

A Parameterized-Complexity Framework for Finding Local Optima

Robert Ganian  

Algorithms and Complexity Group, TU Wien, Austria

Hung P. Hoang  

Algorithms and Complexity Group, TU Wien, Austria

Christian Komusiewicz  

Institute of Computer Science, Friedrich Schiller University Jena, Germany

Nils Morawietz  

LaBRI, Université de Bordeaux, France

Institute of Computer Science, Friedrich Schiller University Jena, Germany

Abstract

Local search is a fundamental optimization technique that is both widely used in practice and deeply studied in theory, yet its computational complexity remains poorly understood. The traditional frameworks, PLS and the standard algorithm problem, introduced by Johnson, Papadimitriou, and Yannakakis (1988) fail to capture the methodology of local search algorithms: PLS is concerned with finding a local optimum and not with using local search, while the standard algorithm problem restricts each improvement step to follow a fixed pivoting rule. In this work, we introduce a novel formulation of local search which provides a middle ground between these models. In particular, the task is to output not only a local optimum but also a chain of local improvements leading to it. With this framework, we aim to capture the challenge in designing a good pivoting rule. Especially, when combined with the parameterized complexity paradigm, it enables both strong lower bounds and meaningful tractability results. Unlike previous works that combined parameterized complexity with local search, our framework targets the whole task of finding a local optimum and not only a single improvement step. Focusing on two representative meta-problems – SUBSET WEIGHT OPTIMIZATION PROBLEM with the c -swap neighborhood and WEIGHTED CIRCUIT with the flip neighborhood – we establish fixed-parameter tractability results related to the number of distinct weights, while ruling out an analogous result when parameterizing by the distance to the nearest optimum via a new type of reduction.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Local Search, Parameterized Complexity, PLS

Digital Object Identifier 10.4230/LIPIcs.ITCS.2026.66

Related Version *Full Version*: <https://arxiv.org/abs/2601.00560>

Funding *Robert Ganian*: Austrian Science Foundation (FWF), project 10.55776/Y1329 and Vienna Science and Technology Fund (WWTF), project 10.47379/ICT22029.

Hung P. Hoang: Austrian Science Foundation (FWF), projects 10.55776/Y1329 and ESP1136425.

Nils Morawietz: French ANR, project ANR-22-CE48-0001 (TEMPOGRAL).

1 Introduction

Local search is one of the most commonly employed paradigms in the design of algorithms for optimization problems. The idea underlying local search algorithms is both natural and simple: start from an initial solution S to the problem and apply a series of local improvement steps until reaching a local optimum, that is, a solution which cannot be improved by any local change. The definition of local improvement steps typically involves straightforward



© Robert Ganian, Hung P. Hoang, Christian Komusiewicz, and Nils Morawietz; licensed under Creative Commons License CC-BY 4.0

17th Innovations in Theoretical Computer Science Conference (ITCS 2026).

Editor: Shubhangi Saraf; Article No. 66; pp. 66:1–66:20

Leibniz International Proceedings in Informatics



LIPIC Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

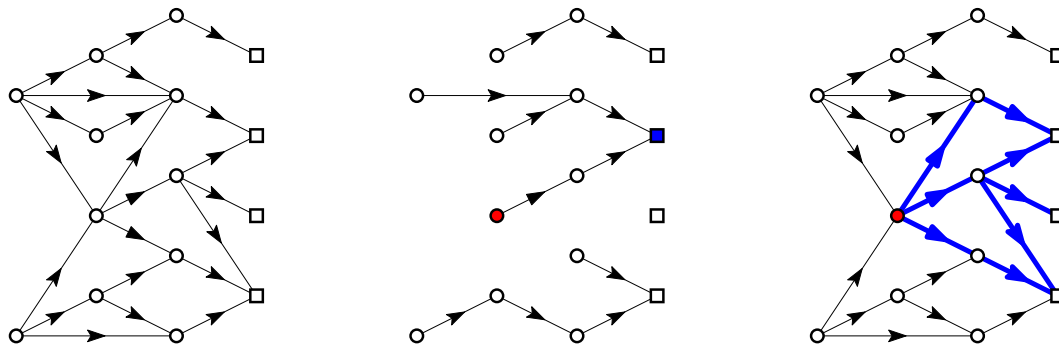
operations such as swapping or moving a constant number of elements in the solution. There are almost always many ways these steps could be applied, and one typically speaks of the *local neighborhood* of a solution S to subsume all possible solutions that can be obtained from S by performing one round of permissible operations. A solution S is called a *local optimum* if there is no better solution in its local neighborhood, and local search algorithms are typically tasked with finding such a local optimum via a sequence of local improvement steps.

While the local search paradigm often performs very well in practice, currently established complexity-theoretic tools exclude tractability of finding local optima for many problems of interest [1, 24]. The central complexity class in the theory of local search is PLS [16] (for “*polynomial local search*”). Inclusion in PLS essentially means that one can search the local neighborhood of any given solution in polynomial time¹. While inclusion in PLS is a natural prerequisite for efficient local search, it does not guarantee that one can find a local optimum in polynomial time – to the contrary, it is now widely believed that PLS-complete problems do not admit any polynomial-time local search algorithm [24]. In this sense, establishing PLS-hardness for a local search problem can be seen as a counterpart to establishing NP-hardness for decision problems.

Given the above discrepancy between the performance of local search algorithms and inclusion in PLS, researchers have proposed a second classical formalization of local search: the *standard algorithm problem* [16]. There, one fixes not only the notion of local neighborhood but also the specific *pivoting rule*, that is, the specific procedure used to compute the next solution in the local improvement step. The underlying task is then to identify the local optimum that will be obtained by exhaustively applying the selected pivoting rule from a provided initial solution S . The standard algorithm problem provides a useful way of making complexity-theoretic statements about local search – several standard algorithm problems are known to be PSPACE-complete [29, 27, 30, 3, 21] – but suffers from the drawback that each such statement applies to a combination of pivoting rule and local search problem. This drawback has already been acknowledged in the seminal work of Schäffer and Yannakakis [29, Page 62, 1st paragraph]: while formal lower bounds made for the standard algorithm problem only hold for specific pivoting rules (such as, e.g., moving to the first discovered improvement in the local neighborhood), it is more desirable to exclude efficient local search for *any* pivoting rule and this is also what can be inferred from most existing lower bound constructions [16, 27, 30, 3, 21].

Crucially, regardless of whether one chooses the PLS or standard algorithm formulation, most local search problems remain computationally intractable. In this article, we introduce a novel perspective which allows us to establish not only stronger **lower bounds**, but also **tractability** for local search problems. We do so by developing a theory of parameterized local search which is inspired by and connected to the well-established *parameterized complexity* paradigm [5, 4]. What distinguishes our work from prior studies that examined parameterized complexity of local search [22, 31, 20, 14, 12, 17, 13] is that our results apply to the local search task as a whole and not only to a single improving step (see the discussion of related work at the end of this section). In fact, our framework targets a wide variety of common settings where computing a single improving step is polynomial-time solvable. Moreover, as a byproduct of our approach we also obtain a formulation of local search problems that lies between PLS and the standard algorithm: one where negative results exclude tractability with respect to every pivoting rule (unlike the latter) while positive ones guarantee efficient local search (unlike the former).

¹ Formal definitions are provided in Section 2.



■ **Figure 1** Comparison of the three models. The graphs are transition graphs, where solutions are vertices and local improvements are edges. **Left:** The PLS formulation asks for a local optimum (i.e., any square in the transition graph) without needing to follow a sequence of improvement steps. **Middle:** In the standard algorithm problem, we have a specific pivoting rule (i.e., every solution has at most one outgoing edge) and a given initial solution (depicted as the red circle). The task is to find the unique local optimum (blue square) reachable from the initial solution. **Right:** In our pivoting formulation, given an initial solution (red circle), we want to output an improving sequence from that solution to a local optimum (i.e., any maximal path in the blue subgraph).

Contributions. We develop our framework on two types of problems arising in the local search setting:

1. SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP [18], whose underlying optimization problem generalizes weighted variants of many graph problems such as MAXIMUM INDEPENDENT SET, MAX CUT, MINIMUM DOMINATING SET, MAXIMUM VERTEX COVER, and others. Here, solutions are vertex or edge subsets satisfying some arbitrary certifiable property and the local neighborhood is defined by swapping up to c vertices or edges for some fixed constant c .
2. WEIGHTED CIRCUIT/FLIP, which is the joint generalization of the classical MAX CIRCUIT/FLIP and MIN CIRCUIT/FLIP problems [16, 1] to arbitrary weights. Here, the input is a circuit along with a weight function over the output gates, solutions are input bit vectors and the local neighborhood is defined by flipping a single bit on the input, that is, inverting its bit.

For all the above problems, we primarily target the following local search task: given an initial solution S , output a series of improving steps from S to a local optimum. We note that this *pivoting* formulation forms a middle ground that avoids the drawbacks of both previously studied formulations of local search (see Figure 1 for an illustration):

- In the PLS formulation, a hypothetical algorithm can find the local optimum in an arbitrary way (without requiring improvement steps and potentially even yielding a solution worse than the given starting solution);
- In the standard algorithm formulation, one is forced to use a fixed pivoting rule and all obtained complexity-theoretic statements apply only to that specific rule.

