we will show that the following data reduction rule, when applied to a \(K_{r,r}\)-free instance of SSP for constant \(r\), leaves us with an instance of VW-SSP with \(O(|vc^3|)\) vertices.

▶ Reduction Rule 2.8. Let \((G, s, t, k, \ell, \kappa, \lambda)\) be a VW-SSP instance with unit weights, where \(G = (V, E)\) is a \(K_{r,r}\)-free graph.

For each maximal set \(U \subseteq V \setminus \{s, t\}\) of twins such that \(|U| > r\), delete \(|U| - r + 1\) vertices of \(U\) from \(G\), and, for an arbitrary remaining vertex \(v \in U\), set \(\lambda(v) := |U| - r\) and \(\kappa(v) := k + 1\).

▶ Lemma 2.9 (*). Reduction Rule 2.8 is correct and can be applied in linear time.
We now prove a size bound for the instances remaining after Reduction Rule 2.8.

\textbf{Proposition 2.10.} Applied to an instance of SSP with a \(K_{r,r}\)-free graph with vertex cover number \(vc\), Reduction Rules 2.8 and 1.4 yield an instance of VW-SSP on at most \((vc + 2) + r(vc + 2)^r\) vertices in linear time.

\textbf{Proof.} Let \((G', s, t, k, \ell, \lambda, \kappa')\) be the instance obtained from applying Reduction Rules 2.8 and 1.4 to an instance \((G, s, t, k, \ell, \lambda, \kappa)\).

Let \(C\) be a minimum-cardinality vertex cover for \(G'\) that contains \(s\) and \(t\), and let the vertex set of \(G'\) be \(V = C \cup Y\). Since \(G'\) is a subgraph of \(G\), one has \(|C| \leq vc(G') + 2 \leq vc(G) + 2 = vc + 2\). It remains to bound \(|Y|\). To this end, we bound the number of vertices of degree at least \(r\) in \(Y\) and the number of vertices of degree exactly \(i\) in \(Y\) for each \(i \in \{0, \ldots, r - 1\}\). Note that vertices in \(Y\) have neighbors only in \(C\).

Since Reduction Rule 1.4 has been applied, there are no vertices of degree zero in \(Y\).

Since Reduction Rule 2.8 has been applied, for each \(i \in \{1, \ldots, r - 1\}\) and each subset \(C' \subseteq C \cap C' \cap [C'] = i\), we find at most \(r\) vertices in \(Y\) whose neighborhood is \(C'\). Thus, for each \(i \in \{1, \ldots, r - 1\}\), the number of vertices with degree \(i\) in \(Y\) is at most \(r \cdot \binom{|C|}{i}\).

Finally, since \(G\) is \(K_{r,r}\)-free, any \(r\)-sized subset of the vertex cover \(C\) has at most \(r - 1\) common neighbors. Hence, since vertices in \(Y\) have neighbors only in \(C\), the number of vertices in \(Y\) of degree greater or equal to \(r\) is at most \((r - 1) \cdot \binom{|C|}{r}\). We conclude that

\[
|V'| \leq |C| + (r - 1) \cdot \binom{|C|}{r} + r \cdot \sum_{i=1}^{r-1} \binom{|C|}{i} \leq (vc + 2) + r(vc + 2)^r.
\]

This completes the first step of the proof of Theorem 2.5. Note that our data reduction works by “hiding” an unbounded number of twins in vertices of unbounded weights. The second step is thus reducing the weights of an VW-SSP instance. To this end, we are going to apply a theorem by Frank and Tardos [22], which was successfully applied in kernelizing weighted problems before [17].

\textbf{Proposition 2.11 (Frank and Tardos [22]).} There is an algorithm that, on input \(w \in \mathbb{Q}^d\) and integer \(N\), computes in polynomial time a vector \(\bar{w} \in \mathbb{Z}^d\) with \(\|\bar{w}\|_\infty \leq 2^{4d^2} N^d(d+2)\) such that \(\text{sign}(w^T b) = \text{sign}(\bar{w}^T b)\) for all \(b \in \mathbb{Z}^d\) with \(\|b\|_1 \leq N - 1\), where

\[\text{sign}(x) = \begin{cases} +1 & \text{if } x > 0, \\ 0 & \text{if } x = 0, \text{ and} \\ -1 & \text{if } x < 0. \end{cases}\]

\textbf{Observation 2.12.} For \(N \geq 2\), Proposition 2.11 gives \(\text{sign}(w^T e_i) = \text{sign}(\bar{w}^T e_i)\) for each \(i \in \{1, \ldots, d\}\), where \(e_i \in \mathbb{Z}^d\) is the vector that has 1 in the \(i\)-th coordinate and zeroes in the others. Thus, one has \(\text{sign}(w_i) = \text{sign}(\bar{w}_i)\) for each \(i \in \{1, \ldots, d\}\). That is, when reducing a weight vector from \(w\) to \(\bar{w}\), Proposition 2.11 maintains the signs of weights.

