Dissertation

The Complexity of Local Max-Cut

Tobias Tscheuschner

Paderborn, Juli 2012

Schriftliche Arbeit zur Erlangung des Grades
Doktor der Naturwissenschaften
an der Fakultät für Elektrotechnik, Informatik und Mathematik
der Universität Paderborn

Abstract

Local search is one of the most successful approaches for solving hard optimization problems. In local search, a set of neighbor solutions is assigned to every solution and one asks for a *local optimum*, i.e., a solution that has no better neighbor. The neighborhood relation between the solutions naturally induces *standard algorithms* that find a local optimum: Begin with a feasible solution and iteratively move to a better neighbor until a local optimum is found. Many empirical and theoretical investigations have shown that these methods quickly terminate in a local optimum for most instances.

For some problems, however, instances were found for which a standard algorithm *can* take an exponential number of improving steps if the initial solution and the rule that chooses among the improving neighbors, i.e., the *pivot rule*, are unluckily chosen. Even worse, for some problems, instances and initial solutions were found in which, independent of the pivot rule, *every* standard algorithm takes an exponential number of steps. We say that these problems have the *all-exp* property. Thus, using a standard algorithm turns out to be impractical in some cases.

But how hard is computing a local optimum then—using standard algorithms or any other approach? To encapsulate the complexity of finding local optima, Johnson et al. (JCSS,1988) introduced the complexity class PLS. Shortly afterwards, Schäffer et al. (JOC,1991) showed PLS-completeness for several local search problems including LocalMax-Cut on graphs with unbounded degree with a FLIP-neighborhood in which one node changes the partition. Moreover, they showed two further results for LocalMax-Cut: It has the all-exp property and the StandardAlgorithmProblem (SAP), i.e., the problem of finding a local optimum that is reachable from a given pair of an instance and initial solution via a standard algorithm, is PSPACE-complete. On the positive side, Poljak (JOC,1995) showed that there are at most $O(n^2)$ improving steps possible for LocalMax-Cut on cubic graphs. He also posed the question whether it has the all-exp property on graphs with maximum degree four. Due to the huge gap between the degree three and an unbounded degree, Ackermann et al. (JACM,2008) asked for the smallest $d \in \mathbb{N}$ for which LocalMax-Cut on graphs with maximum degree d is PLS-complete.

This thesis provides three complexity results for LocalMax-Cut. First, it has the all-exp property if restricted to graphs with maximum degree four—this solves the problem stated by Poljak. Second, the SAP is PSPACE-complete for graphs with maximum degree four. Third, finding a local optimum is PLS-complete for graphs with maximum degree five—this solves the problem of Ackermann et al. almost completely since d is narrowed down to four or five (unless PLS $\subseteq$ P). Since LocalMax-Cut has been the basis for several PLS-reductions in the literature, the results have impact on further problems. Some of the reductions directly carry over the degree in some way and transfer the complexity results to the corresponding problems even for very restricted sets of feasible inputs.

Zusammenfassung

Die lokale Suche ist einer der erfolgreichsten Ansätze zur Lösung schwerer Optimierungsprobleme. Bei der lokalen Suche ist jeder Lösung eine Menge von Nachbarlösungen zugeordnet. Gesucht ist ein *lokales Optimum*, das heißt eine Lösung, die keinen besseren Nachbarn hat. Die Nachbarschaftsbeziehung zwischen den Lösungen induziert auf natürliche Weise so genannte *Standardalgorithmen*, die lokale Optima finden: Beginne mit einer zulässigen Lösung und wechsle iterativ zu einem besseren Nachbarn bis ein lokales Optimum gefunden ist. Viele empirische und theoretische Untersuchungen haben gezeigt, dass diese Methoden bei den meisten Eingaben schnell ein lokales Optimum erreichen.

Für einige Probleme sind allerdings Instanzen gefunden worden, bei denen ein Standardalgorithmus exponentiell viele Schritte benötigen kann, wenn die initiale Lösung und die sogenannte *Pivot-Regel*, die unter den verbessernden Lösungen auswählt, unglücklich gewählt sind. Schlimmer noch, für einige Probleme sind Instanzen und initiale Lösungen gefunden worden, in denen unabhängig von der Pivot-Regel *jeder* Standardalgorithmus exponentiell viele Schritte benötigt. Von solchen Problemen sagen wir, dass sie die *All-exp* Eigenschaft haben. Insgesamt gilt also, dass es es in einigen Fällen unpraktisch ist, einen Standardalgorithmus zur Berechnung eine lokalen Optimums zu wählen.

Aber wie schwer ist es dann, ein lokales Optimum zu finden—mit Standardalgorithmen oder einem anderen Ansatz? Um die Komplexität der Berechnung lokaler Optima zu kapseln, haben Johnson et al. (JCSS,1988) die Klasse PLS eingeführt. Kurz danach zeigten Schäffer et al. (JOC,1991) PLS-Vollständigkeit für verschiedene lokale Suchprobleme einschließlich des Problems LocalMax-Cut auf Graphen unbeschränkten Grades mit FLIP-Nachbarschaft, in der ein Knoten die Partition wechselt. Darüber hinaus zeigten sie zwei weitere Ergebnisse für LocalMax-Cut: Es hat die All-exp Eigenschaft und das Problem, ein lokales Optimum zu berechnen, das ausgehend von einem Paar aus Instanz und initialer Lösung mit Hilfe eines Standardalgorithmus erreichbar ist (kurz: SAP), ist PSPACE-vollständig. Auf der anderen Seite zeigte Poljak (JOC,1995), dass höchstens $O(n^2)$ verbessernde Schritte für LocalMax-Cut auf kubischen Graphen möglich sind bis ein lokales Optimum erreicht wird. Außerdem stellte er die Frage, ob LocalMax-Cut auf Graphen mit Höchstgrad vier die All-exp Eigenschaft hat. Wegen der großen Lücke zwischen dem Grad drei und einem unbeschränkten Grad fragten Ackermann et al. (JACM,2008) nach dem kleinsten $d \in \mathbb{N}$, für das LocalMax-Cut auf Graphen mit Höchstgrad d PLS-vollständig ist.

Die vorliegende Arbeit liefert drei Komplexitätsergebnisse für LocalMax-Cut. Erstens behält es die All-exp Eigenschaft auch wenn es auf Graphen mit Höchstgrad vier eingeschränkt wird—dieses Ergebnis löst das Problem von Poljak. Zweitens ist das SAP PSPACE-vollständig auf Graphen mit Höchstgrad vier. Drittens ist die Berechnung eines

lokalen Optimums PLS-vollständig für Graphen mit Höchstgrad fünf—dieses Ergebnis löst das Problem von Ackermann et al. fast vollständig, da d dadurch entweder vier oder fünf ist (außer PLS $\subseteq$ P). Die Ergebnisse haben Einfluss auf weitere Probleme, da LocalMax-Cut in der Literatur als Basis für verschiedene PLS-Reduktionen diente. Einige der Reduktionen behalten den Grad auf bestimmte Weise bei und übertragen so die Komplexitätsergebnisse auf die entsprechenden Probleme auch für sehr eingeschränkte Mengen zulässiger Eingaben.

Acknowledgments

This thesis has been greatly supported by several people to whom I like to express my gratitude here.

Most of all, I thank Burkhard Monien for his support, his advice and his encouragement. He put a lot of trust in and gave a lot of freedom for me and my development while he, at the same time, guided the progress. The value I assign to the overall influence he had on me, my Ph.D. project, and my worldview can hardly be overestimated.

