Noname manuscript No.
(will be inserted by the editor)

The Parameterized Complexity of Local Search for TSP, More Refined

Jiong Guo · Sepp Hartung · Rolf Niedermeier · Ondřej Suchý

the date of receipt and acceptance should be inserted later

Abstract We extend previous work on the parameterized complexity of local search for the Traveling Salesperson Problem (TSP). So far, its parameterized complexity has been investigated with respect to the distance measures (defining the local search area) "Edge Exchange" and "Max-Shift". We perform studies with respect to the distance measures "Swap" and "r-Swap", "Reversal" and "r-Reversal", and "Edit", achieving both fixed-parameter tractability and W[1]-hardness results. In particular, from the parameterized reduction showing W[1]-hardness we infer running time lower bounds (based on the exponential time hypothesis) for all corresponding distance measures. Moreover, we provide non-existence results for polynomial-size problem kernels and we show that some in general W[1]-hard problems turn fixed-parameter tractable when restricted to planar graphs.

Keywords NP-hard problem, heuristics, problem kernel, fixed-parameter tractability, W[1]-hardness, lower bounds based on ETH

1 Introduction

The Traveling Salesperson Problem (TSP) is probably the most studied combinatorial optimization problem. Almost all algorithm design techniques have been applied to it or

The first and the fourth author were supported by the DFG Cluster of Excellence on Multimodal Computing and Interaction (MMCI) and the DFG project DARE (GU 1023/1-2). Most of the work of Ondřej Suchý was done while he was with MMCI Saarbrücken.
An extended abstract of this paper appeared in *Proceedings of the 22nd International Symposium on Algorithms and Computation (ISAAC'11)*, Yokohama, Japan, Dec. 2011, volume 7074 of LNCS, pages 614–623, Springer, 2011. An important difference compared to the conference version is that we significantly improved the running time lower bounds.

Jiong Guo
Universität des Saarlandes, Cluster of Excellence, Multimodal Computing and Interaction, Saarbrücken, Germany
E-mail: jguo@mmci.uni-saarland.de

Sepp Hartung, Rolf Niedermeier and Ondřej Suchý
TU Berlin, Institut für Softwaretechnik und Theoretische Informatik, Berlin, Germany
E-mail: sepp.hartung@tu-berlin.de, rolf.niedermeier@tu-berlin.de, ondrej.suchy@tu-berlin.de

were even specifically developed for it [21, 29]. Famous results include the Held/Karp-algorithm [24], the polynomial-time factor-1.5 approximation algorithm for Metric TSP of Christofides [8], and the polynomial-time approximation scheme for Euclidean TSP [1]. Many heuristic algorithms for TSP have been developed and evaluated [26, 27]. Most of them follow the paradigm of *local search*: Incrementally try to improve a solution by searching within its local neighborhood defined by a *distance measure*. Perhaps the most prominent and best examined distance measure for TSP is the *k-Edge Exchange* neighborhood (also called *k-Opt neighborhood* in some literature), where one is allowed to exchange at most k edges of the Hamiltonian cycle forming the tour. Implementations of this strategy for $k = 2, 3$ and generalizations such as the Lin-Kernighan-heuristic belong to the best performing heuristic algorithms for real-world instances [27] both in terms of quality and in terms of running time. However, for larger k, for which one would expect a strong increase of quality, the running time becomes infeasible since until now no algorithm is known which significantly beats the trivial $O(n^k)$ running time needed for a brute-force exploration of the local distance-k neighborhood on n-vertex graphs. In an important step forward, considering the problem within the framework of *parameterized complexity* [11, 16, 35], Marx [32] has shown by proving W[1]-hardness that there is no hope for an algorithm running in $f(k) \cdot n^c$ time for any function f (solely depending on k) and any constant c. Note that such an algorithm is desirable since the degree of the polynomial in the input size n does not depend on the parameter k. Moreover, assuming that the ETH (*exponential time hypothesis*) [25] does not fail, Marx [32] has shown that there is no algorithm running in $O(n^{o(\sqrt[3]{k})})$ time.

In this work, besides the k-Edge Exchange neighborhood (briefly, *Edge* distance measure), we consider various other distance measures such as the *Reversal* distance where the order of some consecutive vertices is reversed,[1] the *Swap* distance where one is allowed to exchange two vertices, and the *Edit* distance where one can move a vertex to an arbitrary new position. For λ being any of these distance measures, we study the following problem.

LocalTSP(λ)
Input: An undirected graph $G = (V, E)$ with vertices labeled $v_1, \ldots, v_n$ such that the identical permutation (id) $v_1, v_2, \ldots, v_n, v_1$ is a Hamiltonian cycle in G, an edge weight function $\omega : E \to \mathbb{R}_0^+$, and a positive integer k.
Question: Is there a permutation π with $\lambda(\pi, \mathrm{id}) \leq k$ that yields a Hamiltonian cycle with $\omega(\pi) < \omega(\mathrm{id})$, where $\omega(\pi) := \sum_{i=1}^{n-1} \omega(\{v_{\pi(i)}, v_{\pi(i+1)}\}) + \omega(\{v_{\pi(n)}, v_{\pi(1)}\})$?

Reflecting different distance measures λ, we speak about LocalTSP(Edge), LocalTSP(Reversal), LocalTSP(Swap), LocalTSP(Edit), etc. We use LocalTSP if the measure in use is not important. Notably, all problems have the same set of instances.

Our Results. Table 1 summarizes our results. We show that the W[1]-hardness result of Marx [32] for LocalTSP(Edge) can be extended, that is, we show that LocalTSP(λ) for $\lambda \in \{\text{Swap, Edit, Reversal}\}$ is also W[1]-hard, implying that it is probably not fixed-parameter tractable for the "locality parameter" k. Furthermore, we strengthen Marx's running time lower bound based on the ETH by showing that LocalTSP(λ)

[1] The reversal distance is also widely studied in bioinformatics in the context of genome rearrangements [6, 15].

	r-Swap	Swap	Edit	r-Reversal	Reversal	Edge
general graphs	FPT (Thm. 3.3)	W[1]-h (Thm. 3.1)	W[1]-h (Thm. 3.1)	FPT (Thm. 3.4)	W[1]-h (Thm. 3.1)	W[1]-h [32]
planar graphs	*FPT*	FPT∗ (Thm. 4.2)	FPT∗ (Thm. 4.2)	*FPT*	?	?

Table 1.1 Overview of our results using k as the parameter and assuming r to be a constant. The two results written in italics are a direct consequence of a more general result. Furthermore, unless NP $\subseteq$ coNP/poly, we show that for all distance measures above there cannot be a polynomial-size problem kernel even on planar graphs (Theorem 4.1). Results marked by $*$ indicate that the corresponding algorithm is only a permissive FPT algorithm (see Section 2).

for $\lambda \in \{\text{Swap, Edit, Reversal, Edge}\}$ does not admit an algorithm with running time $O(n^{o(k/\log k)})$.

Additionally, for the Swap distance we show that, restricting by a parameter r the distance of two vertices that are allowed to swap, makes LocalTSP(r-Swap) fixed-parameter tractable with respect to the combined parameter (k, r). Specifically, we outline an algorithm running in $O(r^{2k}(r-1)^{2k} \cdot 4^k \cdot (k^2+n) \cdot n)$ time. Furthermore, we show that an analogously restricted Reversal distance, called r-Reversal, admits an algorithm running in $O(2^{rk} \cdot r^{2k-1} \cdot (r-1)^k \cdot (k^2+rk+n) \cdot n)$ time and thus again leads to fixed-parameter tractability. Continuing to chart the border of tractability, we show that LocalTSP(λ) for $\lambda \in \{\text{Swap, Edit}\}$ is fixed-parameter tractable on planar graphs. In addition, exploring the limitations of polynomial-time preprocessing, we indicate that, unless NP $\subseteq$ coNP/poly, even on planar graphs there is no polynomial-size problem kernel for LocalTSP(λ) for any of the considered distance measures λ.

Related Work. The most important reference point to our work is Marx's study of LocalTSP(Edge) [32] (using different notation). Long before Marx, Balas [2] studied LocalTSP(Max-Shift), where Max-Shift distance k means that in order to obtain an improved Hamiltonian cycle the maximum number of positions that a vertex is allowed to shift is k. Contrasting the parameterized hardness result of Marx [32], Balas showed that LocalTSP(Max-Shift) is fixed-parameter tractable by providing an algorithm running in $O(4^{k-1}k^{1.5}n)$ time.

Local Search in Parameterized Algorithmics. It is very natural to use parameterized algorithmics to study the computational complexity of local search measured in the size of the local neighborhood where one tries to find an improved solution. In fact, when k measures the "diameter" of the local neighborhood, it is often not hard to come up with an algorithm running in $n^{O(k)}$ time, but since such an algorithm usually becomes intractable already for very small k, the question whether there is an algorithm with running time $f(k) \cdot n^{O(1)}$ for a moderately growing function f naturally arises. Parameterized algorithmics provides a framework to prove the existence of such algorithms or to deliver some evidence that it cannot exist.