We apply Proposition 2.11 and Observation 2.12 to the weights of VW-SSP.

\textbf{Lemma 2.13.} An instance \(I = (G, s, t, k, \ell, \lambda, \kappa)\) of VW-SSP on an \(n\)-vertex graph \(G = (V, E)\) can be reduced in polynomial time to an instance \(I' = (G, s, t, k', \ell', \lambda', \kappa')\) of VW-SSP such that

\begin{enumerate}[(i)]
\item \(\{k', \kappa'(v), \ell', \lambda'(v)\} \subseteq \{0, \ldots, 2^{4(n+1)^3} \cdot (n + 2)^{(n+1)(n+3)}\}\), for each vertex \(v \in V\), and
\item \(I\) is a yes-instance if and only if \(I'\) is a yes-instance.
\end{enumerate}
Parameterized Algorithms and Data Reduction for Safe Convoy Routing

Proof. In this proof, we will conveniently denote the weight functions $\lambda, \lambda', \kappa$, and $\kappa'$ as vectors in $\mathbb{N}^n$ such that $\lambda_i = \lambda(i)$ for each $i \in V$, and similarly for the other weight functions.

We apply Proposition 2.11 with $d = n + 1$ and $N = n + 2$ to the vectors $(\lambda, \ell) \in \mathbb{N}^{n+1}$ and $(\kappa, k) \in \mathbb{N}^{n+1}$ to obtain vectors $(\lambda', \ell') \in \mathbb{Z}^{n+1}$ and $(\kappa', k') \in \mathbb{Z}^{n+1}$ in polynomial time.

(i) This follows from Proposition 2.11 with $d = n + 1$ and $N = n + 2$, and from Observation 2.12 since $(\lambda, \ell)$ and $(\kappa, k)$ are vectors of nonnegative numbers.

(ii) Consider an arbitrary $s$-$t$-path $P$ in $G$ and two associated vectors $x, y \in \mathbb{Z}^n$, where

$$x_v = \begin{cases} 1 & \text{if } v \in N(V(P)), \\ 0 & \text{otherwise}, \end{cases}$$

$$y_v = \begin{cases} 1 & \text{if } v \in V(P) \text{ and } \lambda(v) \leq \ell, \\ 0 & \text{otherwise}. \end{cases}$$

Observe that $||(x,-1)||_1 \leq n + 1$ and $||(y,-1)||_1 \leq n + 1$. Since $n + 1 \leq N - 1$, Proposition 2.11 gives $\text{sign}((\lambda, \ell)^\top (x, -1)) = \text{sign}((\lambda', \ell')^\top (x, -1))$ and $\text{sign}((\kappa, k)^\top (y, -1)) = \text{sign}((\kappa', k')^\top (y, -1))$, which is equivalent to

$$\sum_{v \in N(V(P))} \lambda(v) \leq \ell \iff \sum_{v \in N(V(P))} \lambda'(v) \leq \ell' \quad \text{and} \quad \sum_{v \in P} \kappa(v) \leq k \iff \sum_{v \in P} \kappa'(v) \leq k'. \quad \blacktriangleleft$$

We have finished two steps towards the proof of Theorem 2.5: we reduced SSP in $K_{r,r}$-free graphs for constant $r$ to instances of VW-SSP with $O(vc^\epsilon)$ vertices using Proposition 2.10 and shrunk its weights to encoding-length $O(vc^{3r})$ using Lemma 2.13. To finish the proof of Theorem 2.5, it remains to reduce VW-SSP back to SSP on $K_{r,r}$-free graphs.

### 2.3 Limits of data reduction

In Section 2.2, we have seen that SSP allows for problem kernels with size polynomial in $vc$ if the input graph is $K_{r,r}$-free for some constant $r$. A natural question is whether one can loosen the requirement of $r$ being constant.

The following Theorem 2.14(i) shows that, under reasonable complexity-theoretic assumptions, this is not the case: we cannot get problem kernels whose size bound depends polynomially on both $vc$ and $r$. Moreover, the following Theorem 2.14(ii) shows that, unless WK[1] = FPT, SSP does not even have Turing kernels with size polynomial in $vc + r$ [31]. That is, we could not even solve SSP in polynomial time if we had precomputed all answers to SSP instances with size polynomial in $vc + r$ and could look them up in constant time.

Both results come surprisingly: finding a standard shortest $s$-$t$-path is easy, whereas finding a short secluded path in general graphs is so hard that not even preprocessing helps.

**Theorem 2.14 (⋆).** Even in bipartite graphs, Short Secluded Path

i) has no problem kernel with size polynomial in $vc + r$ unless coNP $\subseteq$ NP/poly and

ii) is WK[1]-hard when parameterized by $vc + r$, where $vc$ is the vertex cover number of the input graph and $r$ is the smallest number such that the input graph is $K_{r,r}$-free.