As a direct consequence of previously established lower bounds [29, 26, 23], the problems mentioned above cannot admit a polynomial-time algorithm in the pivoting formulation. Moreover, such results are *unconditional*: there simply exist instances with initial solutions S for which the shortest sequence of local improvement steps is exponential (the so-called *all-exp* property [29]), and hence a local search algorithm cannot terminate in polynomial time. However, this does not rule out tractability in the parameterized sense – there, one typically asks for so-called *fixed-parameter tractability*, which means that there is an algorithm terminating in time $f(k) \cdot n^{O(1)}$ for input size n and some computable function f of a (possibly promised) parameter k .

After setting up the conceptual contributions outlined above, we proceed to our technical contributions: a parameterized analysis of local search problems. We summarize our two main findings below.

► **Main Finding 1.** *Local search for SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP and WEIGHTED CIRCUIT/FLIP is fixed-parameter tractable when parameterized by the number of distinct weights.*

► **Main Finding 2.** *Unless FPT = W[1], neither SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP nor WEIGHTED CIRCUIT/FLIP admit a fixed-parameter local search algorithm when parameterized by the distance (measured by number of improvement steps) to the nearest local optimum.*

Main Finding 1 holds for all considered formulations of local search (i.e., PLS, pivoting, standard algorithm), and is primarily based on Theorems 6.5-6.9. While these proofs are not technically challenging and rely on the weight reduction technique by Frank and Tardos [11], the finding showcases that parameterized complexity can provide meaningful tractability results for local search. In particular, it provides a bridge between the polynomial runtime of local search for many unweighted problems and the hardness results of their corresponding weighted variants.

Main Finding 2 applies only to the newly proposed pivoting formulation of local search. Indeed, on one hand the statement is vacuous in the standard algorithm formulation (if an algorithm cannot choose which improvement step to make, the distance to the local optimum is irrelevant). On the other hand, the existence of a *direct* connection between PLS-hardness and a collapse of complexity-theoretic decision classes is a long-standing open problem in the field – and while we do make progress towards establishing such a connection, completely settling this is beyond our current understanding (see also the Concluding Remarks in Section 7). To the best of our knowledge, Main Finding 2 is also the first lower bound establishing that the hardness of local search is neither due to the non-existence of a nearby local solution (i.e., the all-exp property) [28, 29, 25, 15] nor due to the intractability of computing an improvement step [22, 31, 20, 14, 12, 17, 13]: already finding the “correct” improvement steps is hard.

Main Finding 2 is based on Theorem 4.1 and Corollary 5.5, which are both non-trivial. As a starting point towards the former, we show that MAXIMUM INDEPENDENT SET/3-SWAP, a special case of SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP, has the *all-exp property*. This means that for infinitely many pairs of an instance and an initial solution, all reachable local optima are exponentially far away from the initial solution; that is, regardless of any pivoting rule, the standard local search algorithm always runs in exponential time. We establish this property via a tight PLS-reduction from MAX CUT/FLIP.² This notion of reduction is stronger than the original PLS-reduction and can be used to transfer the all-exp property; see Section 2 for formal definitions of these reductions. While there already exists a PLS-reduction from MAX CUT/FLIP to MAXIMUM INDEPENDENT SET/3-SWAP [18], that reduction is not tight and our adaptations to the known reduction require an involved analysis to show tightness. Equipped with this all-exp property, we are able to construct a reduction from the canonical W[1]-hard problem MULTICOLORED INDEPENDENCE SET to MAXIMUM INDEPENDENT SET/3-SWAP, thus establishing Main Finding 2 for the latter problem.

² Here, the local neighborhood allows to change for any single vertex the side of side of the partition. For example, the partition (A, B) has the flip-neighbor $(A \setminus \{v\}, B \cup \{v\})$ for each $v \in A$.

Towards establishing Main Finding 2 for WEIGHTED CIRCUIT/FLIP, we do not start from a known $W[1]$ -hard decision problem (such as MULTICOLORED INDEPENDENCE SET) but instead develop a new notion of reduction which can translate parameterized local search lower bounds directly. This notion adds to the well-established tight PLS-reduction a constraint that guarantees parametric stability by forbidding the creation of new long paths in the transition graph. Our result here – a concrete application of such a reduction from MAXIMUM INDEPENDENT SET/3-SWAP to MAX-CIRCUIT/FLIP – can thus be seen as a demonstration that the new parameterized lower bounds can be translated to pivoting formulations of other local search problems of interest.

Related Work. The PLS and standard algorithm formulations have been studied for a broad variety of search problems, including not only those captured by SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP and WEIGHTED CIRCUIT/FLIP but also the SIMPLEX METHOD [8] and GRADIENT DESCENT [7]. The all-exp property (which we newly establish for WEIGHTED INDEPENDENT SET/3-SWAP) has previously been conjectured and proven for TRAVELING SALESMAN/ k -OPT [19, 15] and MAX CUT/FLIP [26, 18], among others. Whether the SIMPLEX METHOD admits a pivoting rule that guarantees a polynomial number of iterations is one of the most important open problems in the area of linear programming.

The parameterized complexity of local search was studied for a large number of problems, including: STABLE MARRIAGE/ k -SWAP [22], MAX SAT/ k -FLIP [31], MAX CSP/ k -FLIP [20], TRAVELING SALESMAN under a variety of local distance measures [14], MAX c -CUT/ k -FLIP [12] and CLUSTER EDITING/ k -MOVE, CLUSTER DELETION/ k -MOVE [13]. In all of the above studies, the parameter measured the complexity of finding a single improving step in the local neighborhood. That perspective is complementary to the one explored in this paper – we target the practically motivated case where performing a single improving step is easy, yet finding a local optimum is difficult.

Paper Organization. We begin by setting up the basic preliminaries in Section 2 and introducing our framework in Section 3. Then, we proceed with establishing the technically involved lower bounds captured by Main Finding 2 for SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP in Section 4 and for WEIGHTED CIRCUIT/FLIP in Section 5. The positive results described in Main Finding 1 are detailed in Section 6, and we conclude with a discussion of possible future directions and open questions in Section 7.

2 Preliminaries

For two integers $a < b$, we denote by $[a, b]$ the set of integers between (and including) a and b . We write $[a]$ for $[1, a]$. Given a set S , a subset S' of S , and a function $f : S \rightarrow \mathbb{Q}$, we define $f(S') := \sum_{s \in S'} f(s)$. For two sets A and B , we denote by $A \oplus B$ their symmetric difference (i.e., $A \oplus B := (A \setminus B) \cup (B \setminus A)$). Given a graph G , the open neighborhood $N_G(v)$ of a vertex v is the set of all adjacent vertices of v in G . Its closed neighborhood $N_G[v]$ is defined as $N_G(v) \cup \{v\}$. We drop the subscript when the graph G is clear. For a vertex subset $S \subseteq V(G)$, we denote by $G[S]$ the subgraph of G induced by S .

A *local search problem* P is an optimization problem that consists of a set of instances D_P , a finite set of (feasible) solutions $F_P(I)$ for each instance $I \in D_P$, an objective function f_P that assigns an integer value to each instance $I \in D_P$ and solution $s \in F_P(I)$, and a neighborhood $N_P(s, I) \subseteq F_P(I)$ for each solution $s \in F_P(I)$. The size of every solution $s \in F_P(I)$ is bounded by a polynomial in the size of I . The goal is to find a *locally optimal*

solution for a given instance I ; that is, a solution $s \in F_P(I)$ such that no solution $s' \in N_P(s, I)$ yields a better objective value than $f_P(s, I)$. Formally, this means that for all $s' \in N_P(s, I)$, we have $f_P(s, I) \leq f_P(s', I)$ if P is a minimization problem, and $f_P(s, I) \geq f_P(s', I)$ if P is a maximization problem.

A common naming convention for a local search problem is of the form Q/N , where Q is the corresponding optimization problem and N is the notion of neighborhood. For example, MAX CUT/FLIP indicates the local search problem with instances, solutions, and objective function as defined by the optimization problem MAX CUT with the addition of the neighborhood as defined by the flip operation.

A *standard local search algorithm* for an instance I proceeds as follows. It starts with some initial solution $s \in F_P(I)$. Then it iteratively visits a neighbor with better objective value, until it reaches a local optimum. If a solution has more than one better neighbor, the algorithm has to choose one by some prespecified rule, often referred to as a *pivoting rule*. The *standard algorithm problem* is then defined as follows [16]: Given an instance I and an initial solution s , output the local optimum that would be produced by the standard local search algorithm (with a specific pivoting rule) starting from s .

Next, we formalize a series of classical notions related to the complexity class PLS.

► **Definition 2.1** (The class PLS [16]). *A local search problem P is in the class PLS, if there are three polynomial time algorithms A_P, B_P, C_P such that*

- *Given an instance $I \in D_P$, A_P returns a solution $s \in F_P(I)$;*
- *Given an instance $I \in D_P$ and a solution $s \in F_P(I)$, B_P computes the objective value $f_P(s, I)$ of s ; and*
- *Given an instance $I \in D_P$ and a solution $s \in F_P(I)$, C_P returns a neighbor of s with strictly better objective value, if it exists, and “locally optimal”, otherwise.*

The “basic” reductions used for PLS are defined below.