When the time came to write up my thesis, Martina Hüllmann entered "my" office. Her "occupation", however, turned out to be of great advantage for me and my thesis. Not only has she been an invaluable counterpart for conversations that regularly broadened my horizon, she moreover proofread and verified *all* proofs of my thesis.

Prior to that, I fortunately shared my office with Florian Schoppmann. Together with Florian, I went through various ups and downs of the life of a researcher. If Florian had not been by my side in this time, the downs would clearly have been deeper and the highs far less enjoyable.

Throughout my time as a scientific staff member, I have been lucky to collaborate in an outstanding research group. The scientific as well as the personal level among my coworkers contributed a lot to make my stay at the university valuable. In particular, I thank Christian Scheideler for his kind support in several respects towards the end of my Ph.D. project.

I am very grateful to Robert Elsässer, Martin Gairing, Martina Hüllmann, Michelle Kloppenburg, Thomas Sauerwald, Rahul Savani, and Ulf-Peter Schroeder for carefully reading (preliminary) parts of my thesis.

Last but not least, I greatly thank my family and friends for their continuous support and encouragement which significantly improved my overall well-being.

Paderborn, May 2012 *Tobias Tscheuschner*

Contents

1 Introduction **1**
- 1.1 Local search . . . 2
- 1.2 Contribution of This Thesis . . . 6
- 1.3 Further Related Work . . . 8

2 Preliminaries **13**
- 2.1 Basic Notations . . . 13
- 2.2 Local Search . . . 13
- 2.3 Local Max-Cut . . . 15
- 2.4 Boolean Circuits and Boolean Formulas . . . 16

3 Complexity of Local Max-Cut: Maximum Degree Four **19**
- 3.1 Overview of Contribution . . . 19
- 3.2 Basic Properties of Nodes with Maximum Degree Four . . . 20
- 3.3 P-hardness for Graphs with Nodes of Type I and III . . . 22
- 3.4 Is-Exp Property for Graphs with Nodes of Type I and III . . . 25
- 3.5 Enforcing Technique for Graphs with Nodes of Type I, II and III . . . 27
  - 3.5.1 Basic Subgraphs . . . 28
  - 3.5.2 Combining the Subgraphs . . . 32
  - 3.5.3 Enforcing Pivot-Rules with Combined Subgraphs . . . 57
- 3.6 All-Exp Property . . . 82
- 3.7 PSPACE-completeness of the Standard Algorithm Problem . . . 85

4 Complexity of Local Max-Cut: Maximum Degree Five **93**
- 4.1 Overview of Contribution . . . 93
- 4.2 Usage of the P-hardness Reduction . . . 93
- 4.3 Substituting Certain Nodes of Unbounded Degree . . . 94
- 4.4 PLS-completeness . . . 98

5 Impact of the Results on Other Problems **119**
- 5.1 Max-2SAT with FLIP-neighborhood . . . 119
- 5.2 Congestion Games . . . 120
- 5.3 Partitioning with SWAP-neighborhood . . . 120

6 Conclusion and Open Problems **123**

Bibliography **125**

Chapter 1

Introduction

Optimization problems occur in many areas of our daily life. If we are at some location A and want to reach a location B, we usually try to minimize the length of the path connecting A and B. If we decide between different leisure time options, we mostly try to maximize our personal utility. When facing an optimization problem, we are often interested in two measures: the quality of a solution and the time required to find it. If the expected utility of finding a (better) solution is greater than the expected cost for the time needed to find it, then it is rational to search for a (better) solution. In this respect, the required time plays a crucial role in the process of optimization.

For many optimization problems, the computation of an optimum is NP-hard. Since no polynomial-time algorithm is known that computes optimal solutions for such problems, several approaches were developed to find at least *good* solutions. Approximation algorithms, for instance, compute solutions whose quality is not more than a predetermined factor away from an optimum. Unfortunately, some problems even resist polynomial-time approximation in the sense that a polynomial-time algorithm that computes an approximate solution directly leads to a polynomial-time algorithm computing an optimum.

A popular approach to tackle such problems is to use metaheuristics. Nearly all metaheuristics—local search, simulated annealing, evolutionary algorithms, to name a few popular ones—impose a neighborhood relation on the solutions and use it to consecutively improve the set of solutions: They compute from a set S of solutions representing the current state of the computation a new set S' of solutions among the neighbors of the solutions of S, where solutions with higher quality are preferred. In case of the local search approach, the set of solutions representing the current state contains only a single solution and the preference for better solutions is strict, i.e., the computation continues only with a strictly better solution. A local search terminates at a solution that has no better neighbor solution. Such solutions are called *locally optimal*.

For convex optimization problems, the local search metaheuristic is especially appealing since local optima coincide with global optima. On the other hand, the advantage of metaheuristics with non-strict preference for better solutions is that the computation can escape local optima. This is particularly of use if local optima are frequent in the solution space but of rather different quality.

In this thesis, we focus on the local search approach and in particular on the complexity of computing a local optimum. In particular, we consider the so called LocalMax-Cut problem, narrow the border lines of certain complexity properties down for LocalMax-

CUT, and show the impact our results have on related problems.

1.1 Local search

Local search is a frequently used technique of computing solutions for hard optimization problems. Its basic approach is to start with an arbitrary solution and iteratively improve it by local changes defined by a neighborhood relation between the solutions until a local optimum is found. The structure of local search algorithms is outlined in Algorithm 1.1. The approach has been observed to quickly reach local optima for most instances of a wide range of optimization problems and became very popular due to its simplicity and its speed—for comprehensive considerations of local search, we refer to [1, 2, 5, 47].

Input: Instance I of a local search problem Π
Output: Local optimum of I

1: Compute a solution s of I
2: **while** s is not a local optimum of I **do**
3: Compute better solution s' of the neighborhood of s
4: $s \leftarrow s'$
5: **return** s

Algorithm 1.1: Basic structure of local search algorithms

Successful Applications of Local Search Three outstanding examples for successful application of the local search approach are the simplex algorithms for solving linear programs [9, 58], k-opt heuristics for finding solutions of the TRAVELLINGSALESMANPROBLEM [1] and the k-means algorithm for clustering problems [14, 28]. In the following, we take a look at these famous problems and at the complexity results of local search for them—in particular, we focus on results that are closely related to this thesis.

LINEARPROGRAMMING In a linear program, the input is a matrix A and vectors b, c. The task is to find a vector x maximizing $c^T x$ such that $Ax \leq b$. Due to the convexity of linear programs, local optima coincide with global optima which emphasizes the use of local search in a natural way. In 1947, Dantzig [13] introduced the famous simplex method which find an optimum by starting at a vertex of the polytope induced by the constraints $Ax \leq b$ and iteratively moving to better vertices with respect to $c^T x$ along the edges of the polytope until an optimum is reached. Since their invention, simplex methods were successfully applied to linear programs originating from a wide range of applications including scheduling problems, production planning, routing problems and game theory.

In contrast to the short running time of simplex methods observed for practical instances, Klee and Minty constructed linear programs for which the simplex method

with the steepest descent pivoting rule takes an exponential number of steps [38]. For other pivot rules, similar results were shown (some famous examples are in [58, 37]). On the other hand, independently from each other Kalai [31]—who built on a result of Kalai and Kleitman [32]—and Matousek et al. [44] provided randomized pivot rules that lead to a subexponential number of pivot steps for the simplex algorithm. Each result implies that for every initial solution of a linear program there is a sequence of improving steps to an optimum that has subexponential length. However, a polynomial-time computable pivot rule that finds a path of subexponential length to an optimum is not known.