We will briefly summarize the state of the art on parameterized results for local search. Fellows et al. [14] showed that searching the k-exchange neighborhood for problems such as r-Center, Vertex Cover, Odd Cycle Transversal, Max-Cut, and Min-Bisection can be done on planar graphs in $2^{O(k)} \cdot n^2$ time, and is W[1]-hard on general graphs. Fomin et al. [17] outlined a color-coding based algorithm for Weighted Feedback Arc Set in Tournaments that decides in $O(2^{o(k)} \cdot n \log n)$ time whether there is an improved solution in the k-exchange neighborhood (symmetric difference

of the corresponding arc sets). Marx and Schlotter [34] studied a variant of the Stable Marriage problem with respect to local search in the framework of parameterized algorithmics. To analyze local search in the framework of parameterized algorithmics is relatively new; further applications include Boolean Constraint Satisfaction [28], Incremental Coloring [23], Sat [39], and Vertex Cover [19].

2 Basic Notation and Distance Measures

Notation. Let $\mathcal{S}_n$ denote the set of all permutations on $\{1, \ldots, n\}$ and let $\mathrm{id} \in \mathcal{S}_n$ be the identity. If not otherwise stated, we consider undirected simple graphs $G = (V, E)$ with *vertex set* V and *edge set* E. We set $V(G) := V$, $E(G) := E$, $n := |V|$, and $m := |E|$. A directed graph $G = (V, A)$ consists of a vertex set V and an arc set A. A Hamiltonian cycle through a graph $G = (V, E)$ with vertices labeled $v_1, v_2, \ldots, v_n$ is expressed by a permutation $\pi \in \mathcal{S}_n$ such that the edge set $E(\pi)$ of π, defined as $E(\pi) = \{\{v_{\pi(i)}, v_{\pi(i+1)}\} \mid 1 \leq i < n\} \cup \{\{v_{\pi(n)}, v_{\pi(1)}\}\}$, is a subset of E. Clearly, in case of a directed graph $G = (V, A)$ we require that A contains $(v_{\pi(i)}, v_{\pi(i+1)})$ for all $i < n$ and $(v_{\pi(n)}, v_{\pi(1)})$. For a weight function $\omega : E \to \mathbb{R}_0^+$ we define the weight of π by $\omega(\pi) = \sum_{e \in E(\pi)} \omega(e)$. The Hamiltonian cycle π is called *improved* compared to id when $\omega(\pi) < \omega(\mathrm{id})$. In this sense, LocalTSP(λ) is the question whether there is an improved Hamiltonian cycle π with $\lambda(\pi, \mathrm{id}) \leq k$.

Parameterized Algorithmics. A parameterized problem is said to be *fixed-parameter tractable* if there is an algorithm that solves every instance (I, k) (where k is the parameter) within $f(k) \cdot |I|^c$ time for a constant c and a function f which solely depends on k [11, 16, 35]. A recent development extends parameterized complexity analysis into a multivariate complexity analysis where multiple parameters are combined [12, 36]. A *kernelization algorithm* computes for a given instance (I, k) in polynomial time a new instance (I', k') (called *kernel*) such that (I', k') is a yes-instance if and only if (I, k) is a yes-instance, $k' \leq g(k)$, and $|I'| \leq g(k)$ for a function g which solely depends on k [4, 20, 31]. The function g measures the size of the kernel. If g is a polynomial function, then the kernel is called *polynomial-size kernel*.

The basic class of parameterized intractability is W[1]. A parameterized problem shown to be W[1]-hard by means of a *parameterized reduction* from another W[1]-hard problem is believed not to be fixed-parameter tractable. For two parameterized problems L and L', a *parameterized reduction* from L to L' maps any instance (I, k) of L in $f(k) \cdot |I|^{O(1)}$ time for some function f to an instance (I', k') of L' such that $k' \leq g(k)$ for some function g and $(I, k) \in L \Leftrightarrow (I', k') \in L'$.

It has been shown that, unless the exponential time hypothesis (ETH)[2] fails, Clique (deciding whether a graph contains a complete graph/clique of size k) cannot be solved in $f'(k) \cdot n^{o(k)}$ time for any function f' [7]. Thus, providing a parameterized reduction from Clique to a parameterized problem L, not only proves that L is W[1]-hard but also shows that, unless the ETH fails, L cannot be solved in $f'(k) \cdot n^{o(g^{-1}(k))}$ time for any function f', where g is the function from the corresponding parameterized reduction; refer to a recent survey [30] for more details on lower bounds based on the ETH.

[2] Roughly speaking, the ETH states that 3-Sat cannot be solved in subexponential time.

Permissive Algorithms. Marx and Schlotter [34] proposed to distinguish between *strict* and *permissive* local search algorithms. Strict local search algorithms find an improved solution (or prove that it does not exist) within some limited distance from the given solution. Permissive local search algorithms find *any* improved solution (potentially, with unbounded distance to the given solution), provided that an improved solution exists within the limited distance of the given solution. The motivation for this distinction is that finding an improved solution within a bounded distance of a given solution may be hard even for problems where an optimal solution can easily be found, e. g., Minimum Vertex Cover on bipartite graphs [28]. Gaspers et al. [19] recently showed for Vertex Cover that for a class of inputs strict local search is hard while permissive local search is tractable.

Distance Measures. So far, the distance between Hamiltonian cycles was usually measured in terms of *Edge* distance, counting the number of edges used by one cycle but not used by the other. Another measure considered is the *Max-Shift* distance, which equals the maximum shift of the position of a vertex between the two permutations [2]. We consider several further measures based on the following operations on permutations.

Definition 2.1 For a permutation $1, 2, \ldots, n$, we define the following operations:

$$\begin{aligned} \textit{reversal } \rho(i,j) &\text{ results in } 1,\ldots,i-1,j,j-1,\ldots,i+1,i,j+1,\ldots,n;\\ \textit{swap } \sigma(i,j) &\text{ results in } 1,\ldots,i-1,j,i+1,\ldots,j-1,i,j+1,\ldots,n;\\ \textit{edit } \epsilon(i,j) &\text{ results in } 1,\ldots,i-1,i+1,\ldots,j-1,j,i,j+1,\ldots,n. \end{aligned}$$

For a constant r a swap $\sigma(i,j)$ (or a reversal $\rho(i,j)$) is called an *r-swap* (r-reversal, resp.) if $0 < j-i \leq r-1$ or $n+j-i \leq r-1$. The distance measures *Swap, r-Swap, Edit, Reversal*, and *r-Reversal* count the minimum number of the appropriate operations to apply to one permutation in order to obtain the other.

We do not consider the elements 1 and n to be anyhow special and, therefore, the operations above can also be applied "over them", e. g. $\sigma(n-1, 2)$ is a 4-swap.

We next show how the relation between the distance measures from Definition 2.2 can be used to easily transfer results shown for one distance measure to other ones.

Definition 2.2 A distance measure λ is *bounded* by a distance measure τ (or τ-bounded) if there is a function $f : \mathbb{N} \to \mathbb{N}$ such that for any two permutations $\pi, \pi' \in \mathcal{S}_n$ it holds that $\lambda(\pi, \pi') \leq f(\tau(\pi, \pi'))$.

The relation of boundedness is reflexive and transitive and, therefore, forms a quasi-order on the distance measures. Figure 2.1 depicts all relations between the measures, omitting relations that can be deduced from the transitivity, in this sense showing a Hasse diagram of this quasi-order. Before arguing about correctness of the relations depicted in Figure 2.1, we show a tight relationship between our notion of bounded distance measures and the existence of permissive FPT-algorithms.

Lemma 2.1 *If a distance measure λ is τ-bounded, then a (permissive) FPT-algorithm for* LocalTSP*(λ) is a permissive FPT-algorithm for* LocalTSP*(τ).*

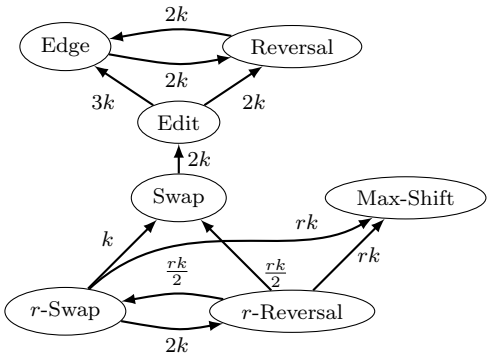

Fig. 2.1 Hasse diagram of the relations between the distance measures. Let $f : \mathbb{N} \to \mathbb{N}$. An arrow labeled "$f(k)$" from a distance measure τ to a measure λ means that λ is τ-bounded with function $f(k)$, implying that two Hamiltonian cycles of distance k with respect to τ have distance at most $f(k)$ with respect to λ.