► **Definition 2.2** (PLS-reduction [16]). *A PLS-reduction from a local search problem P to a local search problem Q is a pair of polynomial-time computable functions Φ and Ψ that satisfy:*

1. *Given an instance $I \in D_P$, Φ computes an instance $\Phi(I) \in D_Q$; and*
2. *Given an instance $I \in D_P$ and a solution $s_q \in F_Q(\Phi(I))$, Ψ returns a solution $s_p \in F_P(I)$ such that if s_q is a local optimum for $\Phi(I)$, then s_p is a local optimum for I .*

A problem $Q \in \text{PLS}$ is PLS-complete if for every problem $P \in \text{PLS}$, there exists a PLS-reduction from P to Q .

A more restrictive (and often useful) kind of reduction was defined later, by Schäffer and Yannakakis. Before defining it, we will need the following notion:

► **Definition 2.3** (Transition graph). *The transition graph T_I of an instance I of a local search problem is a directed graph such that the vertices are solutions of I , and an edge (s, s') exists if s' is a neighbor of s with a better objective value.*

► **Definition 2.4** (Tight PLS-reduction [29]). *A tight PLS-reduction from a local search problem P to a local search problem Q is a PLS-reduction (Φ, Ψ) from P to Q such that for every instance $I \in D_P$, we can choose a subset R of $F_Q(\Phi(I))$ that satisfies:*

1. *R contains all local optima of $\Phi(I)$;*
2. *For every solution $s_p \in F_P(I)$, we can construct in polynomial time a solution $s_q \in R$ so that $\Psi(I, s_q) = s_p$;*

3. If there is a path from $s_q \in R$ to $s'_q \in R$ in the transition graph $T_{\Phi(I)}$ such that there are no other solutions in R on the path, then either there is an edge from the solution $\Psi(I, s_q)$ to the solution $\Psi(I, s'_q)$ in the transition graph T_I or these two solutions are identical. A problem $Q \in \text{PLS}$ is tightly PLS-complete if for every problem $P \in \text{PLS}$, there exists a tight PLS-reduction from P to Q .

Note that tight PLS-reductions preserve the PSPACE-completeness of the standard algorithm problem and also the all-exp property (formalized below) [29].

► **Definition 2.5** (All-exp property). A local search problem P has the all-exp property if there are infinitely many pairs of an instance I of D_P and an initial solution $s \in F_P(I)$, for which the standard local search algorithm always needs an exponential number of iterations for all possible pivoting rules.

Parameterized Complexity. In parameterized complexity [5, 4], the complexity of a problem is studied not only with respect to the input size, but also with respect to some problem parameter(s). The input to a parameterized problem is a tuple (ω, κ) where ω is a string in a fixed alphabet and κ is a non-negative integer called the *parameter*. Typically, κ captures some property of a sought-after solution (e.g., an upper bound on the solution size), or a promised property of ω (e.g., an upper bound on the treewidth or clique-width of the input graph). A parameterized problem is *fixed-parameter tractable* if all instances where the promised property holds can be solved in time $f(\kappa) \cdot |\omega|^{\mathcal{O}(1)}$ for some computable function f .

► **Remark 2.6.** The above promise-based formulation can be avoided if one assumes that a witness for the structure captured by κ is provided on the input [4, page 137]; however, this is unreasonable if we wish to parameterize by the distance to the nearest optimal solution. Alternatively, one may formulate the parameter as a function ζ of ω [10], and all our results may equivalently be stated in this formulation – but only if one does not place restrictions on the running time required to compute ζ (see [2] for a discussion of this type of parameterization).

A well-established complexity assumption that we use for our lower bounds is that the class FPT of all fixed-parameter tractable parameterized decision problems is not equal to the class W[1]. In other words, it is considered unlikely for a W[1]-hard problem to be fixed-parameter tractable, and the existence of such an algorithm would – among others – violate the Exponential Time Hypothesis [4, Section 14.4].

3 Setting up the Framework

We start with the concept of pivoting formulation. Given an instance I of a local search problem, we define an *improving sequence* for I to be a sequence of solutions s_0, \dots, s_r such that for $i \in [r]$, s_i is a neighbor of s_{i-1} with a better objective value. An improving sequence is *maximal* if the last solution is locally optimal.

► **Definition 3.1.** For a local search problem \mathcal{P} , the corresponding pivoting problem $\mathcal{P}[\text{PIVOT}]$ is defined as follows: Given an instance I of \mathcal{P} and an initial solution s for I , output a maximal improving sequence for I starting from s .

Observe that for a local search problem, the task is to find a local optimum, without the restriction on the techniques used to achieve it. For the pivoting problem, we restrict ourselves to using only a local search algorithm (i.e., following a path in the transition graph). We can view it as finding the right pivoting rule to arrive at a local optimum, and hence the name “pivoting”.

For example, consider the following local search problem SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP: We are given a graph $G = (V, E)$, a weight function $\omega : V \cup E \rightarrow \mathbb{Q}$, a certifying function $\sigma : 2^{V \cup E} \rightarrow \{0, 1\}$ that certifies whether a subset of vertices and edges form a solution, and an objective function $f : 2^{V \cup E} \rightarrow \mathbb{Q}$ with $f(S) := \sum_{u \in S} \omega(u)$. Two solutions are neighbors if they differ by a c -swap (i.e., their symmetric difference is at most c). The task is to find a locally maximal solution (i.e., a solution $S \subseteq V \cup E$ such that for all neighboring solution S' of S , we have $f(S) \geq f(S')$). The corresponding pivoting problem is then defined as follows.

SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP[PIVOT]
Input: A graph $G = (V, E)$, a weight function $\omega : V \cup E \rightarrow \mathbb{Q}$, a certifying function $\sigma : 2^{V \cup E} \rightarrow \{0, 1\}$, and an initial solution S (i.e., $\sigma(S) = 1$).
Task: Output a maximal improving sequence starting from S .

We assume that ω and σ can be computed in time polynomial in $|V|$. Note that the graph G may be directed and/or a multi-graph. Further, although it is phrased here as a maximization problem, by reversing the signs of the images of ω , we can also model a minimization problem. We also assume c to be a constant so that the local search variant is in PLS (i.e., a polynomial time algorithm to compute a better neighbor exists).

SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP can be considered as a general problem that takes as special cases many well-known local search problems, such as the following.

- WEIGHTED INDEPENDENT SET/ c -SWAP (resp., WEIGHTED CLIQUE/ c -SWAP, WEIGHTED VERTEX COVER/ c -SWAP): In this case, the graph is vertex-weighted (i.e., the edge weights are zero); and a solution is an independent set (resp., a clique, a vertex cover).
- TRAVELING SALESMAN/ k -OPT: We have an edge-weighted graph, edge sets of spanning cycles are the solutions; and $c = 2k$ (i.e., a k -opt step is a $(2k)$ -swap).
- MAX CUT/FLIP with bounded maximum degree Δ : Here, the graph is edge-weighted (i.e., the vertex weights are zero); a solution is a set of vertices U together with all edges between U and $V \setminus U$; and the swap size is $c = 2 \cdot \Delta + 1$.

► **Remark 3.2.** To model exactly the flip-neighborhood of MAX CUT/FLIP, we add for each vertex $v \in V$ additionally Δ isolated vertices v_i with $i \in [1, \Delta]$ and enforce that each valid solution has to contain either all or none of $\{v\} \cup \{v_i \mid 1 \leq i \leq \Delta\}$ for each vertex $v \in V$. This ensures that the swap size of $c = 2 \cdot \Delta + 1$ does not allow us to swap more than one vertex together with its associated isolated vertices. Thus, two neighboring solutions differ by at most one original vertex, which then allows to model exactly the flips of single vertices in MAX CUT/FLIP.

The next local search problem we consider is WEIGHTED CIRCUIT/FLIP, a generalization of the classical CIRCUIT/FLIP. In this problem, we are given a Boolean circuit D with n input nodes x_1, \dots, x_n and m output nodes y_1, \dots, y_m and a weight function $\omega : [m] \rightarrow \mathbb{Q}$. The goal is to find a locally maximal input for the optimization function $f(x_1, \dots, x_n) = \sum_{i=1}^m \omega(i) y_i$ and 1-swap neighborhood (which we also refer to as a *flip*). The corresponding pivoting problem is as follows.

WEIGHTED CIRCUIT/FLIP[PIVOT]
Input: A Boolean circuit D with n input nodes x_1, \dots, x_n and m output nodes y_1, \dots, y_m , a weight function $\omega : [m] \rightarrow \mathbb{Q}$, and an initial solution $s \in \{0, 1\}^n$.
Task: Output a maximal improving sequence starting from s .

Note that when $\omega(i) = 2^{i-1}$ (i.e., we interpret $f(x_1, \dots, x_n)$ as the number $y_m \dots y_1$ written in binary), then we recover the classical MAX-CIRCUIT/FLIP. Similarly, if we set $\omega(i) = -2^{i-1}$, then we have the MIN-CIRCUIT/FLIP. Further, this problem also captures WEIGHTED MAX-SAT/FLIP.

In our parameterized analysis of the above problems, we assume that the input is additionally equipped with an integer parameter κ (see Section 2).

4 Hardness results for Weighted Independent Set/3-Swap

Recall that the parameter ℓ is the distance of the given starting solution to the nearest local optimum in the transition graph. Observe that the time complexity of a pivoting problem is lower-bounded by the encoding length of the shortest maximal improving sequence from the initial solution (i.e., ℓ times the encoding size of a solution). Hence, it is natural to consider the parameterized complexity of such a problem with respect to ℓ .