On the positive side, finding an optimum of a linear program is known to be polynomial-time computable since Khachiyan [35] introduced his Ellipsoid method. He used an approach different from local search though. Karmarkar [33] subsequently introduced an interior point method which also takes polynomial time and even outperforms the simplex algorithm in some practical applications.

TravellingSalesmanProblem In the TravellingSalesmanProblem (TSP) the input is an undirected weighted complete graph and the output is a cycle of minimum weight that visits all nodes of the graph. One of the most frequently used local search heuristics for this problem is 2-opt. It starts with an initial tour and iteratively improves it by exchanging two edges of the tour with two different ones as long as such an improving step is possible. For random and "real-world" Euclidean instances, this heuristic is known to compute very good tours within a sub-quadratic number of improving steps [29, 51].

On the other hand, it was shown that there are instances and initial solutions of the TSP for which the k-opt heuristic for $k \geq 2$ can take exponentially many improving steps [11, 43]. However, these instances do not fulfill the triangle inequality and the question whether such instances can be constructed for the metric TSP remained open for a long time. Finally, Englert et al. [19] found Euclidean instances for which the 2-opt heuristic can take exponentially many improving steps.

Clustering The Clustering problem asks for a partition of a set of data points into subsets (the clusters) such that some given measure for the similarity within the clusters is maximized. The problem occurs, depending on the application, in many applications including pattern recognition, data compression and load balancing. A well-studied algorithm for clustering points in the Euclidean space is the k-means algorithm. It starts with an initial set of k centers for the clusters, where each data point is assigned to its closest center. Then it improves the solution by repeatedly performing the following two steps. At first, for each cluster a new center is determined as the average of all points of the cluster, which is called the *mean*, and then each point is assigned to the cluster represented by the closest of the new center points. Note that in each improving step, the sum of the distances of the data points to their corresponding closest center, which can be treated as a potential function, decreases. In the Euclidean space, such an improving step of the k-means algorithm is uniquely determined.

For the k-means algorithm, the number of steps was observed to be linear in the

number of data points on typical instances stemming from practical applications [14]. Contrary to this result and similar to the two above-mentioned famous problems, there are also instances and initial solutions of the clustering problem for which the k-means algorithm takes an exponential number of improving steps to converge [59].

PLS For each of the three famous problems mentioned above it was shown that the local search approach quickly reaches a local optimum for the vast majority of instances—in particular for instances arising from practical applications. However, for each of them one could also find instances and initial solutions for which there is a sequence of improving steps of exponential length. Thus, the local search approach might fail to compute a local optimum in a feasible amount of time. One might think that this problem can be circumvented by putting effort on the pivot rule, i.e., the rule that selects for a given solution the better solution with which the computation is continued (for a formal definition of pivot rules we refer to Definition 2.2.1). The idea is to design a pivot rule that guarantees a subexponential or even polynomial length of every sequence of improving steps. Unfortunately, for none of the three famous problems mentioned above is such a pivot rule known. In case of LINEARPROGRAMMING the question whether a deterministic pivot rule exists that induces a subexponential number of steps for the simplex algorithm is one of the most prominent open questions and is restated in many papers considering pivot rules for simplex (see, e.g., [31] or, recently, [22]). For some local search problems, putting effort on the pivot rule is even hopeless: They contain instances and initial solutions for which even the shortest sequence of improving steps ending up in a local optimum has exponential length (we say that these problems have the *all-exp* property). Altogether, it turns out that using local search for finding a local optimum may not necessarily or even not at all lead to a local optimum in a reasonable number of steps.

Since local search is not necessarily a successful approach to find a local optimum, one might speculate whether one can find a local optimum in some other way or, more generally, what the complexity of computing a local optimum is. For this purpose, Johnson et al. [30] introduced the complexity class PLS (for *polynomial local search*, the class is formally introduced in Definition 2.2.2) which consists of the problems for which local optimality can be verified in polynomial time. In the same paper they introduced the problem CIRCUITFLIP and showed that it is complete for PLS. Subsequently, Schäffer and Yannakakis [54] refined the notion of a PLS-reduction and introduced what is called a *tight* PLS-reduction which, beyond the functionality of ordinary PLS-reductions, additionally preserves two properties. First, the all-exp property is preserved. Second, it preserves the PSPACE-completeness of the STANDARDALGORITHMPROBLEM (SAP), i.e., the problem of computing from a given pair of an instance and initial solution a local optimum that is reachable from the initial solution via improving steps. Then they showed by means of tight PLS-reductions that several famous local search problems have the following three properties. First, they are PLS-complete. Second, they have the all-exp property. Third, their corresponding SAP is PSPACE-complete.

LocalMax-Cut The LocalMax-Cut problem is based on the Max-Cut problem. Max-Cut takes as input an undirected graph $G = (V, E)$ with weighted edges $w : E \to \mathbb{N}$ and asks for a partition of V into two sets V_1 and V_2 that maximizes the sum of the weights of those edges which are incident to one node in V_1 and one in V_2. Max-Cut is one of the most famous combinatorial optimization problems with a wide range of applications including statistical physics and circuit layout design (see [8, 50], e.g.) and is known to be NP-complete—in fact, the decision version of Max-Cut was one of the problems of Karp's list of 21 NP-complete problems [34]. The problem LocalMax-Cut arises from Max-Cut by imposing a neighborhood relation on the set of solutions, namely what is called the FLIP-neighborhood. In this neighborhood, two solutions are neighbors if they can be reached from each other by exchanging exactly one node between the sets V_1 and V_2.

A local optimum of LocalMax-Cut is away from an optimum by a factor of at most two. This is due to the fact that in a local optimum P the sum of the weights of the edges incident to a node $v \in V$ that are in the cut in P is at least the half of the sum of the weights of all edges incident to v. Otherwise the flip of v increases the cut—we say that v is *unhappy* if its flip increases the cut—which is impossible in local optima. Schäffer and Yannakakis [54] showed that LocalMax-Cut is PLS-complete by means of a tight PLS-reduction. The tightness of their reduction additionally implied that LocalMax-Cut has the all-exp property and the corresponding SAP is PSPACE-complete—concurrently to Schäffer and Yannakakis Haken [26] constructed instances for LocalMax-Cut that showed its all-exp property (a description of the instances can be found in [21]). However, the reduction of Schäffer and Yannakakis constructs graphs for LocalMax-Cut with unbounded degree.

For graphs with maximum degree three, Loebl [42] showed that there is a polynomial-time algorithm that computes a local optimum of LocalMax-Cut. His algorithm uses an approach different from local search. The complexity of the local search approach for cubic graphs was considered by Poljak [49]. He showed that starting from an arbitrary solution there are at most $O(n^2)$ improving flips possible for LocalMax-Cut until a local optimum is reached. The property can easily be generalized to arbitrary graphs with maximum degree three. However, a similar result is not possible for graphs with maximum degree four. For a problem closely related to LocalMax-Cut, Haken and Luby [27] showed that there are graphs of maximum degree four and initial solutions for which there is a sequence of improving steps of exponential length. Inspired by their result, Poljak asked whether there are graphs of maximum degree four and initial partitions for which *all* sequences of improving flips have exponential length, i.e., whether LocalMax-Cut has the all-exp property for graphs with maximum degree four. Referring to the PLS-completeness of LocalMax-Cut for graphs with unbounded degree and the polynomial-time computability for cubic graphs, Ackermann et al. [3] asked for the *minimum* degree $d \in \mathbb{N}$ for which LocalMax-Cut is PLS-complete on graphs with maximum degree d.

1.2 Contribution of This Thesis

In this thesis, we consider the complexity of LocalMax-Cut for graphs with maximum degree four and then its complexity for graphs with maximum degree five. In the following, we outline our findings and their implications on other problems. At the beginning of the subsequent chapters, we give more detailed summaries of our results.