The proof exploits that Multicolored Clique is WK[1]-hard parameterized by $k \log n$ [31]:

**Problem 2.15 (Multicolored Clique).**

*Input:* A $k$-partite $n$-vertex graph $G = (V, E)$, where $V = \bigcup_{i=1}^k V_i$ for independent sets $V_i$.

*Question:* Does $G$ contain a clique of size $k$?

We transfer the WK[1]-hardness of Multicolored Clique to SSP using the following type of reduction [31]:
The following reduction from vertices to \( \forall i \in \{1, \ldots, k\} \), and the graph \( G' = (V', E') \) is as follows. The vertex set \( V' \) consists of vertices \( s, t, u \) for each edge \( e \in E \), vertices \( w_h \) for \( h \in \{1, \ldots, \binom{k}{2} - 1\} \), and mutually disjoint \( \tilde{n} \)-binary vertex gadgets \( B_1, \ldots, B_k \), each vertex in which has \( \ell' + 1 \) neighbors of degree one. We denote

\[
    E'^* := \{ v_e \in V' \mid e \in E \}, \quad B := B_1 \sqcup B_2 \sqcup \cdots \sqcup B_k, \\
    E'_{ij} := \{ v_{(x,y)} \in E'^* \mid x \in V_i, y \in V_j \}, \quad \text{and} \quad W := \{ w_h \mid 1 \leq h \leq \binom{k}{2} - 1 \}.
\]
The edges of $G'$ are as follows. For each edge $e = \{v_i^p, v_j^q\} \in E$, vertex $v_e \in E_{ij}$ of $G'$ is $p$-connected to $B_i$ and $q$-connected to $B_j$. Vertex $s \in V'$ is adjacent to all vertices in $E_{1,2}$ and vertex $t \in V'$ is adjacent to all vertices in $E_{k-1,k}$. Finally, to describe the edges incident to vertices in $W$, consider any ordering of pairs $\{(i,j) \mid 1 \leq i < j \leq k\}$. Then, vertex $v_0 \in W$ is adjacent to all vertices in $E_{ij}$ and to all vertices in $E_{i',j'}$, where $(i,j)$ is the $h$-th pair in the ordering and $(i',j')$ is the $(h+1)$-st. This finishes the construction.

To prove Theorem 2.14, we show that Construction 2.19 is a polynomial-time many-one reduction that generates bipartite $K_{r,r}$-free graphs with $v + vc \in \text{poly}(k \log n)$.

**Lemma 2.20.** The graph created by Construction 2.19 from an $n$-vertex instance $G = (V_1 \cup V_2 \cup \ldots \cup V_k, E)$ of MULTICOLORED CLIQUE is bipartite, $K_{r,r}$-free for $r := 2k \log n + \binom{k}{2} + 2$, and admits a vertex cover of size $r - 1$.

**Proof.** The constructed graph $G' = (V', E')$ is bipartite with $V' = X \uplus Y$, where

$$X = \{s,t\} \cup W \cup B$$

and

$$Y = N(B) \cup E^*.$$

Hence, $X$ is a vertex cover of size at most $r - 1$ in $G'$. Finally, consider any $K_{r,r}$ whose vertex set is partitioned into two independent sets $X' \uplus Y' \subseteq V'$. Since $|X'| = |Y'| = r$, $|X' \cap Y'| \leq r - 1$, and $|Y' \cap X'| \leq r - 1$, we find $u \in X' \cap Y$ and $v \in Y' \cap X$. Observe that $\{u, v\}$ is an edge in the $K_{r,r}$, but not in $G'$. Thus, the $K_{r,r}$ is not a subgraph of $G'$.

**Lemma 2.21.** Construction 2.19 is a polynomial parameter transformation of MULTICOLORED CLIQUE parameterized by $k \log n$ into SSP in $K_{r,r}$-free graphs parameterized by $v + r$.

**Proof.** Let $I' := (G', s, t, k', \ell')$ be the SSP instance created by Construction 2.19 from an MULTICOLORED CLIQUE instance $G = (V, E)$. In Lemma 2.20, we already showed $v + r \in \text{poly}(k \log n)$. Thus, it remains to show that $G$ is a yes-instance if and only if $I'$ is.

($\Rightarrow$) Let $C$ be the edge set of a clique of size $k$ in $G$. For each $1 \leq i < j \leq k$, $C$ contains exactly one edge $e$ between $V_i$ and $V_j$. Thus, $E_C := \{v_e \in E^* \mid e \in C\}$ is a set of $\binom{k}{2}$ vertices — exactly one vertex of $E_{ij}$ for each $1 \leq i < j \leq k$. Thus, by Construction 2.19, $G'$ contains an $s$-path $P = (V_P, E_P)$ with $|V_P| \leq k^2$: its inner vertices are $E_C \cup W$, alternating between these two sets. To show that $I'$ is a yes-instance, it remains to show that $|N(V_P)| \leq \ell'$.