This section is dedicated to prove the following theorem.

► **Theorem 4.1.** *Unless $\text{FPT} = \text{W}[1]$, there does not exist an algorithm to solve WEIGHTED INDEPENDENT SET/3-SWAP[PIVOT] in FPT-time when parameterized by ℓ (i.e., in time $\mathcal{O}(\ell) \cdot n^{\mathcal{O}(1)}$ for some computable function g).*

Note that since WEIGHTED INDEPENDENT SET/3-SWAP[PIVOT] is a special case of SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP[PIVOT], the hardness result above also extends to the latter problem.

► **Remark 4.2.** Theorem 4.1 is tight in the sense that for any instance of SUBSET WEIGHT OPTIMIZATION PROBLEM/2-SWAP[PIVOT] with all positive weights or all negative weights, the output always has length $\mathcal{O}(n^2)$. Indeed, suppose that all weights are positive; the argument for negative weights is analogous. We label the vertices as v_1, \dots, v_n in the increasing order of their weights. For every solution R , define its potential $p(R) := \sum_{v_i \in R} i$. Then it is easy to see that every 2-swap increases the potential. Since the potential only has $\mathcal{O}(n^2)$ possible values, every improving sequence must have length $\mathcal{O}(n^2)$ as well.

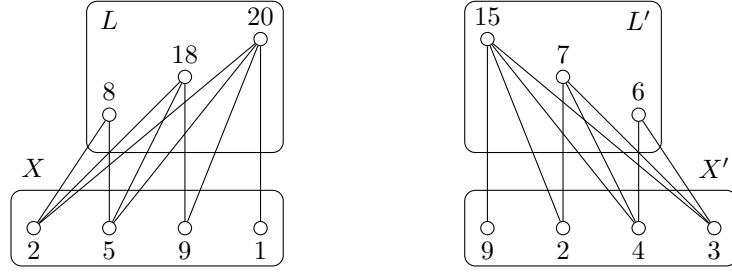
The proof of Theorem 4.1 makes use of the following result.

► **Theorem 4.3.** *There is a tight PLS-reduction from MAX-CUT/FLIP with bounded degree to WEIGHTED INDEPENDENT SET/3-SWAP. In particular, the latter problem has the all-exp property (even when restricting to positive weights).*

WEIGHTED INDEPENDENT SET/3-SWAP has been shown to be PLS-complete [18] via a PLS-reduction from MAX CUT/FLIP. However, this reduction was not argued to be tight, and is in fact unlikely to be. Here, we adapt it into a tight PLS-reduction; the proof is presented in Section 4.2.

The constructions in the proofs of Theorems 4.1 and 4.3 use the concept of (up/down)-elevators, defined as follows. Here, when we assume a certain (fixed) ordering on a set X , we denote by $X[i]$ the i th element in this ordering.

Let $G = (V, E)$ be a vertex-weighted graph and let $X \subseteq V$ be a set of size at least 2. An X -elevator is a clique $L := \{\ell_1, \dots, \ell_s\}$ of size $s := |X| - 1$, such that the vertices of L have the same neighborhood outside of $L \cup X$ and for each $i \in [1, s - 1]$, ℓ_i has exactly the first $i + 1$ vertices of X as neighbors inside of X for an arbitrary but fixed ordering on the vertices of X . We call the vertices of an elevator *levels*. The vertex $\ell_{|L|}$ is the *top-level*, while ℓ_1 is the *bottom-level*.



■ **Figure 2** An example for the elevator gadgets. On the left an X -up-elevator L and on the right an X' -down-elevator L' . The edges of the cliques L and L' are not depicted. Moreover, the edges between the vertices of X or the vertices of X' are arbitrary. Recall that the vertices of L (respectively L') have the same neighborhood outside of $L \cup X$ (respectively $L' \cup X'$).

We say that L is an *up-elevator* if $\omega(\ell_1) = \omega(X[1]) + \omega(X[2]) + 1$ and for each $i \in [2, s-1]$, $\omega(\ell_i) = \omega(\ell_{i-1}) + \omega(X[i+1]) + 1$. Similarly, we say that L is a *down-elevator* if $\omega(\ell_1) = \omega(x_1) + \omega(x_2) - 1$ and for each $i \in [2, s-1]$, $\omega(\ell_i) = \omega(\ell_{i-1}) + \omega(X[i+1]) - 1$. Note that the weights of the vertices in such an up- or down-elevator are uniquely defined by the weights and the order of the vertices of X . See Figure 2 for examples of an up-elevator and a down-elevator.

► **Lemma 4.4.** *Let (G, ω) be a weighted graph. Let $L = \{\ell_1, \dots, \ell_s\}$ be an X -up-elevator for some vertex set X of G , where ℓ_s is the top-level. If an independent set S contains ℓ_i for some $i \neq s$, then $(S \setminus \{\ell_i, X[i+2]\}) \cup \{\ell_{i+1}\}$ is an independent set with a higher weight than that of S .*

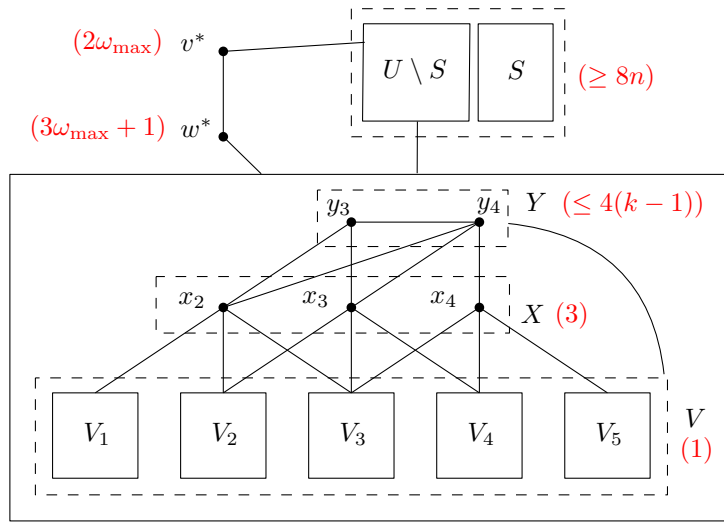
Proof. By definition, $\omega(\ell_{i+1}) > \omega(\ell_i) + \omega(X[i+2])$. Further, $N(\ell_{i+1}) = N(\ell_i) \cup \{X[i+2]\}$. The lemma then follows. ◀

4.1 Proof of Theorem 4.1

We reduce from the well-known $W[1]$ -hard MULTI-COLORED INDEPENDENT SET problem: Given a graph $G = (V_1 \cup \dots \cup V_k, E)$ where $k \geq 3$ and V_i is a clique for each $i \in [k]$, is there a independent set of size k ? Without loss of generality, we assume $|V_k| = 1$. Let $V := V_1 \cup \dots \cup V_k$ and $n := |V|$.

By Theorem 4.3, there exists an instance (U, F, ω) of WEIGHTED INDEPENDENT SET/3-SWAP with $\Omega(n)$ vertices such that all local optima are exponentially far away from some initial solution $S \subseteq U$ in the transition graph of (U, F, ω) ; moreover, such an instance can be constructed in polynomial time via the provided reduction. Assume without loss of generality that each weight of ω is divisible by $8n$ and let ω_{\max} denote the largest assigned weight.

Now we construct an instance of WEIGHTED INDEPENDENT SET/3-SWAP[PIVOT] consisting of a graph $G' := (V', E')$, weight function ω' , and an initial solution T as follows. Initialize G' as the disjoint union of G and (U, F) . Each vertex in V has weight one, and each vertex in U carries the same weight as assigned by ω . Next, add a set X of $k-2$ vertices x_i with $i \in [2, k-1]$. Each such vertex has weight 3. For each $i \in [2, k-1]$, make x_i adjacent to all vertices of $V_{i-1} \cup V_i \cup V_{i+1}$. Next, add an up-elevator Y of X , where we consider the ordering (x_2, \dots, x_{k-1}) of the vertices in X . We label the vertices in Y as y_3, \dots, y_{k-1} such that the neighborhood of y_i in X is exactly $\{x_j \mid j \in [2, i]\}$. Moreover, we add two vertices v^* and w^* of weight $2\omega_{\max}$ and $3\omega_{\max} + 1$, respectively. Then, we add the edge $\{v^*, w^*\}$ and make w^* adjacent to all vertices of $V \cup X \cup Y$. Finally, we make each vertex of $V \cup X \cup Y \cup \{v^*\}$ adjacent to each vertex of $U \setminus S$. See Figure 3 for an illustration.



■ **Figure 3** Illustration of G' in the proof of Theorem 4.1 for $k = 5$. Not all edges are depicted. An edge between a box and a vertex/box indicate a complete bipartite subgraph between the vertices in the former box and the latter vertex/box. Red numbers in brackets indicate the weights.

The initial solution T is then defined by $S \cup V_k \cup \{v^*\}$. Recall that V_k contains only a single vertex and that this vertex is part of every multi-colored independent set of size k in G .

▷ **Claim 4.5.** Let π be a maximal improving sequence for (G', ω') from T . If π contains a solution \bar{T} such that $\bar{T} \cap V$ is an independent set of size k , then \bar{T} is locally optimal. Otherwise, π has length exponential in n .