Maximum degree four We first introduce three different types of nodes of maximum degree four classified by means of the relation between the weights of their incident edges. The classification allows for a node v and a given partition of the corresponding graph to easily derive the happiness of v from its type, its partition, and the partitions of its adjacent nodes. The types and the characterization of their happiness are frequently utilized in the subsequent parts.

For graphs that contain only two of these types, we show the following two results with a proof that is based on essentially the same construction. First, the problem of computing a local optimum is P-hard with respect to logspace reduction. Second, for each polynomial-time computable function $f : \{0,1\}^n \to \{0,1\}^m$ for $n, m \in \mathbb{N}$ we can compute with logarithmic space a graph G with weighted edges such that in every local optimum the output of f can be read from the partitions of the nodes of G. The second result turns out to be very helpful and finds application in all main results of this thesis.

Building on the second result, we construct an infinite family of pairs of graphs and initial partitions in which there is a sequence of improving flips of exponential length; we say that problems in which this is possible have the *is-exp* property. The graphs in the proof of this result contain the same two types of nodes as in the above-mentioned results. Moreover, the construction of the graphs relies on a Boolean circuit that is mapped to a graph via the reduction function introduced in the aforementioned P-hardness proof. This property and the result itself has significant impact on the subsequent main results, since both properties are used in the proofs of the main results for the maximum degree four.

In preparation of our main results, we develop a technique of extending certain given graphs and initial partitions by further nodes and edges such that in the resulting graph an intended behavior for the sequences of improving flips is enforced. More precisely, the given graphs are obtained from Boolean circuits via the reduction of the P-hardness proof. The intended behavior is specified by means of a polynomial-time computable function h that returns for a given partition either one of the possible improving steps, if there is one, or "nil"—the function h has to return "nil" if its input partition is a local optimum but is allowed to return "nil" if it is not locally optimal. The function naturally *induces* a sequence t of improving steps: Begin at the given pair of graph and initial partition and let h iteratively choose the improving steps until a partition is reached for which h returns "nil". Our technique extends the given graph and initial partition according to h by polynomially many nodes and edges such that *every* sequence of improving steps starting at the resulting pair of graph and initial partition has t as a subsequence. For this reason, we say that our technique *enforces* the behavior induced by h.

Using our enforcing technique, we obtain our first main result:

Theorem 3.6.1. LocalMax-Cut has the all-exp property for graphs with maximum degree four.

In the proof, we use the circuit and the initial partition developed for the proof of the is-exp property and show that there is a polynomial-time computable function h that induces the sequence of exponential length named in that proof. Then our enforcing technique directly implies the all-exp property. Since there are at most $O(n^2)$ improving steps possible for LocalMax-Cut on cubic graphs [49], it follows that the degree four is the minimum degree for which LocalMax-Cut has the all-exp property.

Then we prove our second main result:

Theorem 3.7.1. The StandardAlgorithmProblem for LocalMax-Cut is PSPACE-complete for graphs with maximum degree four.

The proof of this result is done by simulating the computation of a linear bounded automaton by means of improving steps starting at a graph of maximum degree four with an initial partition that corresponds to the initial configuration of the automaton. Then we use our enforcing technique to enforce the intended simulation and use the construction of the is-exp proof to fuel the simulation process as long as necessary.

Maximum degree five Our main result for graphs with maximum degree five is as follows.

Theorem 4.4.2. LocalMax-Cut is PLS-complete for graphs with maximum degree five.

To show this property, we first introduce a technique that substitutes nodes of degree greater than five which have certain properties—we will call these nodes *comparing*—by a subgraph that contains only nodes of maximum degree five. For the graph arising from the substitution of each comparing node v by the corresponding subgraph, we show that in certain local optima all nodes of the subgraph that substitutes v and which are additionally adjacent to a node of the original graph have the same color. Namely, they have the color that v would have in the corresponding partition of the original graph if its flip did not increase the weight of the cut. In this respect, the nodes of the subgraph that substitutes v behave in certain local optima as the original node v. Using this technique, we prove PLS-completeness via a PLS-reduction from the PLS-complete problem CircuitFlip. We map instances of CircuitFlip to graphs with maximum degree five where some of the subgraphs of the graph arise from our substitution technique. Then we show that local optima for these graphs induce local optima in the corresponding instances of CircuitFlip.

Impact on other problems In the literature, several tight PLS-reductions are based on LocalMax-Cut. According to Schäffer and Yannakakis [54] tight PLS-reductions not only lead to PLS-hardness of the corresponding problems but also preserve the following two properties. First, the all-exp property. Second, PSPACE-completeness of the corresponding SAP. Some of the tight PLS-reductions in the literature preserve

the degree of the nodes in some sense. Via these reductions our results directly imply stronger complexity results for the corresponding problems. Namely, we get:

Theorem 5.1.1. For the LocalMax-2SAT(i) problem, in which the instances are restricted such that each variable occurs in at most $i \in \mathbb{N}$ clauses, the following complexity results hold: LocalMax-2SAT(8) has the all-exp property, its corresponding SAP is PSPACE-complete, and LocalMax-2SAT(10) is PLS-complete.

Theorem 5.2.1. For the problem CongNash(i) of computing a Nash equilibrium in congestion games in which every strategy contains at most $i \in \mathbb{N}$ resources, the following complexity results hold: CongNash(4) has the all-exp property, its corresponding SAP is PSPACE-complete, and CongNash(5) is PLS-complete.

Theorem 5.3.1. The problem Partitioning(i) of computing a 2-partition with equally-sized partitions for graphs with maximum degree $i \in \mathbb{N}$ maximizing the sum of the weights of the edges in the cut, has the following properties: Partitioning(5) has the all-exp property, its corresponding SAP is PSPACE-complete and Partitioning(6) is PLS-complete.

Personal contribution and bibliographic notes The constructions of all proofs were, with the following exceptions, entirely developed by myself. The proof of the is-exp property for graphs that contain two of the three types of nodes of maximum degree four (i.e., Theorem 3.4.1) was concurrently and independently developed by Burkhard Monien and myself. Both of our proofs were inspired by the construction of Haken and Luby [27].

The pivot rule in the proof of the all-exp property of LocalMax-Cut for graphs of maximum degree four (i.e., Theorem 3.6.1) was invented by Burkhard Monien and is simpler than the rule previously designed by myself.

Finally, some subgraphs of the PLS-completeness proof of LocalMax-Cut for graphs with maximum degree five were adopted from the construction of Schäffer and Yannakakis [54] and adjusted such that they have maximum degree five. The overall structure of the proof was inspired by the proof of Krentel [39].

A preliminary version of the results for the maximum degree four was published in the *Proceedings of the 7th International Conference on Algorithms and Complexity (CIAC'10) [46]*. The PLS-completeness of LocalMax-Cut for graphs with maximum degree five was published in the *Proceedings of the 38th International Colloquium on Automata, Languages and Programming (ICALP'11) [18]*. Lastly, some parts of this thesis appeared in the survey on local search published in the *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP'10) [47]*.

1.3 Further Related Work

Local Search and PLS By definition of the class PLS, a local optimum of a given PLS-problem is verifiable in polynomial time. Thus, PLS is a subset of FNP, i.e., the

complexity class of search problems whose decision version is NP. It is unlikely that a PLS-problem is NP-hard, since according to Johnson et al. [30] this would imply NP = co-NP. On the other hand, no polynomial-time algorithm is known that solves a PLS-hard problem and therefore it is unclear whether PLS is in FP (for Function NP), i.e., the complexity class of search problems whose decision version is P (since we do not consider decision problems in this thesis, we do not explicitly distinguish between the classes FP and P or NP and FNP, respectively). On the positive side, Orlin et al. [48] showed that one can at least compute an approximate local optimum via a fully polynomial time approximation scheme (FPTAS). For further information on local search, its complexity, and related problems we refer the reader to [2, 56, 61, 62].