Since $P$ contains all vertices of $W$, one has $N(V_P) \subseteq B \cup (E^* \setminus E_C)$, where $|E^* \setminus E_C| = |E| - \binom{k}{2}$. To show $|N(V_P)| \leq \ell'$, it remains to show that $|N(V_P) \cap B| \leq k \log \tilde{n}$. To this end, we show that $|N(V_P) \cap B_i| \leq \log \tilde{n}$ for each $i \in \{1, \ldots, k\}$.

The vertices in $W \cup \{s,t\}$ have no neighbors in $B$. Thus, consider arbitrary vertices $v_{e_1}, v_{e_2} \in E_C$ such that $N(v_{e_1}) \cap B_i \neq \emptyset$ and $N(v_{e_2}) \cap B_i \neq \emptyset$ for some $i \in \{1, \ldots, k\}$ (possibly, $e_1 = e_2$). Then, $e_1 = \{v_i^p, v_j^q\}$ and $e_2 = \{v_i^{p'}, v_j^{q'}\}$. Since $C$ is a clique, $e_1$ and $e_2$ are incident to the same vertex of $V_i$. Thus, we have $p = p'$. Both $v_{e_1}$ and $v_{e_2}$ are therefore $p$-connected to $B_i$ and hence have the same log $\tilde{n}$ neighbors in $B_i$. It follows that $N(V_P) \leq \ell'$ and, consequently, that $I'$ is a yes-instance.

($\Leftarrow$) Let $P = (V_P, E_P)$ be an $s$-path in $G'$ with $|V_P| \leq k'$ and $|N(V_P)| \leq \ell'$. The path $P$ does not contain any vertex of $B$, since each of them has $\ell' + 1$ neighbors of degree one. Thus, the inner vertices of $P$ alternate between vertices in $W$ and in $E^*$ and we get $N(V_P) = (E^* \setminus V_P) \cup (N(V_P) \cap B)$. Since $P$ contains one vertex of $E_{ij}$ for each $1 \leq i < j \leq k$, we know $|E^* \setminus V_P| = |E| - \binom{k}{2}$. Thus, since $|N(V_P)| \leq \ell'$, we have $|N(V_P) \cap B| \leq k \log \tilde{n}$. We exploit this to show that the set $C := \{e \in E \mid v_e \in V_P \cap E^*\}$ is the edge set of a clique.
in \(G\). To this end, it is enough to show that, for each \(i \in \{1, \ldots, k\}\), any two edges \(e_1, e_2 \in C\) with \(e_1 \cap V_i \neq \emptyset\) and \(e_2 \cap V_i \neq \emptyset\) have the same endpoint in \(V_i\); then \(C\) is a set of \(\binom{k}{2}\) edges on \(k\) vertices and thus forms a \(k\)-clique.

For each \(1 \leq i < j \leq k\), \(P\) contains exactly one vertex \(v \in E_{ij}\), which has exactly \(\log n\) neighbors in each of \(B_i\) and \(B_j\). Thus, from \(|N(V_P) \cap B_i| \leq k\) \(\log n\) follows \(|N(V_P) \cap B_i| = \log n\) for each \(i \in \{1, \ldots, k\}\). It follows that, if two vertices \(v_{e_1}\) and \(v_{e_2}\) on \(P\) both have neighbors in \(B_i\), then both are \(p\)-connected to \(B_i\) for some \(p\), which means that the edges \(e_1\) and \(e_2\) of \(G\) share endpoint \(v_P^i\).

We conclude that \(C\) is the edge set of a clique of size \(k\) in \(G\). Hence, \(G\) is a yes-instance. \(\blacksquare\)

To prove Theorem 2.14, it is now a matter of putting together Lemma 2.21 and the fact that \textsc{Multicolored Clique} parameterized by \(k \log n\) is \(\text{WK}[1]\)-complete.

\textbf{Proof of Theorem 2.14.} By Lemma 2.21, Construction 2.19 is a polynomial parameter transformation from \textsc{Multicolored Clique} parameterized by \(k \log n\) to \textsc{SSP} parameterized by \(vc + r\) in \(K_{r,r}\)-free graphs.

\textsc{Multicolored Clique} parameterized by \(k \log n\) is known to be \(\text{WK}[1]\)-complete \cite{DBLP:journals/tcs/KratschW09} and hence, does not admit a polynomial-size problem kernel unless \(\text{coNP} \subseteq \text{NP/poly}\). From the polynomial parameter transformation in Construction 2.19, it thus follows that \textsc{SSP} is \(\text{WK}[1]\)-hard parameterized by \(vc + r\) and does not admit a polynomial-size problem kernel unless \(\text{coNP} \subseteq \text{NP/poly}\). \(\blacksquare\)

\section{Tree-like graphs}

In this section, we present results for \textsc{SSP} in tree-like graphs. Such graphs naturally arise as waterways: when ignoring the few man-made canals, the remaining, natural waterways usually form a forest \cite{DBLP:conf/isssl/FominLM12}.