Proof. We start with the following observations:

1. There is no improving swap that removes w^* .
2. If w^* is part of a solution, then the solution is a subset of $U \cup \{w^*\}$.
3. Each reachable solution from T contains either v^* or w^* .

Indeed, Item 1 follows from the fact that w^* has weight $3\omega_{\max} + 1$, which is strictly larger than the sum of the next two highest weights assigned by ω' . Thus, no improving 3-swap can remove vertex w^* from a given solution. Next, Item 2 follows from the fact that w^* is a neighbor of each other vertex of $V' \setminus U$. Finally, Item 3 follows from Item 1 and the fact that each reachable solution from T that does not contain w^* , has to contain v^* , since v^* is contained in T and the weight of v^* is at least the weight of any two vertices besides $\{v^*, w^*\}$. In other words, each improving swap that removes v^* from the current solution, has to add w^* to the solution.

Now suppose π has a solution \bar{T} such that $\bar{T} \cap V = \{v_1, \dots, v_k\}$ is an independent set. Without loss of generality, assume that $v_i \in V_i$ for $i \in [k]$. For the sake of contradiction, assume that \bar{T} is not locally optimal and there is an improving 3-swap W . Since the swap is improving, at least one vertex is added by this swap. Note that this swap cannot add any vertex of $Y \cup \{w^*\} \cup (U \setminus S)$ to the solution, since each such vertex is adjacent to v_1, v_2 , and v_3 , and at most two of these vertices can be removed by W . Similarly, W cannot add any vertex x_i to the solution, since x_i is adjacent to v_{i-1}, v_i , and v_{i+1} and at most two of these vertices can be removed by W . Further, if v^* is not in \bar{T} then it cannot be added by W , due to Items 1 and 3. Hence, the only vertices W could add are from V . However, if W adds a vertex in V_i for some $i \in [k - 1]$, it needs to remove v_i and we cannot add other

vertices from V_i , since V_i is a clique. This implies that W can only add at most one vertex in V and remove at least one vertex in V . Since, vertices in V have the same weights and vertices in $V' \setminus V$ have positive weights and cannot be added by W , W cannot be improving, a contradiction. Hence, \bar{T} is locally optimal.

For the remainder of the proof, we now assume that π does not have such a solution \bar{T} . We show that if π contains the solution $S \cup \{w^*\}$, then it has length exponential in n . Indeed, by Items 1 and 2, the subsequence π' of π starting from $S \cup \{v^*\}$ is a sequence of improving 3-swaps for (U, F, ω) starting from S . Since S is far away from all the local optimum of (U, F, ω) , the sequence π' has exponential length in n as claimed.

Hence, it suffices to show that π must contain the solution $S \cup \{w^*\}$. For each $i \in [1, |\pi|]$, let $T_i := \pi(i)$ denote the i th solution in this sequence. By Item 3, for each $i \in [1, |\pi|]$, T_i contains either v^* and w^* . Moreover, by Item 1, the solutions in π containing v^* are a prefix of π . Let $i \in [1, |\pi|]$, such that $v^* \in T_i$. Observe that this implies that $T_i \cap U = S$. This is due to the following facts:

- $v^* \in T_i$, which implies that $T_i \cap U \subseteq S$, since v^* is a neighbor of each vertex of $U \setminus S$.
- Since Y is a clique, T_i has at most one vertex in Y , whose weight is less than $4k$. Hence, the total weight of all vertices in $T_i \cap (X \cup Y \cup V)$ is less than $8n$, and hence less than the weight of each individual vertex of U .

That is, $T_i \cap U$ contains only vertices of S , and cannot miss any vertex of S , as otherwise, the weight of T_i would be strictly less than the weight of T , which would then contradict the fact that T_i is part of an improving sequence starting with T .

Note that this further implies that T_i contains at least one vertex of $V \cup X \cup Y$, as otherwise, $T_i = S \cup \{v^*\}$, which has strictly less weight than $T = S \cup \{v^*\} \cup V_k$. Combined with Item 2, this then implies that if T_{i+1} exists and contains w^* , the required 3-swap (i) adds w^* to the solution and (ii) removes v^* and a unique vertex of $V \cup X \cup Y$ from T_i . Hence, if T_{i+1} exists and contains w^* , then $T_{i+1} \cap U = T_i \cap U = S$, which implies that $T_{i+1} = S \cup \{w^*\}$.

Consequently, to finish the proof, it suffices to show that T_i is not locally optimal. Since π is a sequence of finite length, this then implies that there is some smallest j , such that T_j contains w^* and moreover fulfills $T_j = S \cup \{w^*\}$ by the above argumentation. To show that T_i is not locally optimal, we distinguish several cases depending on the intersection of T_i with $V \cup X \cup Y$. For each of the cases, we present an improving 3-swap. Let $T'_i := T_i \cap (V \cup X \cup Y)$ and note that $T_i = S \cup \{v^*\} \cup T'_i$.

Case 1. $T'_i \subseteq V$. Recall that for each $j \in [k]$, V_j is a clique in G' , which implies that $|T'_i \cap V_j| \leq 1$. Since $T_i \cap V$ is not an independent set of size k , there is some $p \in [2, k-1]$, such that $|T'_i \cap (V_{p-1} \cup V_p \cup V_{p+1})| \leq 2$. Hence, the swap adding the vertex x_p to the solution while removing the at most two vertices from $V_{p-1} \cup V_p \cup V_{p+1}$ is an improving 3-swap.

Case 2. $T'_i \subseteq V \cup X$ with $T'_i \cap X \neq \emptyset$. If $T'_i \cap X = X$, then $T'_i = X$ and (i) adding y_3 to the solution and (ii) removing x_2 and x_3 from the solution, yields an improving 3-swap. Otherwise, there is some $p \in [2, k-2]$, such that either $x_p \in T'_i$ and $x_{p+1} \notin T'_i$ or vice versa. In both cases, $T'_i \cap (V_p \cup V_{p+1}) = \emptyset$, since x_p and x_{p+1} are both adjacent to each vertex of V_p and V_{p+1} . Assume without loss of generality that $x_p \in T'_i$ and $x_{p+1} \notin T'_i$. Then, (i) adding x_{p+1} to the solution and (ii) removing at most one vertex of V_{p+2} from the solution yields an improving 3-swap.

Case 3. *There is some $p \in [3, k-1]$, such that $y_p \in T'_i$.* If $p < k-1$, then there exists an improving 3-swap by Lemma 4.4. Otherwise, observe that y_{k-1} is the neighbor of all vertices in $V \cup X \cup Y$. Hence, $T_i \cap (V \cup X \cup Y) = \{y_{k-1}\}$. Then adding w^* and removing v^* and y_{k-1} yields an improving 3-swap.

Hence, if G does not contain an independent set of size k , then $S \cup \{w^*\}$ is a solution which is part of every sequence of improving 3-swaps starting at T . This completes the proof of the claim. \triangleleft

Note that the claim above implies that if G has an independent set Q of size k , then there is a maximal improving sequence starting from T with length at most k . Indeed, we add all the vertices of Q to the initial solution by $k - 1$ consecutive 1-swaps. Then we arrive at a solution \bar{T} such that $\bar{T} \cap V = Q$. Hence, \bar{T} is locally optimal by Claim 4.5.

Now suppose for the sake of contradiction that there exists an algorithm to solve WEIGHTED INDEPENDENT SET/3-SWAP[PIVOT] in time $g(\ell) \cdot n^{\mathcal{O}(1)}$ for some computable function g . We set $\ell = k$, and simulate the algorithm on the instance (G', ω', T) above for $g(\ell) \cdot n^{\mathcal{O}(1)}$ steps. If the algorithm returns a solution T^* such that $T^* \cap V$ is an independent set of size k , then we know that G is a YES-instance of MULTI-COLORED INDEPENDENT SET. Otherwise, by Claim 4.5, since the running time $g(\ell) \cdot n^{\mathcal{O}(1)}$ is not exponential in n , the algorithm should not return any locally optimal solution; that is, it either returns a solution that is not locally optimal or does not terminate within this time. In this case, we know that the promise that there is a local optimum within k improving steps from T is broken. As argued above, this implies that G does not have an independent set of size k , and hence (G, k) is a NO-instance. Therefore, we can decide MULTI-COLORED INDEPENDENT SET in time $g(k) \cdot n^{\mathcal{O}(1)}$, a contradiction to the assumption $\text{FPT} \neq \text{W}[1]$. \blacktriangleleft

► **Remark 4.6.** Our proof of Theorem 4.1 above also implies the following:

- It is $\text{W}[1]$ -hard to decide whether there is a local optimum of distance at most k from a given initial solution, when parameterized by k .
- Unless $\text{FPT} = \text{W}[1]$, there is no algorithm to approximate the distance to the nearest local optimum with running time $g(\ell) \cdot n^{\mathcal{O}(1)}$ and approximation ratio $n^{\mathcal{O}(1)}$.