Proof Consider an instance of LocalTSP(τ) with an improved Hamiltonian cycle in τ-distance at most k from the given Hamiltonian cycle. Then this improved Hamiltonian cycle is in λ-distance at most $f(k)$ from the given Hamiltonian cycle. Thus, running the (permissive) FPT-algorithm for LocalTSP(λ) with parameter $f(k)$ returns an improved Hamiltonian cycle and thus is a permissive algorithm for LocalTSP(τ). □

We next argue about the correctness of the relations depicted in Figure 2.1 (we consider r to be a constant in these comparisons): Obviously, an r-swap is a special case of a swap and, therefore, Swap distance is bounded by r-Swap distance. Next, an r-swap can be simulated by at most two r-reversals and a swap can be simulated by two edits. Thus, the r-Reversal distance is bounded by the r-Swap distance and the Edit distance is Swap-bounded. Further, one r-swap or r-reversal shifts a position of any vertex in the Hamiltonian cycle by at most r, and, therefore, k of them shift no vertex by more than rk, which implies that Max-Shift distance is both r-Swap-bounded and r-Reversal-bounded. Similarly an r-reversal can be simulated by at most $r/2$ r-swaps and, hence, r-Swap distance is r-Reversal-bounded. Since one edit can be simulated by at most two reversals and a reversal breaks at most two edges, it follows that Reversal distance is Edit-bounded and Edge distance is Reversal-bounded. Additionally, an edit breaks at most three edges and thus the Edge distance is Edit-bounded. It remains to show that Reversal distance is Edge-bounded.

Lemma 2.2 *The Reversal distance is bounded by the Edge distance.*

Proof Assume that from a given Hamiltonian cycle one can obtain an improved one by first deleting k edges and adding another k edges. Consider the paths of the given Hamiltonian cycle after we remove k edges. Now we build the improved Hamiltonian cycle by gradually connecting the appropriate paths. We start from any path and consider the path that should come next. By two reversals we can achieve that the paths follow each other in the correct order (the first reversal moves the path next to the previous one and the second rotates it into the right direction). By this we introduce at least one edge of the new Hamiltonian cycle. As we never break any edge not deleted from the Hamiltonian cycle and the reversals can be taken to operate outside the already

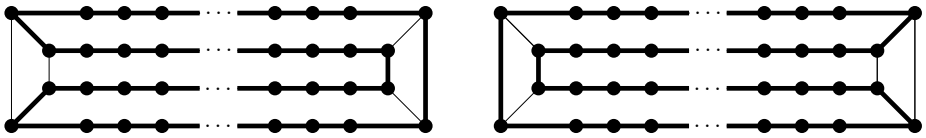

Fig. 2.2 A planar graph with two different Hamiltonian cycles (marked by bold lines). The cycles are only four edge modifications and four reversals from each other, while they can be made arbitrarily far apart for any other of the measures by extending the horizontal lines. Furthermore, the vertices affected by the changes are also arbitrarily large apart from each other in the underlying graph and, as the graph has no other Hamiltonian cycles, there is no other solution with changes concentrated in a constant distance to one particular vertex.

built part of the Hamiltonian cycle, we build the whole improved Hamiltonian cycle by at most $2k$ reversals. This shows that Reversal distance is bounded by Edge distance. □

It is also not hard to come up with examples showing that no further boundedness relations hold between the distance measures. See Figure 2.2 for an interesting case of two Hamiltonian cycles which are close for Reversal and Edge distances, but far apart for all the other distances considered.

3 General Graphs

In this section, we provide parameterized hardness as well as fixed-parameter tractability for LocalTSP using various distance measures.

3.1 W[1]-Hardness and an $O(n^{o(k/\log k)})$ Lower Bound

We show that LocalTSP(λ) is W[1]-hard for $\lambda \in \{$Swap, Edit, Reversal$\}$. Furthermore, for all these distance measures plus the Edge distance we provide a computational lower bound of $O(n^{o(k/\log k)})$. To this end, we build on the W[1]-hardness proof for LocalTSP(Edge) by Marx [32]. In contrast to Marx, who gave a parameterized reduction from the k-Clique problem, we reduce from the Partitioned Subgraph Isomorphism problem. This makes the construction more structured and more powerful.

> Partitioned Subgraph Isomorphism (PSI)
> **Input:** Two undirected graphs H and G with $|V(H)| \leq |V(G)|$, and a (not necessarily proper) coloring $f : V(G) \to V(H)$ of vertices of G with vertices of H.
> **Question:** Is there a mapping $h : V(H) \to V(G)$ such that $\forall v \in V(H) : f(h(v)) = v$ and h is a homomorphism, that is, $\forall \{u, v\} \in E(H) : \{h(u), h(v)\} \in E(G)$?

If such a homomorphism h exists, then we say that there is a *colored H-subgraph in G*. PSI is W[1]-hard for the parameter $k := |E(H)|$ as it is a generalization of the W[1]-hard k-Multicolored Clique problem [13].

Our main result in this section is the following.

Theorem 3.1 LocalTSP(λ) *is W[1]-hard with respect to k for $\lambda \in \{$Swap, Edit, Reversal, Edge$\}$.*

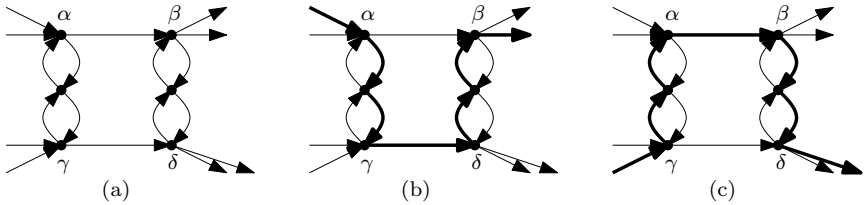

Fig. 3.1 Switch gadget (a) and the two possible ways to traverse it (b) and (c)

Theorem 3.1 follows from the following lemma, the fact that PSI is W[1]-hard and from that the Edit, Reversal, and Edge distances are all Swap-bounded.

Lemma 3.1 *There is a parameterized reduction from* Partitioned Subgraph Isomorphism *parameterized by* $k := |E(H)|$ *to* LocalTSP*(Swap) such that any improved Hamiltonian cycle can be obtained by performing at most* $O(k)$ *swaps.*

Proof We provide a parameterized reduction from PSI parameterized by $k := |E(H)|$ to DirectedLocalTSP(Swap), this is, the variant of LocalTSP(Swap) where the input graph is directed. We show that in the constructed graph any improved Hamiltonian cycle can be obtained by performing at most $24k$ swaps to the given cycle. The claim is then obtained from the parameterized reduction from DirectedLocalTSP(Swap) to LocalTSP(Swap) given by Marx [32, Lemma 3.1].

Construction: Assume that the graph $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$, the graph H with $k = |E(H)|$ and $V(H) := \{1, \ldots, l\}$, and a coloring f constitute an instance of PSI. We assume without loss of generality that H is connected. We construct an equivalent instance of DirectedLocalTSP(Swap) on the directed graph D by multiple copies of the so-called *switch* gadget (see Figure 3.1(a)). There are only two possibilities to traverse a switch on a Hamiltonian cycle, by using either the *upper path* $\alpha \to \beta$ (Figure 3.1(b)) or the *lower path* $\gamma \to \delta$ (Figure 3.1(c)). Next, since the constructed graph D will contain only one non-zero-weight arc, all arcs have weight zero if not explicitly stated otherwise.

Each vertex $v_i \in V$ is represented in D by its *segment* V_i, which is formed by $\deg_H(f(v_i))$ many switch gadgets $V_{i,j}$, where $j \in \{1, \ldots, \deg_H(f(v_i))\}$. To form the segment, we sequentially connect the switches by connecting each β-vertex to the α-vertex of the subsequent switch. Furthermore, we add a start vertex v_i^s which has an outgoing arc to the α-vertex of the first switch and an end vertex v_i^e which has an incoming arc from the β-vertex of the last switch.

We connect all segments $V_1, V_2, \ldots, V_n$ by adding an arc from the end vertex v_i^e of segment V_i to the start vertex v_{i+1}^s of segment V_{i+1} for all $1 \leq i < n$. In addition, we add a start vertex v^s which has an outgoing arc to the start vertex v_1^s of the first segment V_1 and an end vertex v^e which has an incoming arc from the end vertex v_n^e of the last segment V_n.

For each color $j \in V(H)$ we add a so-called *template* t_j, which is a directed simple path consisting of $6 \deg_H(j)$ vertices. Furthermore, for all segments V_i where $f(v_i) = j$ there is an arc from v_i^s to the first template vertex t_j^s and an arc from the last template vertex t_j^e to v_i^e. Finally, there are arcs from v^e to t_1^s, from t_j^e to t_{j+1}^s for all $j \in V(H)$, and an arc of weight one from the last vertex in the last template t_l^e to v^s. This allows to traverse all templates starting from the end vertex v^e, and after reaching t_l^e we can go back to the start vertex v^s.

The given Hamiltonian cycle C in our DirectedLocalTSP(Swap) instance on D is as follows. It starts in v^s, traverses each segment by using the upper path for each switch, follows the zero-weight arcs between subsequent segments, ends up in v^e, afterwards traverses all templates and finally uses the weight-one arc from t_l^e to get back to the start vertex v^s. Hence, the weight of cycle C is one.