Moreover, graphs of small tree-width (formally defined in Section 3.1) are interesting since, as described in Section 2.1, graphs with constant crossing number have treewidth at most \(\sqrt{q}\) for many graph parameters \(q\). Thus, one can derive subexponential-time algorithms for these parameters from single-exponential algorithms for treewidth, like we did in Section 2.1.

First, in Section 3.1, we describe an algorithm that efficiently solves \textsc{SSP} on graphs of small treewidth. Second, in Section 3.2, we show that \textsc{SSP} allows for no problem kernel with size polynomial in the treewidth of the input graph. Third, in Section 3.3, we complement this negative result by a problem kernel with size polynomial in the feedback edge number of the input graph.

\subsection{Fixed-parameter algorithm for graphs with small treewidth}

In this section, we sketch a \(2^{O(\text{tw})} \cdot n^2 \cdot n\)-time algorithm for \textsc{SSP} in graphs of treewidth \(\text{tw}\), which will also conclude the proof of the \(2^{O(\sqrt{q})}\)-time algorithm for \textsc{SSP} in graphs with constant crossing number (Theorem 2.1). Before describing the algorithm, we formally introduce the treewidth concept.

\textbf{Definition 3.1 (tree decomposition, treewidth).} A tree decomposition \(T = (T, \beta)\) of a graph \(G = (V, E)\) consists of a tree \(T\) and a function \(\beta : V(T) \rightarrow 2^V\) that associates each node \(x\) of the tree \(T\) with a subset \(B_x := \beta(x) \subseteq V\), called a bag, such that

\begin{itemize}
  \item[i)] for each vertex \(v \in V\), there is a node \(x\) of \(T\) with \(v \in B_x\),
  \item[ii)] for each edge \(\{u, v\} \in E\), there is a node \(x\) of \(T\) with \(\{u, v\} \subseteq B_x\),
  \item[iii)] for each \(v \in V\) the nodes \(x\) with \(v \in B_x\) induce a subtree of \(T\).
\end{itemize}

The width of \(T\) is \(w(T) := \max_{x \in V(T)} |B_x| - 1\). The treewidth of \(G\) is \(\text{tw}(G) := \min\{w(T) \mid T\ is\ a\ tree\ decomposition\ of\ G\}\).
Theorem 3.2. Short Secluded Path is solvable in $2^{O(tw)} \cdot \ell^2 \cdot n$ time in graphs of treewidth $tw$.

Bodlaender et al. [8] proved that a tree decomposition of width $O(tw(G))$ of a graph $G$ is computable in $2^{O(tw)} \cdot n$-time. Applying the following Proposition 3.3 to such a tree decomposition yields Theorem 3.2:

Proposition 3.3 (*). Vertex-Weighted Short Secluded Path is solvable in $n \cdot \ell^2 \cdot tw^{O(1)} \cdot (2 + 12 \cdot 2^{\omega})^{tw}$ time when a tree decomposition of width $tw$ is given, where $\omega < 2.2373$ is the matrix multiplication exponent.

To prove Theorem 3.2, it thus remains to prove Proposition 3.3. Note that Proposition 3.3 actually solves the weighted problem VW-SSP (Problem 2.6), where the term $\ell^2$ is only pseudo-polynomial for VW-SSP. It is a true polynomial for SSP since we can assume $\ell \leq n$.

3.1.1 Assumptions on the tree decomposition

Our algorithm will work on simplified tree decompositions, which can be obtained from a classical tree decomposition of width $tw$ in $n \cdot tw^{O(1)}$ time without increasing its width [6].

Definition 3.4 (nice tree decomposition). A nice tree decomposition $T$ is a tree decomposition with one special bag $r$ called the root and in which each bag is of one of the following types.

Leaf node: a leaf $x$ of $T$ with $B_x = \emptyset$.

Introduce vertex node: an internal node $x$ of $T$ with one child $y$ such that $B_x = B_y \cup \{v\}$ for some vertex $v \notin B_y$. This node is said to introduce vertex $v$.

Introduce edge node: an internal node $x$ of $T$ labeled with an edge $\{u, v\} \in E$ and with one child $y$ such that $\{u, v\} \subseteq B_x = B_y$. This node is said to introduce edge $\{u, v\}$.

Forget node: an internal node $x$ of $T$ with one child $y$ such that $B_x = B_y \setminus \{v\}$ for some node $v \in B_y$. This node is said to forget $v$.

Join node: an internal node $x$ of $T$ with two children $y$ and $z$ such that $B_x = B_y = B_z$.

We additionally require that each edge is introduced at most once and make the following, problem specific assumptions on tree decompositions.

Assumption 3.5. When solving VW-SSP, we will assume that the source $s$ and destination $t$ of the sought path are contained in all bags of the tree decomposition and that the root bag contains only $s$ and $t$. This ensures that

- every bag contains vertices of the sought solution, and that
- $s$ and $t$ are never forgotten nor introduced.