4.2 All-exp Property of Weighted Independent Set/3-swap

The main argument to show the all-exp property of WEIGHTED INDEPENDENT SET/3-SWAP is the following tight PLS-reduction from MAX CUT/FLIP (Theorem 4.7 below). This reduction is a slight adaptation to the PLS-reduction presented by Komusiewicz and Morawietz [18]. Roughly speaking, the reduction constructs an instance with two parts C and D where the vertices of C model the respective solutions of MAX CUT and the vertices of D are used as gadgets to ensure that we can simulate the flip of a vertex in the MAX CUT instance by a sequence of improving 3-swaps. In the original reduction by Komusiewicz and Morawietz, some maximal independent sets can be improved by only swapping vertices in C , thus leading to intermediate independent sets that are not maximal. Such independent sets could be improved by 1-swaps which could be combined with improving 1-swaps or 2-swaps in distant parts of the graph. Consequently, there is not a strong correspondence between improving flips in the MAX CUT instance and improving swaps in the INDEPENDENT SET instance. To establish this correspondence, we add further vertices to C , thus ensuring that each maximal solution can only be improved by swapping at least one vertex of D into the solution. By turning D into a clique, we now ensure that at each time step the flip of only a single vertex in the MAX CUT instance can be simulated.

► **Theorem 4.7** (\star). *There exists a tight PLS-reduction from MAX CUT/FLIP restricted to instances (G, ω) of maximum degree at most $(\log |V(G)|)^{\mathcal{O}(1)}$ to WEIGHTED INDEPENDENT SET/3-SWAP.*

The proof is deferred to the related full version.

5 A new notion of reduction

In this section, we define a new notion of PLS-reduction that transfers the result in Theorem 4.1 to other problems. We also demonstrate it by describing such a reduction from WEIGHTED INDEPENDENT SET/3-SWAP[PIVOT] to MAX-CIRCUIT/FLIP[PIVOT], which is a special case of WEIGHTED CIRCUIT/FLIP[PIVOT].

5.1 ℓ -tight PLS-reduction

► **Definition 5.1** (ℓ -tight PLS-reduction). *An ℓ -tight PLS-reduction from a problem $P \in \text{PLS}$ to a problem $Q \in \text{PLS}$ is a PLS-reduction (Φ, Ψ) from P to Q such that for every instance $I \in D_P$, we can choose a subset R of $F_Q(\Phi(I))$ that satisfies:*

1. R contains all local optima of $\Phi(I)$;
2. For every solution $s_p \in F_P(I)$, we can construct in polynomial time a solution $s_q \in R$ so that $\Psi(I, s_q) = s_p$;
3. If there is a path from $s_q \in R$ to $s'_q \in R$ in the transition graph $T_{\Phi(I)}$ such that there are no other solutions in R on the path, then either there is an edge from the solution $\Psi(I, s_q)$ to the solution $\Psi(I, s'_q)$ in the transition graph T_I or these two solutions are identical;
4. For two solutions $s_q \in R$ and $s'_q \in R$ such that either $\Psi(I, s_q) = \Psi(I, s'_q)$ or there is an edge $(\Psi(I, s_q), \Psi(I, s'_q))$ in the transition graph T_I , the shortest path from s_q to s'_q is at most ℓ .

Note that the first three conditions are identical to the tight PLS-reduction in Definition 2.4. While tight PLS-reduction forbids introduction of shortcuts in the transition graph, the distance between two solutions can be arbitrarily increased. Our new condition 4 limits this increase.

► **Theorem 5.2.** *Let \mathcal{P} and \mathcal{P}' be two local search problems in PLS. Suppose there exists an $\varphi(\ell)$ -tight PLS-reduction from \mathcal{P} to \mathcal{P}' for some integer ℓ and polynomial function φ . If there exists an algorithm to solve $\mathcal{P}'[\text{PIVOT}]$ in FPT-time when parameterized by ℓ , then there also exists such an algorithm to solve $\mathcal{P}[\text{PIVOT}]$.*

Proof. Suppose there exists an algorithm A' that solves $\mathcal{P}'[\text{PIVOT}]$ in time $g(\ell) \cdot n^{\mathcal{O}(1)}$ for some computable function g . Let (I, s) be an instance of $\mathcal{P}[\text{PIVOT}]$ (i.e., I is an instance of \mathcal{P} , s a solution of I), and ℓ be the promised distance from s to a local optimum. Let (Φ, Ψ) be the $\varphi(\ell)$ -tight PLS-reduction given in the lemma statement. Let R be the set as guaranteed by Definition 5.1 for the reduction (Φ, Ψ) and instance I of \mathcal{P} . Note that since (Φ, Ψ) is a PLS-reduction, in time polynomial in $|I|$, we can obtain the solution $I' := \Phi(I)$ of \mathcal{P}' . By Condition 2 of Definition 5.1, in polynomial time, we can also obtain solution $s' \in R$ such that $\Psi(I, s') = s$. Note that if the promise is not violated, then by Condition 4 of Definition 5.1, there exists a path from s' to a local optimum of I' of length at most $\varphi(\ell)\ell$. Then using the algorithm A' , we can find such a path in time $g(\varphi(\ell)\ell) \cdot |I'|^{\mathcal{O}(1)}$. Note that the start and end of this path are in R , and the end of this path corresponds to a local optimum of I . Hence, applying Condition 3 of Definition 5.1, this corresponds to a path of length at most $\varphi(\ell)\ell$ from s to a local optimum of I . In other words, we can solve (I, s, ℓ) in time $g'(\ell) \cdot |I|^{\mathcal{O}(1)}$ for some computable function g' . ◀

► **Remark 5.3.** The reduction in the proof of Theorem 4.7 is an $\mathcal{O}(\Delta)$ -tight PLS reduction.

5.2 Reduction to Max-Circuit/Flip

Having formalized ℓ -tight PLS-reductions and established their ability to transfer the desired lower bound, we proceed to our showcase of how this reduction can be applied to local search problems beyond SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP.

► **Theorem 5.4.** *There is an $\mathcal{O}(c)$ -tight PLS reduction from SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP to MAX-CIRCUIT/FLIP.*

Proof. We adapt the proof of PLS-reduction from any problem in PLS to MIN-CIRCUIT/FLIP in [24]. Let $\mathcal{I} = (G, \omega, \sigma)$ be an instance of SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP. Let $G = (V, E)$ and $n := |V| + |E|$. We now construct a Boolean circuit D as follows. Note that we can encode any solution of \mathcal{I} in n bits, where each bit is associated with a vertex/edge in $V \cup E$ and is 1 if and only if that vertex/edge is in the solution. For a length- n string u that encodes a solution s of \mathcal{I} , let $\omega(u) := \omega(s)$, and let $B(u)$ be the set of strings encoding the solutions that can be obtained by an improving c -swap from s . For two strings u and w , let $H(u, w)$ be the Hamming distance between u and w , and let $V(u, w)$ be the set of $H(u, w)$ strings that are obtained if we change u into w by flipping the bits in which u differs from w in increasing order of their positions. We define a function $h : \{0, 1\}^{V \cup E} \rightarrow \mathbb{Q}$ as follows. For a length- n string u that encodes a solution of \mathcal{I} , we define the value of h for some *structured* strings below:

- $h(uu00) = (2n + 4)\omega(u)$,
- $h(uv00) = (2n + 4)\omega(u) + H(u, v)$, where v is a string in $V(u, w)$ for some w in $B(u)$,
- $h(uw10) = (2n + 4)\omega(u) + n + 1$, for $w \in B(u)$,
- $h(vu11) = (2n + 4)\omega(u) - H(u, v) - 2$, where v is a string in $V(w, u)$ for some w such that $u \in B(w)$, and
- $h(uu10) = (2n + 4)\omega(u) - 1$.

For an unstructured string x (i.e., a string that is not of any form above), $h(x)$ has a value equal to μ minus the number of ones in x , where μ is smaller than any value for any structured string. Then the Boolean circuit D is the one with $2n + 2$ input nodes and computes the value $h(x)$ for a string x of length $2n + 2$. Such a circuit with polynomial size exists, as h can be computed in polynomial time [24, Theorem 6.3].

The description above constitutes the function Φ that maps an instance \mathcal{I} of SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP to an instance D of MAX-CIRCUIT/FLIP. The function Ψ that maps a solution of D to that of \mathcal{I} is as follows: For a structured string x of the form $uv00$ or $uv10$, $\Psi(\mathcal{I}, x) = u$ (u and v may be identical). For a structured string x of the form $vu11$, we assign $\Psi(\mathcal{I}, x) = u$. Lastly, if x is unstructured, $\Psi(\mathcal{I}, x) = 0 \dots 0$.

It is easy to see that unstructured strings cannot be a local optimum. Further, among the structured strings, only the string of the form $uu00$ can be locally optimal, and this is if and only if u encodes a local optimum of \mathcal{I} . Hence, (Φ, Ψ) is a PLS-reduction.

Now, consider the set R as the set of all structured strings. We argue that the set R satisfies the conditions in Definition 5.1. For the first condition, as argued above, only structured strings can be a local optimum of D , and hence R contains all local optima. For the second condition, we map every solution u of \mathcal{I} to the solution $uu00$ in R . For the third condition, observe that an improvement step from a structured string can yield only a structured string. Hence, we only need to consider two adjacent structured strings x_1 and x_2 such that there is an edge from x_1 to x_2 in the transition graph T_D . By the choice of the function Ψ and by construction, it follows that $\Psi(\mathcal{I}, x_1)$ and $\Psi(\mathcal{I}, x_2)$ are either identical or there is an edge from $\Psi(\mathcal{I}, x_1)$ to $\Psi(\mathcal{I}, x_2)$ in the transition graph $T_{\mathcal{I}}$. Hence, the third condition is satisfied. Lastly, consider a length- n string u . The structured strings x such

that $\Psi(\mathcal{I}, x) = u$ either have the form $uv00$, $uv10$, or $vu11$ for some v such that $H(u, v) \leq c$. Hence, for any two structured strings x_1 and x_2 such that either $\Psi(\mathcal{I}, x_1) = \Psi(\mathcal{I}, x_2)$, we must have $H(x_1, x_2) \leq 2c + 2$. Similarly, if there is an edge $(\Psi(\mathcal{I}, x_1), \Psi(\mathcal{I}, x_2))$ in $T_{\mathcal{I}}$, then we must have $H(x_1, x_2) \leq 4c + 4$.