Observe that, once a cycle enters a segment V_i via v_i^e and begins to traverse the first switch by using the upper path, it has to traverse all switches in this segment by the upper path. We call such a segment *passive*. Symmetrically, if one switch is traversed by the lower path, all switches in the same segment have to be traversed by the lower path, and we call such a segment *active*. All segments on the cycle C are passive and, so far, it is the only possible Hamiltonian cycle through D.

Now, by adding some further arcs to D we will "encode" the structure of G to ensure the existence of an alternative cycle C' with weight strictly less than $w(C) = 1$ if and only if there is a colored H-subgraph in G. The only possibility to get a cycle with weight less than $w(C) = 1$ is to skip the weight-one arc (t_l^e, v^s). The idea is that the cycle C' starts in v^s, follows the order of the segments but (in distinction to C) "decides" for each segment V_i whether it traverses the segment through the switches or it "skips" the segment by using the template $t_{f(v_i)}$. After the cycle reaches v^e, the new arcs allow C' to traverse all skipped segments by using the lower path for each switch, and in this way all these switches become active. The vertices in G which correspond to the active segments in D on C' are intended to form a colored H-subgraph.

For the purpose of a formal description of the new arcs in D which are necessary to traverse the active segments starting at v^e, consider the directed graph H' obtained from H by replacing each edge by two arcs with opposite directions. Further, consider a closed Eulerian cycle $p' := p_0, p_1, p_2 \ldots, p_{2k-1}, p_{2k}$ in H', where $p_i \in V(H') = V(H)$, $p_0 = p_{2k}$, and $(p_{i-1}, p_i) \in A(H')$ for every $i \in \{1, \ldots, 2k\}$. We remove p_0 from p' to obtain $p := p_1, \ldots, p_{2k}$. In the trail p, we define $b(p_i) := |\{i' \mid i' \leq i \text{ and } p_{i'} = p_i\}|$ to count how many times vertex p_i already appeared on p when we reach p_i (we have $b(p_i) \in \{1, \ldots, \deg_H(p_i)\}$).

We want our new cycle C' to follow the Eulerian trail p, using the $b(p_i)$-th switch of the segment of some vertex of color p_i in the i-th place. To allow this, we add arcs between the segments as follows. If there is an edge $\{v_i, v_j\} \in E(G)$ and the arc $(f(v_i), f(v_j))$ appears in p as $(p_\phi, p_{\phi+1})$, then add into D the arc from the δ-vertex of the switch $V_{i,b(p_\phi)}$ to the γ-vertex of switch $V_{j,b(p_{\phi+1})}$. To complete the construction of D, connect this structure to the rest of the graph by adding for each vertex v_i with $f(v_i) = p_1$ an arc from v_e to the γ-vertex of switch $V_{i,1}$ and by adding for each vertex v_j with $f(v_j) = p_{2k}$ an arc from the δ-vertex of the switch $V_{j,b(p_{2k})}$ to the vertex v^s.

Correctness: We show that there is a Hamiltonian cycle C' of weight zero within Swap-distance at most $24k$ of C if and only if there is a colored H-subgraph in G. Furthermore, we show that any improved Hamiltonian cycle in D can be obtained by performing at most $24k$ swaps.

"$\Leftarrow$": Assume that the vertices $v_{i_1}, v_{i_2}, \ldots, v_{i_l} \in V(G)$ form a colored H-subgraph in G such that $f(v_{i_j}) = j$ for all $j \in V(H)$. Then there is a cycle C' that, starting in v^s, uses the template t_j instead of segment V_{i_j}, continues after reaching v^e to the γ-vertex of switch $V_{i_1,1}$, then mimics the trail p and finishes by using the arc from the δ-vertex of switch $V_{i_l,b(p_{2k})}$ to vertex v^s. Note that each switch is traversed exactly once. Moreover, for each $j \in V(H)$ there is exactly on active segment V_i corresponding to the vertex $v_i \in V(G)$ with $f(v_i) = j$.

It remains to count the number of swaps that have to be performed to C to get C'. To this end, observe that the template of color j as well as the switches of a segment (without start and end vertex) of a vertex of this color consists of exactly $6\deg_H(j)$ vertices each. Since activating a segment is done by swapping the vertices in the corresponding switches with the vertices in a template, we only need $6\deg_H(j)$ swaps to activate a segment of color j. In total we need exactly $\sum_{j\in V(H)} 6\deg_H(j) = 12k$ swaps to activate all segments. Finally, we need at most $12k$ swaps to "sort" the vertices in the active segments such that the order of the switches corresponds to the trail p. We can conclude that the Hamiltonian cycle C' can be obtained from C by performing at most $24k$ swaps.

"⇒": For the reverse direction, assume that in D there is a cycle C' of weight zero. We prove that C' is within Swap-distance $24k$ from C and that there is a colored H-subgraph in G. As cycle C' has to be different from C, at least one segment V_u has to be active in it. Call a color j *active* if there is a vertex $v_i \in V$ with $f(v_i) = j$ such that the segment V_i is active and *passive* otherwise.

We first show that in C' all colors are active. Towards a contradiction assume there is a passive color x. Then there must be two colors x' and y' such that x' is passive, y' is active, and $\{x', y'\} \in E(H)$, since there is at least one active color and H is connected. The trail p uses (y', x') or (x', y'). Consider the first case, that is, $y' = p_i$ and $x' = p_{i+1}$ for some $i \in \{1, \ldots, 2k-1\}$. Also suppose that $V_{u'}$ is an active segment with $f(v_{u'}) = y'$. Due to our construction, the only way how the cycle C' can leave the switch $V_{u',b(p_i)}$ is that there is an active segment corresponding to a vertex of color x', which is a contradiction. As the second case also leads to a contradiction in a similar way, we get that every color is active.

On the other hand, there is at most one active segment of each color, as the only way to make a segment active is to replace it by the template. Hence, for each color j there is exactly one vertex v_{i_j} such that the segment V_{i_j} is active. Since, in order to traverse these active segments, the Hamiltonian cycle C' has to follow the Eulerian trail p, this enforces that the vertices v_{i_j} and $v_{i_{j'}}$ are adjacent in G whenever $\{j, j'\} \in E(H)$ and thus v_{i_j}'s form a colored H-subgraph in G.

Moreover, since the improved Hamiltonian cycle C' contains exactly one active segment for each vertex in $V(H)$, it follows that C' is within Swap-distance $24k$ of C: As already argued above, replacing $|V(H)|$ many segments in the cycle C by the corresponding templates and then sorting the switches within these active segments according to the Eulerian trail p is possible by at most $24k$ swaps. □

We next show the running time lower bound that can be derived from Lemma 3.1. To this end, the following theorem of Marx [33], proving a lower bound for PSI, is extremely useful.

Theorem 3.2 ([33, Corollary 6.3]) *Unless the ETH fails,* Partitioned Subgraph Isomorphism *cannot be solved in $f(H) \cdot n^{o(k/\log k)}$ time, where f is an arbitrary function and $k := |E(H)|$.*

Lemma 3.1 together with Theorem 3.2 implies the following corollary. For the case of Edge distance, it improves the lower bound $O(n^{o(\sqrt[3]{k})})$ given by Marx [32, Corollary 3.5].

Corollary 3.1 *Unless the ETH fails,* LocalTSP(λ) *does not admit a (permissive) algorithm with running time $O(n^{o(k/\log k)})$ for $\lambda \in \{$Swap, Edit, Reversal, Edge$\}$.*

3.2 Tractability

In Section 3.1 we have shown that on general graphs LocalTSP(λ) for $\lambda \in \{$Swap, Edit, Reversal, Edge$\}$ is W[1]-hard. In this section we show that LocalTSP becomes fixed-parameter tractable when using the more restrictive distance measures r-Swap and r-Reversal instead of Swap and Reversal, respectively. Actually, we prove a stronger result, that is, fixed-parameter tractability with respect to the combined parameter (k, r). The corresponding algorithms are based on the bounded search tree technique, and they are mainly based on the observation that the solution can be assumed to be given by a sequence of r-swaps (or r-reversals, resp.) that are somehow related.

We first describe the algorithm for LocalTSP(r-Swap). To this end, we need the following definition. Let S be a sequence of swaps. We define an undirected auxiliary *swap graph* G^S as follows. There is a vertex for each swap in the sequence S, and two swaps $\sigma(i,j)$ and $\sigma(t,l)$ are adjacent if either t or l is contained in $\{i-1, i, i+1, j-1, j, j+1\}$. Furthermore, if a swap $\sigma(i,j)$ is applied, then we call the positions i and j and the vertices at these positions *affected*.

Lemma 3.2 *If a* LocalTSP*(λ) instance for $\lambda \in \{r$-Swap, Swap$\}$ admits an improved Hamiltonian cycle, it also admits an improved Hamiltonian cycle which can be obtained by swaps (or r-swaps) such that their swap graph is connected.*

Proof Suppose that we are given a sequence S of swaps whose application to the Hamiltonian cycle $\mathrm{id} \in \mathcal{S}_n$ creates an improved Hamiltonian cycle $\pi \in \mathcal{S}_n$. Towards a contradiction, assume that $C_1, \ldots, C_p$ with $p \geq 2$ are the connected components of the corresponding swap graph G^S. For any of these components C, we denote by $\pi^C \in \mathcal{S}_n$ the permutation that results from applying the swaps in C to id preserving their order relative to S.