MAX-CUT In contrast to LOCALMAX-CUT on cubic graphs, which is in P [42, 49], finding a global optimum of MAX-CUT remains NP-complete for graphs with maximum degree three according to Yannakakis [60]. Even the unweighted case of MAX-CUT, i.e., the case in which all edges have weight 1, was shown to be NP-complete [23]. This result also stands in contrast to the complexity of LOCALMAX-CUT. A sequence of improving steps on graphs with unit weights is upper bounded by $|E|$, since each improving step increases the number of edges that are in the cut at least by one. Another interesting fact is that finding a *minimum* cut is possible in polynomial time by means of computing a maximum flow [17].

A major advance in the approximation of MAX-CUT was accomplished by Goemans and Williamson [25] who used semidefinite programming to compute solutions with an approximation factor of about 0.878. Subsequently, it was shown by Khot et al. [36] that this approximation factor is even best possible under the assumption that the unique games conjecture is true. Moreover, the best possible approximation factor holds for the weighted as well as for the unweighted version of MAX-CUT according to Crescenzi et al. [12].

Smoothed Complexity It was observed that the running time of local search algorithms, in particular, simplex methods for linear programs, is very low on most instances occurring in practical applications. Inspired by this observation, the complexity of the simplex algorithms was investigated for many distributions of random inputs and shown to be in expected polynomial time [4, 10, 55]. The same observation was made for the 2-opt heuristic for computing solutions of the TSP on random instances in the unit hypercube $[0,1]^d$ [11]. However, as for the artificially constructed inputs for which an exponential number of improving steps are possible, it can be argued that the random instances may have certain properties that do not reflect the properties of instances arising in practical applications.

To understand why the running time is polynomial on so many instances stemming from practical applications, Spielman and Teng [57] introduced the notion of smoothed complexity which measures the expected running time of an algorithm under small random perturbations of the input. They showed that the simplex algorithm for linear programs has polynomial smoothed complexity. Subsequently, the notion of smoothed

complexity was adapted for algorithms of various other local search problems. Famous problems with polynomial smoothed complexity are the following: The 2-opt heuristic for Euclidean instances [19], the k-means algorithm [7] and, recently, Elsässer [18] showed that LocalMax-Cut has polynomial smoothed complexity on graphs with logarithmic degree with high probability. The last-mentioned result shows an interesting contrast to the PLS-completeness for graphs with maximum degree five proven in this thesis. Although LocalMax-Cut is hard to solve in general on graphs with a logarithmic degree greater than four, it can be solved in polynomial time for slightly perturbated instances with high probability.

Constraint Satisfaction Problems In the paper of Johnson et al. [30], where the class PLS was introduced, the authors conjectured that a PLS-problem is only PLS-complete if the corresponding problem of verifying a local optimum is P-hard. In contrast to this conjecture, Krentel [40] showed PLS-completeness for a constraint satisfiability problem for which the corresponding verification of a local optimum can be done using logarithmic space. His proof essentially provides the basis of the construction Schäffer and Yannakakis used to prove the PLS-completeness of LocalMax-Cut [54]. The proofs of Schäffer and Yannakakis and Krentel are similar in the sense that the degree of the nodes of the graphs constructed by Schäffer and Yannakakis corresponds to the number of occurrences of the variables in the constraints of Krentel. In both proofs these numbers—i.e., the degree and the number of occurrences, respectively—are unbounded.

However, in a follow-up paper Krentel [39] sketched a proof of PLS-completeness for a constraint satisfiability problem with a constraint length of at most four, at most three occurrences of any variable and trivalent variables. Inspired by the problem considered by Krentel, Dumrauf and Monien [16] (alternatively, see [15]) introduced the MaximumConstraintAssignment (MCA), a generalized version of the problem considered by Krentel. The set of feasible inputs to the problem (p,q,r)-MCA for $p,q,r \in \mathbb{N}$ are functions (i.e., the constraints) that map assignments for the variables to integers. The functions are limited in the sense that each constraint has at most p variables, the maximum occurrence of each variable is q and its valence is r. The neighborhood of an assignment contains all assignments in which the value of a single variable is changed. The value of the solution is the sum over the values of the constraint functions with respect to the given assignment. In these terms, the problem for which Krentel showed PLS-completeness is $(4,3,3)$-MCA. In their paper, Dumrauf and Monien show PLS-completeness for $(3,2,3)$-MCA, $(2,3,6)$-MCA and $(6,2,2)$-MCA. Let us remark that the LocalMax-Cut for graphs with maximum degree k, which is in the focus of this thesis for $k=4$ and $k=5$, can be formulated as a $(2,k,2)$-MCA problem with a restricted set of feasible constraint functions.

Congestion Games Congestion games were introduced by Rosenthal [52] as a model for the behavior of selfish players that share resources whose cost depend on the number of players that use the corresponding resource. In his paper, he showed via a potential function argument that every congestion game has a (pure) Nash equilibrium, i.e., a

state in which neither player can improve its utility by unilaterally changing its strategy. Subsequently, Monderer and Shapley [45] strengthened the relation of congestion games to potential functions and showed that congestion games are isomorphic to potential games, i.e., games in which the players aim to improve a given potential function.

The close relation between congestion games and the class PLS was shown by Fabrikant et al. [20]. They proved PLS-completeness for the following three problems. First, computing a Nash equilibrium in congestion games. Second, computing a Nash equilibrium in symmetric congestion games, i.e., congestion games in which the strategies of all players are the same. Third, computing a Nash equilibrium in network congestion games, i.e., games in which the strategies of the players correspond to paths in an underlying network. On the positive side, they showed that a Nash equilibrium for symmetric network congestion games is polynomial-time computable via min-cost flow algorithms. This is in particular of interest, since Ackermann et al. [3] subsequently showed that symmetric network congestion games have the all-exp property. In their paper, Ackermann et al. also proved that the number of improving steps in congestion games is polynomial if the combinatorial structure of the strategies of the players are based on matroids. Moreover, they simplified the proof of the PLS-completeness of computing a Nash equilibrium for network congestion games in comparison to the earlier proof of Fabrikant et al. [20].

Chapter 2

Preliminaries

2.1 Basic Notations

Sets The set of natural numbers without zero, i.e., $\{1, 2, 3, \ldots\}$, is denoted by $\mathbb{N}$, the set of natural numbers including zero is denoted by $\mathbb{N}_{\mathbf{0}}$, the set of rational numbers is denoted by $\mathbb{Q}$ and the set of non-negative rational numbers is denoted by $\mathbb{Q}_{\mathbf{>0}}$. For the set of functions that grow polynomially in a variable $n \in \mathbb{N}$ we write $\boldsymbol{O(poly(n))}$, i.e., $O(\text{poly}(n)) := \bigcup_{k \in \mathbb{N}} O(n^k)$ for $n \in \mathbb{N}$.