Overall, this shows that (Φ, Ψ) is a $(4c + 4)$ -tight PLS-reduction. \blacktriangleleft

Combined with Theorems 4.1 and 5.2, the theorem above immediately implies the following.

► **Corollary 5.5.** *Unless $\text{FPT} = \text{W}[1]$, there does not exist an algorithm to solve $\text{MAX-CIRCUIT/FLIP}[PIVOT]$ in $g(\ell) \cdot n^{\mathcal{O}(1)}$ time for any computable function g .*

6 Fixed-parameter Algorithms

In this section, we establish the algorithmic upper bounds that form the foundation of Main Finding 1. We recall that all results obtained in this section can also be directly translated to the PLS and standard algorithm problem formulations.

6.1 Subset Weight Optimization Problem/ c -Swap

In this section, we show algorithmic results for problems in $\text{SUBSET WEIGHT OPTIMIZATION PROBLEM}/c\text{-SWAP}[PIVOT]$ parameterized by the shortest distance ℓ , swap size c and the number k of distinct weights. We start with an XP-algorithm parameterized by ℓ and c .

► **Theorem 6.1.** *$\text{SUBSET WEIGHT OPTIMIZATION PROBLEM}/c\text{-SWAP}[PIVOT]$ can be solved in time $\mathcal{O}(n^{c\ell})$, where n is the instance size.*

Proof. Consider an instance (G, ω) of a problem in $\text{SUBSET WEIGHT OPTIMIZATION PROBLEM}/c\text{-SWAP}[PIVOT]$. Let $n := |V(G)| + |E(G)|$. By definition, any solution of \mathcal{I} has at most $\mathcal{O}(n^c)$ neighbors in the transition graph. Hence, for any solution Q of \mathcal{I} , there are at most $\mathcal{O}(n^{c\ell})$ solutions that are reachable from Q via a path of the length at most ℓ in the transition graph. Exploring these solutions in a depth-first search fashion, we obtain an algorithm with the claimed running time. \blacktriangleleft

Next, we consider the case when both the swap size and the number of distinct weights are bounded in the following theorem. We start with a straightforward XP-algorithm for this parameter following a simple observation below.

► **Observation 6.2.** *For an instance I of a problem $\mathcal{P} \in \text{PLS}$ and a solution s of I , let ℓ_{\max} be the longest distance from s to a local optimum. Then the instance (I, s) of $\mathcal{P}[PIVOT]$ can be solved in $\ell_{\max} \cdot n^{\mathcal{O}(1)}$.*

Proof. Since \mathcal{P} is in PLS, for any solution s' , we can conclude that it is locally optimal or find a better solution in polynomial time. Since any improving sequence from s has length at most ℓ_{\max} , the statement then follows. \blacktriangleleft

► **Corollary 6.3.** *$\text{SUBSET WEIGHT OPTIMIZATION PROBLEM}/c\text{-SWAP}[PIVOT]$ can be solved in time $\mathcal{O}(n^{2k+c})$, where n is the number of vertices and k is the number of distinct weights.*

Proof. It is easy to see that given a set R of $\mathcal{O}(n^2)$ numbers among k different choices of values, the set $\{\sum s \in S \mid S \subseteq R\}$ has $\mathcal{O}(n^{2k})$ elements. This implies that there are $\mathcal{O}(n^{2k})$ possible objective values for the $\text{SUBSET WEIGHT OPTIMIZATION PROBLEM}/c\text{-SWAP}[PIVOT]$ instance. Since all solutions in an improving sequence must have pairwise distinct objective

values, it follows that the length of the sequence is $\mathcal{O}(n^{2k})$. Combined with Observation 6.2, the corollary then follows by the fact that it takes $\mathcal{O}(n^c)$ time to find a better solution in the local neighborhood, if one exists. \blacktriangleleft

Next, we show an FPT-algorithm (Corollary 6.6 below) based on the following result, which already had a big impact on kernelization results for weighted optimization problems [6].

► **Theorem 6.4** (Frank and Tardos [11]). *Given a rational vector $w \in \mathbb{Q}^k$ and an integer N , there is a polynomial-time algorithm to output an integral vector $\bar{w} \in \mathbb{Z}^k$ such that $|\bar{w}|_\infty \leq 2^{4k^3} N^{k(k+2)}$ and $\text{sign}(w^\top b) = \text{sign}(\bar{w}^\top b)$ for any integer vector b with $|b|_1 \leq N - 1$.*

► **Theorem 6.5.** *For SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP with k different values of weights and for any pivoting rule, the standard local search algorithm takes $g(c, k) \cdot n^{\mathcal{O}(1)}$ steps for some computable function g .*

Proof. Applying Theorem 6.4 with $N = c + 1$ and $w = (w_1, \dots, w_k)$ being the vector of the k different weights, we obtain a vector $\bar{w} = (\bar{w}_1, \dots, \bar{w}_k)$ such that (i) $|\bar{w}|_\infty \leq g(c, k)$ for some computable function g and (ii) $\text{sign}(w^\top b) = \text{sign}(\bar{w}^\top b)$ for any integer vector b with $|b|_1 \leq c$. Let \mathcal{I} be the original instance and $\bar{\mathcal{I}}$ be the instance obtained from \mathcal{I} by replacing any weight with value w_i by \bar{w}_i for $i \in [k]$. (i) implies that the absolute value of the objective for $\bar{\mathcal{I}}$ is bounded by $n^{\mathcal{O}(1)}g(c, k)$. Since the weights are integral, the objective improves by at least 1 at each step. Hence, the standard local search algorithm for $\bar{\mathcal{I}}$ should terminate in $n^{\mathcal{O}(1)}g(c, k)$ steps for any pivoting rule. Next, (ii) implies that the transition graph is the same for both \mathcal{I} and $\bar{\mathcal{I}}$. As such, a local optimum for $\bar{\mathcal{I}}$ is also one for \mathcal{I} . The theorem then follows. \blacktriangleleft

Since we can find an improving swap in $n^{\mathcal{O}(c)}$, Theorem 6.5 immediately implies the following FPT result.

► **Corollary 6.6.** *SUBSET WEIGHT OPTIMIZATION PROBLEM/ c -SWAP[PIVOT] with k different values of weights can be solved in time $g(c, k) \cdot n^{\mathcal{O}(c)}$ for some computable function g .*

Note that we cannot presumably replace the $\mathcal{O}(c)$ in the exponent by a constant. This is due to the fact that it is W[1]-hard [9] for parameter c to decide whether a given independent set in an unweighted graph is locally optimal. Moreover, as a consequence from the reduction behind [17, Theorem 3.7], the problem cannot be solved in $g(c) \cdot n^{o(c)}$ time, unless the ETH fails.

6.2 Generalized-Circuit/Flip

The same proof techniques in the previous section can also be applied to GENERALIZED-CIRCUIT/FLIP[PIVOT]. First, we obtain a similar XP result when parameterizing the problem by the shortest distance ℓ .

► **Theorem 6.7.** *An instance of GENERALIZED-CIRCUIT/FLIP[PIVOT] with n input gates can be solved in time $\mathcal{O}(n^\ell)$.*

Proof. Every solution has at most n neighbors in the transition graph. Hence, exhaustive search of all solutions reachable from the initial solution by a directed path of length at most ℓ in the transition graph runs in time $\mathcal{O}(n^\ell)$. \blacktriangleleft

Note that there are only 2^m states of the output, and in any directed path in the transition graph, we cannot encounter any state twice. This implies that the distance from a solution to any reachable local optimum is at most 2^m (i.e., $\ell < 2^m$). While the result above then implies an XP-algorithm when parameterizing by m , we can instead immediately obtain an FPT-algorithm from Observation 6.2.

► **Corollary 6.8.** *An instance of GENERALIZED-CIRCUIT/FLIP[PIVOT] with n input gates and m output gates can be solved in time $2^m n^{\mathcal{O}(1)}$.*

If we parameterize by the number k of distinct weights and an additional parameter t defined in Theorem 6.9 below, we again obtain that any pivoting rule yields a fixed-parameter algorithm.

► **Theorem 6.9.** *Let \mathcal{I} be an instance of GENERALIZED-CIRCUIT/FLIP[PIVOT] with independently chosen weights among k different values. Further, let t be the maximum number of output gates connected to any particular input gate. Then for any pivoting rule, the standard local search algorithm takes $f(t, k) \cdot n^{\mathcal{O}(1)}$ steps.*

Proof. The proof is analogous to that of Theorem 6.5, when we replace c by t . ◀

7 Concluding Remarks

There are numerous avenues for future research. First, one could identify further local search problems for which the pivoting formulation parameterized by the distance ℓ of the starting solution to the closest local optimum is not FPT. Conversely, it is open whether there is any *natural* PLS-complete problem whose pivoting formulation admits a fixed-parameter algorithm when parameterized by ℓ . Observe that such a fixed-parameter algorithm exists for PLS-complete local search problems where the largest neighborhood has constant size or – slightly more generally – the maximum outdegree in the transition graph is constant. Such problems do exist, for example one may define an ordering of solutions and redefine the neighborhood to only contain the smallest solution with respect to this ordering. We are, however, not aware of any natural PLS-complete problem with only constant-size neighborhoods. In light of this discussion it seems more meaningful to ask whether there is any PLS-complete problem with an unbounded number of large neighborhoods whose pivoting formulation is FPT with respect to ℓ .