Such a tree decomposition can be obtained from a nice tree decomposition by rooting it at a leaf (an empty bag) and adding $s$ and $t$ to all bags. This will increase the width of the tree decomposition by at most two.

Our algorithm will be based on computing partial solutions for subgraphs induced by a node of a tree decomposition by means of combining partial solutions for the subgraphs induced by its children. Formally, these subgraphs are the following.

Definition 3.6 (subgraphs induced by a tree decomposition). Let $G = (V, E)$ be a graph and $T$ be a nice tree decomposition for $G$ with root $r$. Then, for any node $x$ of $T$,

$$V_x := \{v \in V \mid v \in B_y \text{ for a descendant } y \text{ of } x\},$$

and

$$G_x := (V_x, E_x),$$

where $E_x = \{e \in E \mid e \text{ is introduced in a descendant of } x\}$.

Herein, we consider each node $x$ of $T$ to be a descendent of itself.

Having defined subgraphs induced by subtrees, we can define partial solutions in them.
3.1.2 Partial solutions

Assume that we have a solution path $P$ to VW-SSP. Then, the part of $P$ in $G_x$ is a collection $\mathcal{P}$ of paths (some might consist of a single vertex). When computing a partial solution for a parent $y$ of $x$, we ideally want to check which partial solutions for $x$ can be continued to partial solutions for $y$. However, we cannot try all possible partial solutions for $G_x$ - there might be too many. Moreover, this is not necessary: by Definition 3.1(ii)-(iii), vertices in bag $B_y$ cannot be vertices of and cannot have edges to vertices of $V_x \setminus B_x$. Thus, it is enough to know the states of vertices in bag $B_x$ in order to know which partial solutions of $x$ can be continued to $y$. The state of such vertices is characterized by

- which vertices of $B_x$ are end points of paths in $\mathcal{P}$, inner vertices of paths in $\mathcal{P}$, or paths of zero length in $\mathcal{P}$,
- which vertices of $B_x$ are allowed to be neighbors of the solution path $P$,
- how many neighbors the solution path $P$ is allowed to have in $G_x$, and
- which vertices of $B_x$ belong to the same path of $P$.

Definition 3.7 (partial solution). Let $(G, s, t, k, \ell, \kappa, \lambda)$ be an instance of VW-SSP. For a set $\mathcal{P}$ of paths in $G$ and a set $N$ of vertices in $G$, let

$$\Lambda(P, N) := \sum_{P \in \mathcal{P}} \sum_{v \in N(V(P))} \lambda(v) + \sum_{v \in N} \lambda(v) \quad \text{and} \quad K(P) := \sum_{P \in \mathcal{P}} \sum_{v \in V(P)} \kappa(v).$$

Moreover, let $T$ be a tree decomposition for $G$, $x$ be a node of $T$, $D_z \cup D_e \cup D_i \cup N \subseteq B_x$ such that $\{s, t\} \subseteq D_z \cup D_e$, $p$ be a partition of $D := D_z \cup D_e \cup D_i$, and $l \leq \ell$.

Then, we call $(D_z, D_e, D_i, N, l)$ a pre-signature and $S = (D_z, D_e, D_i, N, l, p)$ a solution signature at $x$. A set $\mathcal{P}$ of paths in $G_x$ is a partial solution of cost $K(P)$ for $S$ if

i) $D_z$ are exactly the vertices of zero-length paths $P \in \mathcal{P}$,
ii) $D_e$ are exactly the end points of non-zero-length paths $P \in \mathcal{P}$,
iii) $D_i$ are exactly those vertices in $B_x$ that are inner vertices of paths $P \in \mathcal{P}$,
iv) for each path $P \in \mathcal{P}$, $N(V(P)) \cap B_x \subseteq N$,
v) $\Lambda(P, N) \leq \ell$, and
vi) $\mathcal{P}$ consists of exactly $|p|$ paths such that each two vertices $u, v \in D$ belong to the same path of $\mathcal{P}$ if and only if they are in the same set of the partition $p$.

For a solution signature $S$ at a node $x$, we denote

$$\mathcal{E}_x(S) := \{\mathcal{P} \mid \mathcal{P} \text{ is a partial solution for } S\},$$
$$\min_{K_x}(S) := \min\{K(P) \mid \mathcal{P} \in \mathcal{E}_x(S)\}.$$
Because of Assumption 3.5, our input instance to VW-SSP is a yes-instance if and only if
\[ \min_{K_x}(\emptyset, \{s, t\}, \emptyset, \emptyset, \emptyset) \leq k. \] (3.1)

Therefore, our aim is computing this cost. The naive dynamic programming approach is:
- compute \( \min_{K_x}(S) \) for each solution signature \( S \) and each leaf node \( x \),
- compute \( \min_{K_x}(S) \) for each solution signature \( S \) and each inner node \( x \) under the assumptions that \( \min_{K_y}(S') \) has already been computed for all solution signatures \( S' \) at children \( y \) of \( x \).