We shall show that the sets $E(\pi^{C_1}) \triangle E(\mathrm{id}), \ldots, E(\pi^{C_p}) \triangle E(\mathrm{id})$ form a partition of the set $E(\pi) \triangle E(\mathrm{id})$ ($\triangle$ denotes the symmetric difference). Having proved this, the rest of the argumentation is as follows. Since $\omega(\pi) < \omega(\mathrm{id})$ or equivalently $\omega(E(\pi) \setminus E(\mathrm{id})) < \omega(E(\mathrm{id}) \setminus E(\pi))$, it follows that there is at least one component C of G^S with $\omega(E(\pi^C) \setminus E(\mathrm{id})) < \omega(E(\mathrm{id}) \setminus E(\pi^C))$. This implies that $\omega(\pi^C) < \omega(\mathrm{id})$ and thus applying only swaps contained in C also results in an improved Hamiltonian cycle π^C.

It remains to prove that $E(\pi^{C_1}) \triangle E(\mathrm{id}), \ldots, E(\pi^{C_p}) \triangle E(\mathrm{id})$ is a partition of $E(\pi) \triangle E(\mathrm{id})$. First, for all $1 \leq i < j \leq p$, by definition of the swap graph it follows that the positions, and thus also the vertices, affected by C_i are disjoint from the positions and vertices that are affected by C_j. Formally, $(E(\pi^{C_i}) \triangle E(\mathrm{id})) \cap (E(\pi^{C_j}) \triangle E(\mathrm{id})) = \emptyset$. For any component C, we next argue that $E(\pi^C) \triangle E(\mathrm{id}) \subseteq E(\pi) \triangle E(\mathrm{id})$. Clearly, for an edge $e = \{i,j\} \in E(\pi^C) \triangle E(\mathrm{id})$, either vertex i or j has to be affected by at least one swap in C. Then, no swap in $S \setminus C$ affects any of i and j, because such a swap would be adjacent to at least one swap in C. Hence, $e \in E(\pi) \triangle E(\mathrm{id})$.

Finally, consider an edge $e = \{i,j\} \in E(\pi) \triangle E(\mathrm{id})$. By the same argument as above, all swaps that affect any of i and j belong to the same component of G^S. Thus, since either vertex i or j is affected by a swap, it follows that there is a component C of G^S such that $e \in E(\pi^C) \triangle E(\mathrm{id})$. □

Theorem 3.3 LocalTSP*(r-Swap) is fixed-parameter tractable with respect to the combined parameter (k, r). It is solvable in $O(r^{2k}(r-1)^{2k} \cdot 4^k \cdot (k^2+n) \cdot n)$ time.*

Proof Let (G, ω, k) be an instance of LocalTSP(r-Swap). Furthermore, let S be a sequence of at most k r-swaps such that applying S to id results in an improved

Hamiltonian cycle π. By Lemma 3.2 we can assume that G^S is connected. The algorithm consists of two parts. First, the algorithm guesses the positions of all swaps in S and, second, it finds their correct order.

To describe the first part, for convenience, we assume for all swaps $\sigma(i,j)$ that $j \in \{i+1, i+2, \ldots, i+r-1\}$. Furthermore, we define an ordering relation $\leq$ on swaps with $\sigma(i,j) \leq \sigma(t,p)$ if and only if $i < t$ or $i = t \wedge j \leq p$. Let $\sigma_1, \sigma_2, \ldots, \sigma_s$ with $s \leq k$ be the swaps of S sorted with respect to $\leq$ in ascending order.

In the first part of the algorithm, by branching into all possibilities for the positions of the swaps, the algorithm guesses all swaps in the order given above: At the beginning, the algorithm branches into all possibilities to find the position i_1 for $\sigma_1(i_1, j_1)$ and then into the $r-1$ possibilities to find the position j_1. Now, suppose we have already found the swap $\sigma_t(i_t, j_t)$. We next describe how to find the swap $\sigma_{t+1}(i_{t+1}, j_{t+1})$. By the ordering we know that $i_1 \leq \ldots \leq i_t \leq i_{t+1}$ and, since all swaps are r-swaps, for all $1 \leq p \leq t$ with $j_p > i_t$ it holds that $j_p - i_t \leq r-1$. From this and since G^S is connected (Lemma 3.2), it follows that $i_{t+1} - i_t \leq r$. Thus, we can find the position of i_{t+1} by branching into $r+1$ possibilities. Afterwards, by branching into $r-1$ possibilities we find the position j_{t+1}. Overall, the positions of σ_{t+1} can be guessed by branching into at most r^2 possibilities, and there are at most $r^{2k-1} \cdot n$ possible positions of the swaps in total.

In the second part, the algorithm guesses the order of the r-swaps. Clearly, the trivial way to do that is by trying all permutations of the swaps, resulting in a total running time of $O(r^{2k-1}k! \cdot n)$. This already shows that the problem is fixed-parameter tractable for (k, r). We next describe how this can be accelerated in case that $4r^2 < k$. To this end, let $\sigma^{(1)}, \sigma^{(2)}, \ldots, \sigma^{(s)}$ be all swaps in S in the order of their application resulting in π. Clearly, if there are two subsequent swaps $\sigma^{(t)}(i,j)$ and $\sigma^{(t+1)}(i',j')$ such that $\{i,j\} \cap \{i',j'\} = \emptyset$, then reversing their order in the application of the swaps also results in π. More generally, instead of finding a total order of the swaps, it is sufficient to find a partial order of the swaps that defines the order for any pair of swaps $\sigma(i,j)$ and $\sigma(t,p)$ where $|\{i,j\} \cap \{t,p\}| = 1$. Clearly, we do not have to define the order of two swaps which are of the same *type*, that is, where $\{i,j\} = \{t,p\}$. Thus, for a position i, consider all swaps which affect position i. Since all these swaps are r-swaps, there can be at most $2r-2$ different types that affect position i. Hence, if there are k_i swaps that affect position i, then there are at most $(2r-2)^{k_i}$ different permutations of these swaps. Combining the number of possibilities of all affected positions, since each swap affects exactly two positions, it follows that there are at most $(2r-2)^{2k}$ permutations of all swaps yielding different Hamiltonian cycles. Once the partial orders at all relevant positions are determined, we check whether this can be obtained by some total order of the swaps, and find this order in $O(k^2)$ time, by representing the partial orders by arcs in a directed graph on the set of swaps and finding a topological order for this graph. Then we apply the swaps in this order in $O(k)$ time and check whether we obtain an improved Hamiltonian cycle in linear time. Together with the first part, the whole algorithm runs in $O(r^{2k}(r-1)^{2k} \cdot 4^k \cdot (k^2+n) \cdot n)$ time. □

Since the r-Swap distance is bounded by the r-Reversal distance (the corresponding function is $rk/2$, see Figure 2.1), the above theorem implies also the existence of an $O(r^{rk}(r-1)^{rk} \cdot 2^{rk} \cdot ((rk)^2+n) \cdot n)$-time permissive algorithm for LocalTSP(r-Reversal), that is, an algorithm that returns an improved Hamiltonian cycle whenever there is an improved Hamiltonian cycle in r-Reversal distance at most k from the given cycle. By modifying the algorithm from Theorem 3.3, we can obtain a strict local search algorithm for LocalTSP(r-Reversal) with a better running time.

Theorem 3.4 LocalTSP*(r-Reversal) is fixed-parameter tractable with respect to the combined parameter (k, r). It is solvable in $O(2^{rk} \cdot r^{2k-1} \cdot (r-1)^k \cdot (k^2 + rk + n) \cdot n)$ time.*

Proof We modify the algorithm from Theorem 3.3. We use the first part without changing it. That is, the position of the r-reversals are guessed in the "$\leq$" order in $O(r^{2k-1} \cdot n)$ time. Again one can show that if the t-th guessed reversal is $\rho(i_t, j_t)$, then the $(t+1)$-th reversal $\rho(i_{t+1}, j_{t+1})$ must fulfill $i_{t+1} - i_t \leq r$, as, otherwise, the first t reversals or the last $k-t$ reversals would yield an improved Hamiltonian cycle themselves.