In addition to ℓ , other parameters related to the transition graph could be considered. For example, since parameterization by the diameter of the transition graph or the distance to the furthest local optimum trivially yield fixed-parameter algorithms, one could consider parameters that are sandwiched between ℓ and the diameter. By the discussion above, it is also motivated to consider parameter combinations (q, ℓ) where q is some parameter that is always upper-bounded by the maximum out-degree of the transition graph.

Finally, it is open to obtain any parameterized hardness results for classic (non-pivoting) formulations of PLS problems. A major obstacle towards such a result is that the existence of any polynomial-time Turing reduction of some NP-hard problem X to a problem L in PLS implies $\text{NP}=\text{coNP}$ [16]. This excludes the standard approach of providing a polynomial parameter transformation from MULTICOLORED INDEPENDENT SET or similar classic problems. In other words, any parameterized reduction from MULTICOLORED INDEPENDENT SET or similar problems would need an exponential running time dependence on the parameter k . An alternative, perhaps more promising, approach would be to develop analogs of XP or W[1] for the PLS class and to identify complete problems for such classes. This, however, would not relate the hardness of such problems to existing complexity classes.

References

- 1 E. H. L. Aarts and J. K. Lenstra. *Local Search in Combinatorial Optimization*. Wiley, Chichester, 1997.
- 2 Maurice Chandoo. Fundamentals of parameterized complexity revisited. *CoRR*, abs/1804.11089, 2018. [arXiv:1804.11089](https://arxiv.org/abs/1804.11089).
- 3 Xi Chen, Chenghao Guo, Emmanouil V. Vlatakis-Gkaragkounis, Mihalis Yannakakis, and Xinzhi Zhang. Smoothed complexity of local Max-Cut and binary Max-CSP. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1052–1065. ACM, 2020. doi:10.1145/3357713.3384325.
- 4 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.
- 5 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.
- 6 Michael Etscheid, Stefan Kratsch, Matthias Mnich, and Heiko Röglin. Polynomial kernels for weighted problems. *J. Comput. Syst. Sci.*, 84:1–10, 2017. doi:10.1016/J.JCSS.2016.06.004.
- 7 John Fearnley, Paul Goldberg, Alexandros Hollender, and Rahul Savani. The complexity of gradient descent: $CLS = PPAD \cap PLS$. *J. ACM*, 70(1):7:1–7:74, 2023. doi:10.1145/3568163.
- 8 John Fearnley and Rahul Savani. The complexity of the simplex method. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 201–208. ACM, 2015. doi:10.1145/2746539.2746558.
- 9 Michael R. Fellows, Fedor V. Fomin, Daniel Lokshtanov, Frances A. Rosamond, Saket Saurabh, and Yngve Villanger. Local search: Is brute-force avoidable? *J. Comput. Syst. Sci.*, 78(3):707–719, 2012. doi:10.1016/J.JCSS.2011.10.003.
- 10 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006. doi:10.1007/3-540-29953-X.
- 11 András Frank and Éva Tardos. An application of simultaneous Diophantine approximation in combinatorial optimization. *Combinatorica*, 7(1):49–65, 1987. doi:10.1007/BF02579200.
- 12 Jaroslav Garvardt, Niels Grüttemeier, Christian Komusiewicz, and Nils Morawietz. Parameterized Local Search for Max c-Cut. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 5586–5594. ijcai.org, 2023. doi:10.24963/IJCAI.2023/620.
- 13 Jaroslav Garvardt, Nils Morawietz, André Nichterlein, and Mathias Weller. Graph Clustering Problems Under the Lens of Parameterized Local Search. In Neeldhara Misra and Magnus Wahlström, editors, *18th International Symposium on Parameterized and Exact Computation, IPEC 2023, September 6-8, 2023, Amsterdam, The Netherlands*, volume 285 of *LIPICs*, pages 20:1–20:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.IPEC.2023.20.
- 14 Jiong Guo, Sepp Hartung, Rolf Niedermeier, and Ondrej Suchý. The parameterized complexity of local search for TSP, more refined. *Algorithmica*, 67(1):89–110, 2013. doi:10.1007/S00453-012-9685-8.
- 15 Sophia Heimann, Hung P. Hoang, and Stefan Hougardy. The k-opt algorithm for the traveling salesman problem has exponential running time for $k \geq 5$. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*, volume 297 of *LIPICs*, pages 84:1–84:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICs.ICALP.2024.84.
- 16 David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis. How easy is local search? *J. Comput. Syst. Sci.*, 37(1):79–100, 1988. doi:10.1016/0022-0000(88)90046-3.

- 17 Christian Komusiewicz and Nils Morawietz. Parameterized Local Search for Vertex Cover: When Only the Search Radius Is Crucial. In Holger Dell and Jesper Nederlof, editors, *17th International Symposium on Parameterized and Exact Computation, IPEC 2022, September 7-9, 2022, Potsdam, Germany*, volume 249 of *LIPICs*, pages 20:1–20:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.IPEC.2022.20.
- 18 Christian Komusiewicz and Nils Morawietz. Finding 3-Swap-Optimal Independent Sets and Dominating Sets is Hard. *ACM Trans. Comput. Theory*, 17(1):1:1–1:31, 2025. doi:10.1145/3700642.
- 19 Mark W. Krentel. Structure in locally optimal solutions (extended abstract). In *30th Annual Symposium on Foundations of Computer Science, FOCS 1989, Research Triangle Park, NC, USA, 30 October - 1 November 1989*, pages 216–221. IEEE Computer Society, 1989. doi:10.1109/SFCS.1989.63481.
- 20 Andrei A. Krokhin and Dániel Marx. On the hardness of losing weight. *ACM Trans. Algorithms*, 8(2):19:1–19:18, 2012. doi:10.1145/2151171.2151182.
- 21 Bodo Manthey, Nils Morawietz, Jesse van Rhiijn, and Frank Sommer. Complexity of local search for Euclidean clustering problems. In Julián Mestre and Anthony Wirth, editors, *35th International Symposium on Algorithms and Computation, ISAAC 2024, December 8-11, 2024, Sydney, Australia*, volume 322 of *LIPICs*, pages 48:1–48:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICs.ISAAC.2024.48.
- 22 Dániel Marx and Ildikó Schlotter. Parameterized complexity and local search approaches for the stable marriage problem with ties. *Algorithmica*, 58(1):170–187, 2010. doi:10.1007/S00453-009-9326-Z.
- 23 Lukas Michel and Alex Scott. Superpolynomial smoothed complexity of 3-flip in local max-cut. *CoRR*, abs/2310.19594, 2023. doi:10.48550/arXiv.2310.19594.
- 24 Wil Michiels, Emile Aarts, and Jan Korst. *Theoretical aspects of local search*. Monographs in Theoretical Computer Science. An EATCS Series. Springer, Berlin, 2007.
- 25 Burkhard Monien, Dominic Dumrauf, and Tobias Tscheuschner. Local search: Simple, successful, but sometimes sluggish. In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part I*, volume 6198 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2010. doi:10.1007/978-3-642-14165-2_1.
- 26 Burkhard Monien and Tobias Tscheuschner. On the power of nodes of degree four in the local max-cut problem. In Tiziana Calamoneri and Josep Díaz, editors, *Algorithms and Complexity, 7th International Conference, CIAC 2010, Rome, Italy, May 26-28, 2010. Proceedings*, volume 6078 of *Lecture Notes in Computer Science*, pages 264–275. Springer, 2010. doi:10.1007/978-3-642-13073-1_24.
- 27 Christos H. Papadimitriou. The complexity of the Lin-Kernighan heuristic for the traveling salesman problem. *SIAM J. Comput.*, 21(3):450–465, 1992. doi:10.1137/0221030.
- 28 Christos H. Papadimitriou, Alejandro A. Schäffer, and Mihalis Yannakakis. On the complexity of local search (extended abstract). In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, STOC 1990, Baltimore, Maryland, USA, May 13-17, 1990*, pages 438–445. ACM, 1990. doi:10.1145/100216.100274.
- 29 Alejandro A. Schäffer and Mihalis Yannakakis. Simple local search problems that are hard to solve. *SIAM Journal on Computing*, 20(1):56–87, 1991. doi:10.1137/0220004.
- 30 Leonard J. Schulman. Clustering for edge-cost minimization (extended abstract). In F. Frances Yao and Eugene M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, STOC 2000, Portland, OR, USA, May 21-23, 2000*, pages 547–555. ACM, 2000. doi:10.1145/335305.335373.
- 31 Stefan Szeider. The parameterized complexity of k-flip local search for SAT and MAX SAT. *Discret. Optim.*, 8(1):139–145, 2011. doi:10.1016/J.DISOPT.2010.07.003.