However, this approach is not suitable to prove Proposition 3.3, since the number of possible solution signatures is too large: the number of different partitions \( p \) of \( tw \) vertices is the \( tw \)-th Bell number, whose best known upper bound is \( O(tw^{tw}/\log tw) \).

### 3.1.3 Reducing the number of partitions

To reduce the number of needed partitions, we use an approach developed by Bodlaender et al. [6], which also proved its effectivity in experiments [18]. We will replace the task of computing (3.1) for all possibly partitions by computing only sets of weighted partitions containing the needed information.

**Definition 3.8 (sets of weighted partitions).** Let \( \Pi(U) \) be the set of all partitions of \( U \). A set of weighted partitions is a set \( A \subseteq \Pi(U) \times \mathbb{N} \). For a weighted partition \((p, w) \in A\), we call \( w \) its weight.

Using sets of weighted partitions, we can reformulate our task of computing \( \min_{K_x}(S) \) for all bags \( B_x \) and all solution signatures \( S \) as follows. Consider a pre-signature \( S = (D_x, D_i, D_e, N, l) \) for a node \( x \) of a tree decomposition. Then, for each \( p \in \Pi(D_x \cup D_i \cup D_e) \), \((S, p) \) is a solution signature. Thus, we can consider
\[
A_x(S) := \left\{ \left( p, \min_{p \in \mathcal{E}_x(S, p)} K(P) \right) \mid p \in \Pi(D_x \cup D_i \cup D_e) \land \mathcal{E}_x(S, p) \neq \emptyset \right\}. \quad (3.2)
\]

Now, our problem of verifying (3.1) at the root node \( r \) of a tree decomposition is equivalent to checking whether \( A_r(\emptyset, \{s, t\}, \emptyset, \emptyset, \emptyset) \) contains a partition \( \{\{s, t\}\} \) of weight at most \( k \). Thus, we can, in a classical dynamic programming manner
- compute \( A_x(S) \) for each pre-signature \( S \) and each leaf node \( x \),
- compute \( A_x(S) \) for each pre-signature \( S \) and each inner node \( x \) under the assumption that \( A_y(S') \) has already been computed for all pre-signatures \( S' \) at children \( y \) of \( x \).

Yet we will not work with the full sets \( A_x(S) \) but with “representative” subsets of size \( 2^{O(tw)} \).

Since the number of pre-signatures is \( 2^{O(tw)} \cdot l \), this will allow us to prove Proposition 3.3.

In order to describe the intuition behind representative sets of weighted partitions, we need some notation.

**Definition 3.9 (partition lattice).** The set \( \Pi(U) \) is semi-ordered by the coarsening relation \( \subseteq \), where \( p \sqsubseteq q \) if every set of \( p \) is included in some set of \( q \). We also say that \( q \) is coarser than \( p \) and that \( p \) is finer than \( q \).

For two partitions \( p, q \in \Pi(U) \), by \( p \sqcup q \) we denote the (unique) finest partition that is coarser than both \( p \) and \( q \).

To get an intuition for the \( p \sqcup q \) operation, recall from Definition 3.7 that we will use a partition \( p \) to represent connected components of partial solutions: two vertices are connected if and only if they are in the same set of \( p \). In these terms, if \( p \in \Pi(U) \) are the vertex sets
of the connected components of a graph \((U, E)\) and \(q \in \Pi(U)\) are the vertex sets of the connected components of a graph \((U, E')\), then \(p \cup q\) are the vertex sets of the connected components of the graph \((U, E \cup E')\).

Now, assume that there is a solution \(P\) to VW-SSP in a graph \(G\) and consider an arbitrary node \(x\) of a tree decomposition. Then, the subpaths \(\mathcal{P}\) of \(P\) that lie in \(G_x\) are a partial solution for some solution signature \((D_a, D_c, D_i, N, l, p)\) at \(x\). The partition \(p\) of \(D := D_a \cup D_c \cup D_i\) consists of the sets of vertices of \(D\) that are connected by paths in \(\mathcal{P}\). Since, in the overall solution \(P\), the vertices in \(D\) are all connected, the vertices of \(D\) are connected in \(G \setminus E_x\) according to a partition \(q\) of \(D\) such that \(p \cup q = \{D\}\). Now, if in \(P\), we replace the subpaths \(\mathcal{P}\) by any other partial solution \(\mathcal{P'}\) to a solution signature \((D_{a'}, D_{c'}, D_i, N, l, p')\) such that \(K(\mathcal{P'}) \leq K(\mathcal{P})\) and \(p' \cup q = \{D\}\), then we obtain a solution \(P'\) for \(G\) with at most the cost of \(P\). Thus, one of the two weighted partitions \((p, K(p))\) and \((p', K(p'))\) in \(A_x(D_a, D_c, D_i, N, l)\) is redundant.

This concept of redundancy can be formalized as representative sets and representative sets of size \(2^{O(tw)}\) can be efficiently computed using results of Bodlaender et al. [6]. To prove Proposition 3.3, it is enough to derive a recurrence relation for (3.2) that plays well together with the framework of Bodlaender et al. [6].