For the second part we again observe that we do not need to know the total order of the guessed r-reversals. It is enough to know the order of reversals $\rho(i, j)$ and $\rho(i', j')$ if $\{i, i+1, \ldots, j\} \cap \{i', i'+1, \ldots, j'\} \neq \emptyset$ and $(i, j) \neq (i', j')$. Also observe that for this purpose it suffices to provide the order on positions where some of the guessed reversals start, that is, on positions i for which there is a j such that $\rho(i, j)$ is one of the guessed reversals. Now assume that the orders for all such positions $1 \leq t' < t$ have been already determined and we want to find the order at position t. Suppose that there are a_t reversals $\rho(i, j)$ with $i < t \leq j$ and b_t of them with $t = i < j$. Observe that the order of the a_t reversals is already known from some previous position. We first determine the order of the b_t reversals starting at position t. As there are at most $r-1$ types of r-reversals starting at t, there are at most $(r-1)^{b_t}$ different orders of them. Now it remains to determine the relative order of the reversals starting at t and those starting before. With the known orders within these groups, we have less than $2^{a_t+b_t}$ such orders. To determine the running time of the algorithm, we multiply the number of possibilities over all positions. The total number of orders of the reversals yielding different Hamiltonian cycles is at most $\prod_{t=1}^{n}(r-1)^{b_t} \cdot 2^{a_t+b_t} = (r-1)^{\sum_{t=1}^{n} b_t} \cdot 2^{\sum_{t=1}^{n} a_t+b_t} \leq (r-1)^k \cdot 2^{rk}$ as the sum $\sum_{t=1}^{n} b_t$ of the number of reversals starting at some position is at most k, while the sum $\sum_{t=1}^{n} a_t + b_t$ of the number of reversals affecting the particular position is at most rk for r-reversals.

Once the partial orders at all positions are determined, we check whether this can be obtained by some total order of the reversals, and find this order in $O(k^2)$ time. Then we apply the reversals in this order in $O(kr)$ time and check whether we obtain an improved Hamiltonian cycle in linear time. Therefore the whole algorithm runs in $O(2^{rk} \cdot r^{2k-1} \cdot (r-1)^k \cdot (k^2 + rk + n) \cdot n)$ time. □

Note that the swap graph of a swap sequence that yields the *best* improved Hamiltonian cycle in the local neighborhood does not have to be connected, and thus Lemma 3.2 cannot be extended to this case. However, we remark that, for LocalTSP(λ) with $\lambda \in \{r\text{-Swap}, r\text{-Reversal}\}$, by applying a standard dynamic programming approach, the algorithms given in the proofs of Theorems 3.3 and 3.4 can be extended such that not only *any* improved Hamiltonian cycle is found but also the best improved Hamiltonian cycle within the local neighborhood.

Further, analyzing the proofs of Theorems 3.3 and 3.4, one can show that if there is an improved Hamiltonian cycle in LocalTSP(r-Swap) or LocalTSP(r-Reversal), then there is also an improved cycle which differs from the given one only on vertices $v_i, v_{i+1}, \ldots, v_{i+rk-1}$ for some i. Therefore, one can reduce an input instance to polynomially many instances of the same problem, each having its size bounded by a polynomial in k and r. It is enough to replace the part of the cycle between v_{i+rk-1} and v_i by a length-rk path formed by dummy vertices. Such a self-reduction is known as polynomial *Turing kernelization* [31].

Proposition 3.1 LocalTSP*(r-Swap) and* LocalTSP*(r-Reversal) admit a reduction to n Turing kernels, each with at most* $2rk$ *vertices, and the reduction can be computed in linear time.*

In contrast to Proposition 3.1, in the next section we show that LocalTSP(λ) does not admit a polynomial-size kernel for any distance measure λ considered in this work, even when restricted to planar graphs.

4 Planar Graphs

In this section we investigate the complexity of LocalTSP on planar graphs. Note that whether LocalTSP(Edge) on planar graphs parameterized by the locality parameter k is fixed-parameter tractable or not is the central open question stated by Fellows et al. [14]. We do not answer this question; however, we show that on planar graphs LocalTSP(λ) for $\lambda \in \{\text{Swap, Edit}\}$ is fixed-parameter tractable for parameter k. Before that, we show that LocalTSP(λ) on planar graphs does not admit a polynomial-size kernel for any of the distance measures λ considered in this work.

4.1 Non-Existence of a Polynomial-Size Kernel

Bodlaender et al. [5] have shown that a parameterized problem does not admit a polynomial-size kernel (unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$) if its unparameterized variant is NP-hard and if it is *compositional*. A parameterized problem is compositional if there is a polynomial time algorithm that takes as input instances $(I_1, k), \ldots, (I_t, k)$ and computes a new instance (I, k') where k' is upper-bounded by a polynomial in k and (I, k') is a yes-instance if and only if (I_j, k) is a yes-instance for some $1 \leq j \leq t$.

We prove in detail that LocalTSP(r-Swap) does not admit a polynomial-size problem kernel. As can be seen in Figure 2.1, all distance measures considered in this work are r-Swap bounded and thus r-Swap can be viewed as the least powerful distance measure. Thus, a similar argumentation is also valid for the other distance measures.

To show that LocalTSP(r-Swap) on planar graphs has no polynomial-size kernel, we first consider a more restricted variant, namely LargeLocalTSP(r-Swap), where it is required that the underlying planar graph has more than $2rk$ vertices. We show that LargeLocalTSP(r-Swap) is NP-hard on planar graphs by a many-to-one reduction from Weighted Antimonotone 2-Sat; by exploiting the properties implied by the requirement that there are more than $2rk$ vertices, we then show that it is compositional.

Lemma 4.1 LargeLocalTSP*(r-Swap) on planar graphs is NP-hard.*

Proof We reduce from the NP-complete Weighted Antimonotone 2-Sat problem to the LocalTSP(r-Swap) problem and then show how to extend the construction to LargeLocalTSP(r-Swap).

> Weighted Antimonotone 2-Sat
> **Input:** A Boolean formula in 2-conjunctive normal form (2-CNF) where all literals are negative, and a positive integer c.
> **Question:** Is there a satisfying assignment such that at least c variables are set to True?

Weighted Antimonotone 2-Sat can be easily seen to be equivalent to the NP-complete Independent Set problem, where the vertices correspond to the variables of the formula, the edges correspond to the clauses of the formula, the vertices taken into the independent set correspond to the variables set to True and the question is whether the graph admits an independent set of size at least c.

Let $\mathcal{F}$ be a 2-CNF formula which forms together with a positive integer c an instance of Weighted Antimonotone 2-Sat. We first form a 3-CNF $\mathcal{F}'$ by adding to $\mathcal{F}$ a new so-called *dummy variable* y such that each clause contains the literal $\neg y$. Furthermore, except for y, for every variable x in $\mathcal{F}$ we add the clause $(\neg x \vee y)$ to $\mathcal{F}'$. Observe that the formulas $\mathcal{F}$ and $\mathcal{F}'$ have a trivial satisfying assignment where all variables are set to False.

Next, we apply to $\mathcal{F}'$ a reduction from 3-Sat to Planar Hamiltonian Cycle due to Garey et al. [18], obtaining a planar graph G such that G admits a Hamiltonian cycle if and only if $\mathcal{F}'$ has a satisfying assignment. Then, the trivial satisfying assignment where all variables are set to False induces a Hamiltonian cycle in G. Moreover, it follows from the details of the construction that for every variable there are two edges such that the usage of these edges in a Hamiltonian cycle specifies whether, in the corresponding satisfying assignment, the variable has to be set True or False. This means that there is an edge that is used if and only if the variable is set False and another edge that is used if and only if the variable is set True. We briefly refer to them by the False-edge and True-edge, respectively.

In order to form a LocalTSP(r-Swap) instance, we assign weight one to every False-edge. Hence, if $\mathcal{F}'$ contains n variables, then the satisfying assignment where all variables are set to False has weight n. Finally, assign weight c to the True-edge of y, whereas all remaining edges have weight zero. Every permutation of t elements can be sorted by at most $\binom{t}{2}$ 2-swaps. Hence, denoting the Hamiltonian cycle through G that uses all False-edges by id and setting $k = \binom{|V(G)|}{2}$ allows to choose any permutation as a solution of the LocalTSP(r-Swap) instance (G, ω, k).

We next show the correctness of the construction above, meaning that there is a satisfying assignment for $\mathcal{F}$ with at least c variables set to True if and only if there is a Hamiltonian cycle through G with weight strictly less than n. First, assume that there is a satisfying assignment for $\mathcal{F}$ with at least c variables set to True. It is clear that extending this assignment by setting $y =$ True we also get a satisfying assignment for $\mathcal{F}'$. Moreover, since in the corresponding Hamiltonian cycle through G at least c zero-weight True-edges are used, and since the weight-c True-edge for y is used instead of the weight-1 False-edge, the weight of the Hamiltonian cycle is at most $n + c - 1 - c < n$.

For the reverse direction, assume that there is a Hamiltonian cycle with weight strictly less than n and consider the corresponding satisfying assignment for $\mathcal{F}'$. Then, there is at least one variable, say x, that is set to True. Since the clause $(\neg x \vee y) \in \mathcal{F}'$ has to be satisfied, it follows that y is also set to True. Hence, removing y from the assignment results in a satisfying assignment for $\mathcal{F}$ and since the True-edge of y has weight c but the weight of the cycle is less than n, in total there have to be at least c other variables set to True.

The presented argumentation shows that LocalTSP(r-Swap) is NP-hard on planar graphs. Moreover, it is clear from the construction by Garey et al. [18] that there is an edge (for instance, the edge from the last clause-gadget to the first variable-gadget) that

has to be used in any Hamiltonian cycle. Thus, subdividing this edge an appropriate number of times, we also get an equivalent LargeLocalTSP(r-Swap) instance. □

In order to apply the framework of Bodlaender et al. [5], it remains to show that LargeLocalTSP(r-Swap) on planar graphs is compositional. To prove this, we first need the following easy observation.