2.2 Local Search

Definition 2.2.1. *A **local search problem** Π consists of a set of instances $\mathscr{I}$, a set of feasible solutions $\mathscr{F}(I)$ and an objective function $f_I : \mathscr{F}(I) \to \mathbb{Q}$ for every instance $I \in \mathscr{I}$. In addition, for every solution $s \in \mathscr{F}(I)$ there is a **neighborhood** $\mathscr{N}(s, I) \subseteq \mathscr{F}(I)$. For an instance $I \in \mathscr{I}$, the problem is to find a **local optimum**, i.e., a solution $s \in \mathscr{F}(I)$ such that for all $s' \in \mathscr{N}(s, I)$ we have $f_I(s) \geq f_I(s')$ in case of maximization and $f_I(s) \leq f_I(s')$ in case of minimization. A **standard algorithm** [30] is an algorithm that computes a local optimum by first computing a feasible solution and then iteratively moving to a better neighbor until a local optimum is reached. A **pivot rule** is a function that returns for a given pair (I, s) of instance $I \in \mathscr{I}$ and solution $s \in \mathscr{F}(I)$ a solution in $\mathscr{N}(s, I)$ with a better objective function value than s if there is one, and "**nil**" otherwise.*

PLS

Definition 2.2.2 ([30]). *A local search problem Π is in the class **PLS** if the following three polynomial-time algorithms exist: algorithm A computes for every instance $I \in \mathscr{I}$ a feasible solution $s \in \mathscr{F}(I)$, algorithm B computes for every $I \in \mathscr{I}$ and $s \in \mathscr{F}(I)$ the value $f(s)$, and algorithm C is a pivot rule.*

Definition 2.2.3 ([30]). *A problem $\Pi \in$ PLS is* PLS-***reducible*** *to a problem $\Pi' \in$ PLS if the following polynomial-time computable functions Φ and Ψ exist. The function Φ maps instances I of Π to instances of Π' and Ψ maps pairs (s, I), where s is a solution of $\Phi(I)$, to solutions of I such that for all instances I of Π and local optima s^* of $\Phi(I)$ the solution $\Psi(s^*, I)$ is a local optimum of I. Finally, a local search problem Π is* PLS-***complete*** *if $\Pi \in$ PLS and every problem in* PLS *is* PLS*-reducible to Π.*

Improving steps

Definition 2.2.4. *Let Π be a problem in* PLS*, $\mathscr{I}$ be the set of its instances, $\mathscr{F}(I)$ be the set of feasible solutions and $f_I : \mathscr{F}(I) \to \mathbb{Q}$ be the objective function for $I \in \mathscr{F}(I)$. Let $I \in \mathscr{I}$ and $s_1, \ldots, s_n \in \mathscr{F}(I)$ for $n \in \mathbb{N}$ such that $s_{i+1} \in \mathscr{N}(s_i, I)$ for all $1 \leq i < n$. Then the sequence $s := (s_1, \ldots, s_n)$ is called a **sequence of steps**. If $n = 2$ then s is also called a **step**. Moreover, s is called **improving** if $f_I(s_{i+1}) > f_I(s_i)$ for all $1 \leq i < n$ in case of maximization and $f_I(s') < f_I(s)$ for all $1 \leq i < n$ in case of minimization. We say that Π has the **is-exp** property if there is an infinite family of pairs (I, s) with $I \in \mathscr{I}$ and $s \in \mathscr{F}(I)$ for which there is a sequence of improving steps of exponential length[1] in I starting from s. Furthermore, we say that Π has the **all-exp** property if there is an infinite family of pairs (I, s) with $I \in \mathscr{I}$ and $s \in \mathscr{F}(I)$ such that every sequence of improving steps in I starting from s has exponential length.*

Definition 2.2.5 ([30]). *Let Π be a problem in* PLS*, $\mathscr{I}$ be the set of its instances, and $\mathscr{F}(I)$ be the set of feasible solutions. For an instance $I \in \mathscr{I}$ and a solution $s \in \mathscr{F}(I)$ the **StandardAlgorithmProblem** asks for a solution $s' \in \mathscr{F}(I)$ for which s' is reachable from s via a sequence of improving steps.*

Definition 2.2.6 ([54]). *Let Π be a problem in* PLS *and I be an instance of Π. The **neighborhood graph $NG(I)$** of the instance I is a directed graph with one vertex for each feasible solution of I and an arc $s \to t$ for feasible solutions s, t of I if $t \in \mathscr{N}(s, I)$. The **transition graph $TG(I)$** is the subgraph of $NG(I)$ that contains the arcs $s \to t$ for which t has a strictly better objective value than s (i.e., greater if Π is a maximization problem and smaller if it is a minimization problem).*

Definition 2.2.7 ([54]). *Let $\Pi, \Pi' \in$ PLS *and (Φ, Ψ) be a* PLS*-reduction from Π to Π'. The reduction is called **tight** if for any instance I of Π there is a subset $\mathscr{R}$ of the set of feasible solutions of the image instance $J := \Phi(I)$ of Π' so that the following properties are satisfied:*

- *$\mathscr{R}$ contains all local optima of J.*
- *For every feasible solution p of I, we can construct in polynomial time a solution $q \in \mathscr{R}$ of J such that $\Psi(q, I) = p$.*
- *Suppose that the transition graph of J, $TG(J)$, contains a directed path $q \to\!\!\to q'$ such that $q, q' \in \mathscr{R}$ but all internal path vertices are outside $\mathscr{R}$ and let $p := \Psi(q, I)$ and $p' := \Psi(q', I)$ be the corresponding feasible solutions of I. Then either $p = p'$ or $TG(I)$ contains an arc from p to p'.*

[1] A sequence is said to have exponential length with respect to the size n of the input I if it contains at least c^{n^d} steps for some constants $c > 1$ and $d > 0$.

2.3 Local Max-Cut

Definition 2.3.1. *The problem **LocalMax-Cut** is a local search problem. An instance of* LocalMax-Cut *is an undirected graph $G = (V, E)$ with positive edge weights $w : E \to \mathbb{Q}_{>0}$. A feasible solution is a partition of V into two sets V_1, V_2. The objective is to maximize the sum of the weights of the edges $\{u, v\}$ with $u, v \in V$ for which $u \in V_1$ and $v \in V_2$. The neighborhood of a solution s contains each solution arising from s by moving a single node from one of the sets V_1 and V_2 to the other.*

Observation 1. *For* LocalMax-Cut*, a pivot rule is a function that maps a partition of a given graph to an unhappy node and returns "nil" if the partition is a local optimum.*

Definition 2.3.2. *A **generalized pivot rule** for* LocalMax-Cut *is a function that either maps a partition of a given graph to an unhappy node or returns "nil".*

The difference between a generalized pivot rule and a pivot rule for LocalMax-Cut is that a generalized pivot rule may return "*nil*" in partitions that are not a local optimum. Note that each pivot rule for LocalMax-Cut is also a generalized pivot rule.

Prerequisite: Weighted Graphs In this thesis, we consider the LocalMax-Cut problem only *with* weighted edges. Thus, whenever we introduce a graph $G = (V, E)$, we omit the attribute "weighted" and assume $w : E \to \mathbb{Q}_{>0}$ to be the function for the edge weights of the graph.

Degree For a graph $G = (V, E)$ and a node $v \in V$ we let $\boldsymbol{deg_G(v)}$ be the degree of v in G, i.e., the number of edges incident to v in G. Moreover, we let $\boldsymbol{deg(G)}$ be the degree of G, i.e., $deg(G) := max_{v \in V} deg_G(v)$.