### 3.2 Hardness of kernelization for graphs of small treewidth

In the previous section, we have seen that SSP is efficiently solvable in tree-like graphs, namely, in graphs of small treewidth. We can complement this result as follows.

**Theorem 3.10 (⋆).** *Short Secluded Path has no problem kernel with size polynomial in \(tw + k + \ell\), even on planar graphs with maximum degree six, where \(tw\) is the treewidth, unless \(coNP \subseteq NP/poly and the polynomial-time hierarchy collapses to the third level.***

To prove Theorem 3.10, we use a special kind of reduction called cross composition [9].

**Definition 3.11 (cross composition).** A polynomial equivalence relation \(\sim\) is an equivalence relation over \(\Sigma^*\) such that

- there is an algorithm that decides \(x \sim y\) in polynomial time for any two instances \(x, y \in \Sigma^*\), and such that
- the number of equivalence classes of \(\sim\) over any finite set \(S \subseteq \Sigma^*\) is polynomial in \(\max_{x \in S} |x|\).

A language \(K \subseteq \Sigma^*\) cross-composes into a parameterized language \(L \subseteq \Sigma^* \times \mathbb{N}\) if there is a polynomial-time algorithm, called cross composition, that, given a sequence \(x_1, \ldots, x_p\) of \(p\) instances that are equivalent under some polynomial equivalence relation, outputs an instance \((x^*, k)\) such that

- \(k\) is bounded by a polynomial in \(\max_{x \in L} |x| + \log p\) and
- \((x^*, k) \in L\) if and only if there is an \(i \in \{1, \ldots, p\}\) such that \(x_i \in K\).

Cross compositions can be used to rule out problem kernels of polynomial size using the following result of Bodlaender et al. [9].

**Proposition 3.12 (Bodlaender et al. [9]).** *If a NP-hard language \(K \subseteq \Sigma^*\) cross-composes into the parameterized language \(L \subseteq \Sigma^* \times \mathbb{N}\), then there is no polynomial-size problem kernel for \(L\) unless \(coNP \subseteq NP/poly and the polynomial-time hierarchy collapses to the third level.*

Using a cross composition, Lackow and Fluschnik [37] proved that SSP on planar graphs of maximum degree six does not admit a problem kernel with size polynomial in \(k + \ell\). To prove Theorem 3.10, one can show that the graph created by their cross composition has treewidth at most \(3n + 3\), where \(n\) is the number of vertices in each input instance to their cross composition.
3.3 Effective data reduction for graphs with small feedback edge set

In the previous section, we have seen that SSP has no problem kernel with size polynomial in the treewidth of the input graph. We can complement this result by proving a polynomial-size problem kernel for another parameter that measures the tree-likeness of a graph: the feedback edge number of a graph is the smallest number of edges one has to delete to obtain a forest. Formally, we can prove the following theorem.

**Theorem 3.13 (⋆).** Short Secluded Path has a problem kernel with size polynomial in the feedback edge number of the input graph.

The outline of the proof of Theorem 3.13 is similar to that of the proof of Theorem 2.5: we first, in linear time, produce a weighted instance with $O(fes)$ vertices, then reduce the weights using Lemma 2.13, and finally transform the weighted instance back into an instance for SSP.

Towards the first step, we apply data reduction rules that reduce the number of degree-one vertices and the length of paths of degree-two vertices to $O(fes)$. Because of Reduction Rule 1.4, the graph without the $fes$ edges of a feedback edge set is a tree. Thus, its overall number of vertices and edges will be bounded by $O(fes)$.

4 Conclusion

Concluding, we point out that our algorithms for VW-SSP on graphs of bounded treewidth (Theorem 3.2) can easily be generalized to a problem variant where also edges have a weight counting towards the path length, and so can our subexponential-time algorithms in planar graphs (Theorem 2.1). Moreover, the technique of Bodlaender et al. [6] that our algorithm is based on has experimentally been proven to be practically implementable [18].

In contrast, we observed SSP to be a problem for which provably effective polynomial-time data reduction is rather hard to obtain (Theorems 2.14 and 3.10). Therefore, studying relaxed models of data reduction with performance guarantees like approximate [36, 19] or randomized kernelization [35] seems worthwhile.

Indeed, our few positive results on kernelization, that is, our problem kernels of size $vcO(r)$ in $K_{r,r}$-free graphs and of size $O(1)$ in graphs of feedback edge number $fes$ for SSP (Theorems 2.5 and 3.13), for now, can be mainly seen as a proof of concept, since they employ the quite expensive weight reduction algorithm of Frank and Tardos [22] and we have no “direct” way of reducing VW-SSP back to SSP. On the positive side, our solution algorithms also work for VW-SSP, so that reducing weights or reducing back to SSP may be unnecessary from a practical point of view.

References


10:18 Parameterized Algorithms and Data Reduction for Safe Convoy Routing