Observation 4.1 For a LargeLocalTSP(r-Swap) instance (G, ω, k), let $\pi \in \mathcal{S}_n$ be an improved Hamiltonian cycle that can be obtained by a sequence S of at most k r-swaps. Then, there is an edge $e = \{v_t, v_{t+1}\} \in E(\pi) \cap E(\mathrm{id})$ such that the vertices v_t and v_{t+1} are unaffected and no swap goes over these vertices. Formally, this means that for all $\sigma(i, j) \in S$ it holds that neither t nor $t + 1$ is contained in $\{i, i + 1, \ldots, j - 1, j\}$.

Proof A LargeLocalTSP(r-Swap) instance fulfills $n > 2kr$. Then, since performing k r-swaps cannot affect more than $2k$ vertices, it follows that there are some r consecutive vertices that are unaffected. The second statement follows because one r-swap cannot "span" more than $r - 1$ vertices and thus no swap affects or goes over these vertices. □

Lemma 4.2 LargeLocalTSP*(r-Swap) on planar graphs is compositional.*

Proof Suppose that we are given, for a positive integer $t \in \mathbb{N}$, instances (G_1, ω_1, k), $(G_2, \omega_2, k), \ldots, (G_t, \omega_t, k)$ of LargeLocalTSP(r-Swap). For each graph G_i with $1 \leq i \leq t$ we introduce several copies of G_i, and in each copy we choose a start and an end vertex. Then, a graph G with weight function ω is composed by arranging the copies in an arbitrary order and connecting the end vertex of a copy by a zero-weight edge to the start vertex of the subsequent copy. Finally, the end vertex of the last copy is connected by a zero-weight edge to the start vertex of the first copy. We describe a Hamiltonian path from the start to the end vertex of each copy. All these paths together with the zero-weight edges between the copies form the Hamiltonian cycle id and (G, ω, k) forms the composed instance. Furthermore, since the start and the end vertex of each copy are connected by an edge, and, therefore, there is an embedding of the copy with both vertices on the boundary of the outer face, graph G is planar.

In order to form the composed graph G, for $1 \leq i \leq t$ and $n_i := |V(G_i)|$ let $v_1^i, v_2^i, \ldots, v_{n_i}^i, v_1^i$ be the given Hamiltonian cycle in G_i. For $1 \leq j < n_i$, we add one copy of G_i with the vertex v_{j+1}^i being the start vertex and v_j^i being the end vertex. The given Hamiltonian cycle in G_i then induces a Hamiltonian path $v_{j+1}^i, v_{j+2}^i, \ldots, v_{n_i}^i, v_1^i, \ldots, v_j^i$ from v_{j+1}^i to v_j^i in this copy. We complete the construction by adding a copy of G_i where v_1^i is the start and $v_{n_i}^i$ is the end vertex.

In the following we prove the correctness of the reduction, that is, the composed instance (G, ω, k) is a yes-instance of LargeLocalTSP(r-Swap) if and only if there is a yes-instance (G_i, ω_i, k) with $1 \leq i \leq t$. First, suppose that there is an improved Hamiltonian cycle for G which performs at most k swaps. By our construction it is obvious that any Hamiltonian cycle through G enters a copy at its start vertex and leaves it at its end vertex. Thus, there is at least one copy where the improved Hamiltonian cycle for G implies an improved Hamiltonian path from the start to the end vertex. As the improved Hamiltonian path does not use the edge between the end vertex and the start vertex of the particular copy, by adding this edge we obviously get an improved Hamiltonian cycle in the corresponding graph, which is in r-Swap distance at most k from the given Hamiltonian cycle.

For the reverse direction, suppose that there is an improved Hamiltonian cycle for the graph G_i. By Observation 4.1 there is at least one "preserved" edge $\{v^i_j, v^i_{j+1}\}$ or $\{v^i_{n_i}, v^i_1\}$, where no swap goes over its endpoints. It is clear that the same swaps are also r-swaps in the copy of G_i with v^i_{j+1} (or v^i_1) as a start vertex and v^i_j (or $v^i_{n_i}$) as the end vertex. Hence all swaps which were performed to G_i can be performed to this copy, resulting in an improved Hamiltonian cycle for G. □

Employing the framework of Bodlaender et al. [5], Lemma 4.2 and Lemma 4.1 together imply that LargeLocalTSP(r-Swap) does not admit a polynomial-size kernel, unless NP ⊆ coNP/poly. Thus, the next theorem follows.

Theorem 4.1 *Unless NP ⊆ coNP/poly,* LocalTSP*(r-Swap) on planar graphs does not admit a polynomial-size kernel with respect to the parameter k for any $r \geq 2$.*

Following exactly the same argumentation as for LocalTSP(r-Swap), we state that one can show that on planar graphs LocalTSP(λ) for $\lambda \in \{r$-Reversal (for any $r \geq 2$), Swap, Edit, Reversal, Max-Shift, Edge$\}$ does not admit a polynomial-size kernel with respect to parameter k, unless NP ⊆ coNP/poly.

Corollary 4.1 *Unless NP ⊆ coNP/poly, LocalTSP(λ) for $\lambda \in \{r$-Reversal, Swap, Edit, Reversal, Max-Shift, Edge$\}$ on planar graphs does not admit a polynomial-size kernel with respect to the parameter k for any $r \geq 2$.*

4.2 LocalTSP(Edit) and LocalTSP(Swap) are Fixed-Parameter Tractable

LocalTSP(Edit) and LocalTSP(Swap) on planar graphs are unlikely to allow for polynomial-size kernels; however, they admit a permissive FPT-algorithm. In the following we argue for LocalTSP(Swap); the result for the Edit distance can be obtained along the same lines. The proof relies on the following two lemmas.

Lemma 4.3 *If a* LocalTSP*(Swap) instance with parameter k admits an improved Hamiltonian cycle, then it also admits an improved Hamiltonian cycle which differs from the given one only within the distance-$3k$ neighborhood around some vertex.*

Proof Due to Lemma 3.2 from Section 3.2, it suffices to prove the statement only for improved Hamiltonian cycles obtained by a sequence of swaps where the corresponding swap graph is connected. Consider the set A of all vertices affected by swaps and their neighbors in the given Hamiltonian cycle Q. Since there are at most k swaps we have $|A| \leq 6k$. Clearly, the improved Hamiltonian cycle R coincides with Q outside A.

Now we consider the connected components of $G[A]$. Furthermore, we also consider some maximal path P of R such that P is formed only by vertices of A. Obviously, P contains vertices of only one component of $G[A]$; call this component C. Let x, y be the two neighbors of P on R. Since the part of Q outside of A is preserved, the path P' of Q between x and y also contains only vertices of C. Moreover, since x and y are not affected by any swap and, therefore, there must be the same number of vertices between them in Q and in R, path P' has the same length as P. By repeating the argument, one can show that C can be partitioned into such paths of Q and R in a one-to-one correspondence. Therefore, one can obtain R in C from Q only using swaps within C.

Due to the above argument, it suffices to consider the case that each swap is within one component of $G[A]$. Observe that swaps in different components of $G[A]$ are not

adjacent in the swap graph. Therefore, since we assumed the swap graph to be connected, $G[A]$ has only one component, that is, $G[A]$ is connected. Finally, in a connected graph with at most $6k$ vertices there is at least one vertex that has distance at most $3k$ to all others. □

The following lemma shows that, regardless of the distance measure, on planar graphs it is fixed-parameter tractable to find the best improved Hamiltonian cycle that differs from the given one only within the neighborhood of one specific vertex.

Lemma 4.4 *For an instance of* LocalTSP *on planar graphs and a vertex v one can find in $O(2^{O(k)} \cdot n + n^3)$ time the best Hamiltonian cycle among those differing from the given one only within distance k from v.*

Proof Start by deleting the edges having an endpoint of distance greater than k from v which are not part of the given Hamiltonian cycle (they cannot be used by the new Hamiltonian cycle anyway). Now contract the paths of the given Hamiltonian cycle formed by vertices in distance more than k from v into one vertex (to avoid duplicate edges). Then the whole graph is still planar and has diameter at most $2k + 2$ and, therefore, by a result of Robertson and Seymour [37] (see also [3]) has treewidth at most $6k + 6$. Thus it has branch-width at most $6k + 6$ [38]. Dorn et al. [10] showed that TSP on planar graphs (referred to as Planar Hamiltonian Cycle) with branch-width l can be solved in $O(2^{3.292l} \cdot l \cdot n + n^3)$ time. To modify their algorithm for our problem it is enough to force their algorithm to preserve the edges of paths that represent the parts of the given Hamiltonian cycle in distance more than k from vertex v. This is easy, as Dorn et al. [10] basically consider the solution to be the set of edges of the Hamiltonian cycle, and the branch-decomposition based dynamic programing actually starts with individual edges. It is enough to fill the tables so that the only solution on a required edge is to take this edge. □

Theorem 4.2 *There is a permissive FPT-algorithm for* LocalTSP*(λ) on planar graphs with respect to k for $\lambda \in \{Swap, Edit\}$.*

Proof We prove the theorem only for the Swap distance, the result for the Edit distance can be obtained along the same lines. Assume that there is an improved Hamiltonian cycle in Swap distance at most k from the given Hamiltonian cycle. By Lemma 4.3 we know that in this case there is an improved Hamiltonian cycle, differing from the given one only within distance at most $3k$ from some vertex v. We can find such a Hamiltonian cycle or a Hamiltonian cycle that is at least as good, by applying the algorithm from Lemma 4.4 on the $3k$-neighborhood of each vertex. The $O(n \cdot (2^{O(k)} \cdot n + n^3))$ running time follows. □

Following the same approach as Fellows et al. [14], Theorem 4.2 can be easily generalized to any class of graphs with bounded local treewidth. As Lemma 4.3 does not assume anything about the graph, we only have to modify Lemma 4.4. The lemma is true in any class of graphs with bounded local treewidth, but the corresponding running time depends on the respective class.