Partitions Let $G = (V, E)$ be a graph. The graph G together with a 2-partition P of V into two sets $V_1, V_2 \subseteq V$ is called a **partitioned graph** and denoted by $\boldsymbol{(G, P)}$. Since all partitions in this thesis are 2-partitions, we simply say partition instead of 2-partition. The set of all partitions of V is denoted by $\boldsymbol{\mathcal{P}(V)}$. Let $P \in \mathcal{P}(V)$. We let $\boldsymbol{c_{(G,P)}} : V \to \{0, 1\}$ with $c_{(G,P)}(u) = 1$ for $u \in V$ if and only if $u \in V_1$ with respect to P and call $c_{(G,P)}(u)$ the **color** of u. In particular, we say that u is **black** if $c_{(G,P)}(u) = 1$ and that it is **white** if $c_{(G,P)}(u) = 0$. If the considered graph is clear from the context then we simply write $c_P(u)$, and if the partition is also clear then we even just write $c(u)$. We say that an edge $\{u, v\} \in E$ is **in the cut** in P if $c_P(u) \neq c_P(v)$. For a vector $v := (v_1, \ldots, v_n)^T$ of nodes $v_i \in V$ for $1 \leq i \leq n$ and $n \in \mathbb{N}$ we let both $c(v)$ and $c(v_1, \ldots, v_n)$ refer to the vector $(c(v_1), \ldots, c(v_n))^T$. Since all vectors in this thesis are transposed, we from now on omit the "T" in the exponent. For a subset $V' \subseteq V$ we let $\boldsymbol{P|_{V'}}$ be the partition of V' such that $c_P(v) = c_{P|_{V'}}(v)$ for all $v \in V'$.

Flips Let $G = (V, E)$ be a graph. For a partition $P_0 \in \mathscr{P}(V)$ and a sequence $s := (u_1, \ldots, u_q)$ of nodes for $q \in \mathbb{N}$ and $u_i \in V$ for all $1 \leq i \leq q$ we call s a **sequence of flips starting at** (G, P_0). If the graph is clear from the context then we also say that s is a sequence of flips in P_0, and if the partition is also clear then we even just say that s is a sequence of flips. If $q = 1$ then s is called a **flip of** u_1. We denote by $\boldsymbol{P_{s,i}}$ for $1 \leq i \leq q$ the partition arising from $P_{s,i-1}$ by a flip of u_i where $P_{s,0} := P_0$. If the considered sequence is clear from the context then we simply write P_i. The sequence s of flips is called **improving** if the sequence $(P_0, \ldots, P_q)$ of steps is improving. Throughout the thesis we only consider sequences of flips that are improving and therefore we may omit the attribute "improving". The sequence s of flips is called **final** if P_q is a local optimum. A node u is **happy** in (G, P_0) (or *happy in* P_0 if the considered graph is clear from the context or just *happy* if even the partition is clear) if the flip of u is not improving in P_0 and **unhappy in** P_0 otherwise—note that a partition $P \in \mathscr{P}(V)$ is a local optimum if and only if v is happy in P for all $v \in V$. For $1 \leq i \leq j \leq q$ we let $\boldsymbol{s_i^j} := (u_i, \ldots, u_j)$—we let s_i^j for $j < i$ be the empty sequence. For two sequences $s = (v_1, \ldots, v_q)$ and $t = (w_1, \ldots, w_r)$ of flips for $q, r \in \mathbb{N}$, $v_i \in V$ for all $1 \leq i \leq q$ and $w_i \in V$ for all $1 \leq i \leq r$ the composition $(v_1, \ldots, v_q, w_1, \ldots, w_r)$ of s and t is denoted by $\boldsymbol{s \circ t}$. For a partition $P_0 \in \mathscr{P}(V)$ and a generalized pivot rule h starting at (G, P_0) we call the sequence $(w_1, \ldots, w_q)$ starting at (G, P_0) for which $h(P_i) = w_{i+1}$ for all $0 \leq i < q$ and $h(P_q) = nil$ for all $0 \leq i < q$ **induced by** h. For $A, B \subseteq V$ with $A \cap B = \emptyset$ and $P_0 \in \mathscr{P}(V)$ we write $\boldsymbol{A <_{P_0} B}$ if for every sequence $s = (w_1, \ldots, w_q)$ starting at (G, P_0) with $q \in \mathbb{N}$ and every $1 \leq i \leq q$ for which $w_i \in B$ there is a $1 \leq j < i$ such that $w_j \in A$. If the partition is clear from the context then we just write $A < B$. For a sequence s of flips and a subset $V' \subseteq V$ we let $\boldsymbol{s|_{V'}}$ be the sequence arising from s by deleting the flips the nodes of $V \setminus V'$.

2.4 Boolean Circuits and Boolean Formulas

Boolean Circuits In the literature, Boolean circuits are defined in various, conceptually equivalent ways. In this thesis, we use a definition that is inspired by the definition of Arora and Barak [6].

Definition 2.4.1. *A* ***Boolean circuit*** *C is a directed acyclic graph (V, E) with a maximum indegree of two for all nodes and a logical operation—i.e., AND, OR, NOT, NAND, NOR, XOR or XNOR—assigned to each node of V whose indegree is not zero. The nodes with indegree zero are called* ***input nodes*** *of C, the nodes with outdegree zero are called* ***output nodes*** *of C and all noninput nodes are called* ***gates****. If a logical operation $*$ is assigned to a gate $g \in V$ then we call g a* ***$*$-gate****. The indegree of a node v is called the* ***fan-in*** *of v and its outdegree is called its* ***fan-out****.*

Definition 2.4.2. *Let C be a Boolean circuit with $n \in \mathbb{N}$ inputs and $m \in \mathbb{N}$ outputs. An* ***input*** *of C is a vector $x \in \{0,1\}^n$ and the* ***output*** *of C on input x, denoted by $\boldsymbol{C(x)}$, is derived by assigning a value $val(v)$ to each node v of C in the following way: If v is an input node then we let $val(v) = x_i$, otherwise $val(v)$ is the output of the logical operation*

(a) NOT-gate **(b)** NOR-gate **(c)** Input marker **(d)** Output marker

Figure 2.1: A NOT- and a NOR-gate as well as the markers incident to input nodes and output nodes.

assigned to v with respect to the values of the nodes adjacent to v via an ingoing edge of v. Then the output $C(x)$ is defined as the vector of the values of the output nodes of C.

A property that we frequently use is the following:

Proposition 2.4.3 (see, e.g., [53]). *Let C be a Boolean circuit with $N \in \mathbb{N}$ nodes and $n \in \mathbb{N}$ input nodes. Then there is a Boolean circuit C' with $O(N)$ nodes and n input nodes that contains only NOR-gates with a fan-in of two such that for all $x \in \{0,1\}^n$ we have $C(x) = C'(x)$.*

Throughout the thesis we introduce several Boolean circuits via drawings. In all Boolean circuits, we only use NOT-gates or NOR-gates. The gates are drawn according to the ANSI-standard. A NOT-gate is depicted in Figure 2.1a and a NOR-gate in Figure 2.1b. We do not draw the input nodes. Instead, the input of the nodes adjacent to the input nodes are labelled by the marker in Figure 2.1c. On the other hand, the output gates are drawn. Here, we label the output of the output gates by the marker depicted in Figure 2.1d.

Boolean Formulas For a set W of Boolean variables we let $\boldsymbol{\Phi(W)}$ be the set of all Boolean formulas in disjunctive normal form over variables of W. The empty Boolean formula is denoted by the empty set, i.e., $\emptyset$. Let $\phi \in \Phi(W)$ with $\phi = \bigvee_{i=1}^{n} M_i$ for $n \in \mathbb{N}$ and $M_i = \bigwedge_{j=1}^{m_i} l_{i,j}$ where m_i is the number of literals of monomial M_i and $l_{i,j}$ for any $1 \leq i \leq n$, $1 \leq j \leq m_i$ is a literal over a Boolean variable of W. Let $W^{\phi} \subseteq W$ be the set of variables of ϕ. For an assignment $t : W^{\phi} \to \{0,1\}$ we let $\boldsymbol{val_t(\phi)}$ be the truth value of ϕ if each variable x of ϕ has the value $t(x)$. We let $\boldsymbol{Mons(\phi)}$ be the set of monomials of ϕ, $\boldsymbol{Lits(M)}$ be the set of literals of a monomial M and $\boldsymbol{pos(l)}$ be the function that returns 0 if literal l negates its corresponding variable and 1 otherwise.