5 Conclusion

We left open the central open problem posed by Fellows et al. [14] whether LocalTSP(Edge) restricted to planar graphs is fixed-parameter tractable. However, we

indicated (see Section 2) that a permissive FPT-algorithm for LocalTSP(Edge) implies a permissive FPT-algorithm for LocalTSP(Reversal) and vice versa. Thus, the questions whether the problems are fixed-parameter tractable, are equivalent and this might help to shed new light on Marx's question. To this end, it might be beneficial to explore the connections of LocalTSP(Reversal) to the topic of Sorting by Reversals as studied in bioinformatics [6]. Moreover, it might be worthwhile to explore whether there are *strict* algorithms for LocalTSP(λ) for $\lambda \in \{\text{Swap, Edit}\}$ on planar graphs.

Assuming the Exponential Time Hypothesis [7, 25], we showed that there is no $O(n^{o(k/\log k)})$-time algorithm for LocalTSP(λ) for $\lambda \in \{\text{Swap, Edit, Reversal, Edge}\}$. Is there also a matching upper bound or can the lower bound still be improved?

Finally, our investigations might also be extended by moving from local neighborhoods for TSP to so-called exponential (but structured) neighborhoods as undertaken already in a non-parameterized setting [9, 22].

References

1. S. Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998. → p. 2.
2. E. Balas. New classes of efficiently solvable generalized traveling salesman problems. *Annals of Operations Research*, 86:529–558, 1999. → pp. 3 and 5.
3. H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209:1–45, 1998. → p. 18.
4. H. L. Bodlaender. Kernelization: New upper and lower bound techniques. In *Proceedings of the 4th International Workshop on Parameterized and Exact Computation (IWPEC '09)*, volume 5917 of *Lecture Notes in Computer Science*, pages 17–37. Springer, 2009. → p. 4.
5. H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8):423–434, 2009. → pp. 14, 16, and 17.
6. A. Caprara. Sorting by reversals is difficult. In *Proceedings of the 1st Conference on Research in Computational Molecular Biology (RECOMB '97)*, pages 75–83. ACM Press, 1997. → pp. 2 and 19.
7. J. Chen, B. Chor, M. Fellows, X. Huang, D. W. Juedes, I. A. Kanj, and G. Xia. Tight lower bounds for certain parameterized NP-hard problems. *Information and Computation*, 201(2):216–231, 2005. → pp. 4 and 19.
8. N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie Mellon University Pittsburg, Management Sciences Research Group, 1976. → p. 2.
9. V. G. Deineko and G. J. Woeginger. A study of exponential neighborhoods for the travelling salesman problem and for the quadratic assignment problem. *Mathematical Programming*, 87(3):519–542, 2000. → p. 19.
10. F. Dorn, E. Penninkx, H. L. Bodlaender, and F. Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58:790–810, 2010. → p. 18.
11. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999. → pp. 2 and 4.
12. M. R. Fellows. Towards fully multivariate algorithmics: Some new results and directions in parameter ecology. In *Proceedings of the 20th International Workshop on Combinatorial Algorithms (IWOCA '09)*, volume 5874 of *Lecture Notes in Computer Science*, pages 2–10. Springer, 2009. → p. 4.
13. M. R. Fellows, D. Hermelin, F. A. Rosamond, and S. Vialette. On the parameterized complexity of multiple-interval graph problems. *Theoretical Computer Science*, 410(1): 53–61, 2009. → p. 7.
14. M. R. Fellows, F. V. Fomin, D. Lokshtanov, F. Rosamond, S. Saurabh, and Y. Villanger. Local search: Is brute-force avoidable? *Journal of Computer and System Sciences*, 78(3): 707–719, 2012. → pp. 3, 14, and 18.
15. G. Fertin, A. Labarre, I. Rusu, É. Tannier, and S. Vialette. *Combinatorics of Genome Rearrangements*. The MIT Press, 2009. → p. 2.

16. J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006. → pp. 2 and 4.
17. F. Fomin, D. Lokshtanov, V. Raman, and S. Saurabh. Fast local search algorithm for weighted feedback arc set in tournaments. In *Proceedings of the 24th Conference on Artificial Intelligence (AAAI '10)*, pages 65–70, 2010. → p. 3.
18. M. Garey, D. Johnson, and R. Tarjan. The planar Hamiltonian circuit problem is NP-complete. *SIAM Journal on Computing*, 5(4):704–714, 1976. → p. 15.
19. S. Gaspers, E. J. Kim, S. Ordyniak, S. Saurabh, and S. Szeider. Don't be strict in local search. In *Proceedings of the 26th Conference on Artificial Intelligence (AAAI '12)*, pages 486–492, 2012. → pp. 4 and 5.
20. J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1):31–45, 2007. → p. 4.
21. G. Gutin and A. Punnen. *The Traveling Salesman Problem and its Variations*. Kluwer Academic Publishers, 2002. → p. 2.
22. G. Gutin, A. Yeo, and A. Zverovitch. The traveling salesman problem and its variations. chapter 1: Exponential neighborhoods and domination analysis for the TSP, pages 223–256. Kluwer Academic Publishers, 2002. → p. 19.
23. S. Hartung and R. Niedermeier. Incremental list coloring of graphs, parameterized by conservation. In *Proceedings of the 7th Annual Conference on Theory and Applications of Models of Computation (TAMC '10)*, volume 6108 of *Lecture Notes in Computer Science*, pages 258–270. Springer, 2010. → p. 4.
24. M. Held and R. M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962. → p. 2.
25. R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001. → pp. 2 and 19.
26. D. S. Johnson and L. A. McGeoch. The traveling salesman problem: A case study in local optimization. In *Local Search in Combinatorial Optimization*, pages 215–310. 1997. → p. 2.
27. D. S. Johnson and L. A. McGeoch. Experimental analysis of heuristics for the STSP. In *The Traveling Salesman Problem and its Variations*, pages 369–443. Kluwer Academic Publishers, 2004. → p. 2.
28. A. Krokhin and D. Marx. On the hardness of losing weight. *ACM Transactions on Algorithms*, 8(2):19:1–19:18, 2012. → pp. 4 and 5.
29. E. Lawler, J. Lenstra, A. Kan, and D. Shmoys. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, volume 3. Wiley New York, 1985. → p. 2.
30. D. Lokshtanov, D. Marx, and S. Saurabh. Lower bounds based on the Exponential Time Hypothesis. *Bulletin of the EATCS*, 84:41–71, 2011. → p. 4.
31. D. Lokshtanov, N. Misra, and S. Saurabh. Kernelization–preprocessing with a guarantee. In *The Multivariate Algorithmic Revolution and Beyond–Essays Dedicated to Michael R. Fellows*, volume 7370 of *Lecture Notes in Computer Science*, pages 130–161. Springer, 2012. → pp. 4 and 13.
32. D. Marx. Searching the k-change neighborhood for TSP is W[1]-hard. *Operations Research Letters*, 36(1):31–36, 2008. → pp. 2, 3, 7, 8, and 10.
33. D. Marx. Can you beat treewidth? *Theory of Computing*, 6(1):85–112, 2010. → p. 10.
34. D. Marx and I. Schlotter. Stable assignment with couples: Parameterized complexity and local search. *Discrete Optimization*, 8(1):25–40, 2011. → pp. 4 and 5.
35. R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006. → pp. 2 and 4.
36. R. Niedermeier. Reflections on multivariate algorithmics and problem parameterization. In *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science (STACS '10)*, volume 5 of *LIPIcs*, pages 17–32. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010. → p. 4.
37. N. Robertson and P. D. Seymour. Graph minors. V. Excluding a planar graph. *Journal of Combinatorial Theory. Series B*, 41:92–114, 1986. → p. 18.
38. N. Robertson and P. D. Seymour. Graph minors. X. Obstructions to tree-decomposition. *Journal of Combinatorial Theory. Series B*, 52(2):153–190, 1991. → p. 18.
39. S. Szeider. The parameterized complexity of k-flip local search for SAT and MAX SAT. *Discrete Optimization*, 8(1):139–145, 2011. → p. 4.