Chapter 3

Complexity of Local Max-Cut: Maximum Degree Four

3.1 Overview of Contribution

In this chapter, we devise several complexity results for LocalMax-Cut on graphs with maximum degree four. For this, we first introduce three different types of nodes. The types classify nodes of maximum degree four based on the relation between the weights of their incident edges. The classification allows a simple characterization of the happiness of a node in a given partition. The characterization in turn is frequently exploited in the subsequent parts.

Then we show two results with basically the same construction for graphs that contain only two of the introduced types. First, the problem of computing a local optimum is P-hard with respect to logspace reduction. Second, for each polynomial-time computable function $f : \{0,1\}^n \to \{0,1\}^m$ for $n, m \in \mathbb{N}$ one can compute with logarithmic space a graph G such that in every local optimum the output of f can be read from the colors of the nodes of G. The second result turns out to be very useful. In fact, it is applied in the proofs of all main results of this thesis.

As a first application of the result, we construct an infinite family of pairs of graphs and initial partitions for which there is a sequence of improving flips of exponential length. The graphs in the proof of this result contain the same two types of nodes as in the construction for the P-hardness result which implies the is-exp property of LocalMax-Cut on such graphs. Actually, the construction relies on a Boolean circuit that is mapped to a graph via the reduction function of the P-hardness proof.

Then we devise a technique of enforcing any polynomial-time computable pivot rule on certain graphs. More concretely, the technique takes as input a Boolean circuit C, a partition P of the nodes of the graph G^C obtained from C via the reduction function in the P-hardness proof and a polynomial-time computable generalized pivot rule for G. The generalized pivot rule naturally induces a sequence t of improving flips starting at (G, P^C): Begin with the initial partition P, let the pivot rule choose an improving flip, perform the flip, get thereby another partition and repeat this procedure until the generalized pivot rule outputs "nil". The technique computes in polynomial time a graph $G' = (V', E')$ with $V \subseteq V'$ and an initial partition $P' \in \mathscr{P}(V')$ such that *every* final sequence of flips starting at (G', P') has t as a subsequence. In other words, for any polynomial-time computable generalized pivot rule h for the graph G^C, the technique

extends G by polynomially many nodes and edges such that—independently of the pivot rule that is performed in the extended graph—*every* final sequence of improving flips in the extended graph has the sequence t as a subsequence. For this reason, we say that our technique *enforces* the generalized pivot rule h.

Using our technique, we show the all-exp property for graphs with maximum degree four. For this, we use the circuit and the initial partition developed for the proof of the is-exp property and show that there is a polynomial-time computable pivot rule that induces the sequence of exponential length of the proof of the is-exp property. Then the enforcing technique directly implies the all-exp property.

Finally, we show the PSPACE-completeness of the StandardAlgorithmProblem. We do this by simulating the computation of a linear bounded automaton within a graph of maximum degree four by using the enforcing technique and we use the construction of the is-exp proof to fuel the simulation process as long as necessary.

Prerequisite: Maximum Degree Four Since we only consider graphs with maximum degree four in this chapter, we assume an implicit statement that the graph has maximum degree four each time we introduce a graph.

3.2 Basic Properties of Nodes with Maximum Degree Four

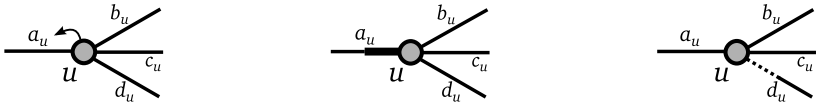
Definition 3.2.1. *Let $G = (V, E)$ be a graph. For a node $u \in V$ and edges a_u, b_u, c_u, d_u incident to u with $w(a_u) \geq w(b_u) \geq w(c_u) \geq w(d_u)$ we distinguish the following types for u:*

$$\textit{Type of } u := \begin{cases} \textit{I}, & \textit{if } w(a_u) > w(b_u) + w(c_u) + w(d_u) \\ \textit{II}, & \textit{if } w(a_u) + w(d_u) > w(b_u) + w(c_u) \textit{ and } \\ & w(a_u) < w(b_u) + w(c_u) + w(d_u) \\ \textit{III}, & \textit{if } w(a_u) + w(d_u) < w(b_u) + w(c_u). \end{cases}$$

These three types do not cover all possible nodes for graphs of maximum degree four—which is due to the fact that the inequalities are strict—but if a node has one of these types then we can characterize its happiness in local optima:

Observation 2. *For a graph $G = (V, E)$, $P \in \mathscr{P}(V)$, $u \in V$ and edges a_u, b_u, c_u, d_u incident to u with $w(a_u) \geq w(b_u) \geq w(c_u) \geq w(d_u)$ the following three conditions are satisfied:*

- *If u is of Type I then u is happy in P if and only if a_u is in the cut.*
- *If u is of Type II then u is happy in P if and only if a_u and at least one other edge is in the cut or b_u, c_u and d_u are in the cut.*
- *If u is of Type III then u is happy in P if and only if at least two of the edges a_u, b_u, c_u are in the cut.*



(a) Type I **(b)** Type II **(c)** Type III

Figure 3.1: Illustration of the three types for node u.

Throughout this thesis we introduce several graphs containing nodes of these three types. To simplify the reading process we introduce the graphs by means of drawings. In Figure 3.1 we show how we distinguish the different types of nodes in our illustrations. A node u of Type I has a little arrow pointing to the heaviest edge incident to u (see Figure 3.1a). If u is of Type II then it has an incident edge which has a thick half (see Figure 3.1b). The half-thick edge is the heaviest edge a_u incident to u and the thick half of a_u is adjacent to u. If u is of Type III then the lightest edge incident to u is half-dotted (see Figure 3.1c) where the dotted half of the edge is adjacent to u.

Besides introducing graphs, the drawings throughout this chapter sometimes simultaneously introduce partitions of the nodes. In that case, we give a node a black filling if its color is black in the corresponding partition and we give it a white filling if its color is white.

Definition 3.2.2. *For a graph $G = (V, E)$ we let $\boldsymbol{V_I}, \boldsymbol{V_{II}}$ and $\boldsymbol{V_{III}}$ be the sets of nodes of Type I, II and III, respectively. For two adjacent nodes $u, v \in V$ we say that u has **influence** on v if one of the following conditions is satisfied:*

- *v is of Type I and $\{u, v\}$ is the heaviest edge incident to v.*
- *v is of Type II.*
- *v is of Type III and $\{u, v\}$ is not the lightest edge incident to v.*

For an edge $e := \{u, v\}$ we say that e has influence on v if u has influence on v.

Note that the happiness of a node u in a partitioned graph G_P is independent of the color of a neighbor that has no influence on u.

Definition 3.2.3. *Let $G = (V, E)$ be a graph. For a node $v \in V_{III}$ to which an edge e is incident we call e the **third edge of** v if there are exactly two edges with strictly greater weight than e incident to v. We let $\boldsymbol{V^3_{III}}$ be the set of nodes $v \in V_{III}$ to which an edge e is incident that is the third edge of v. We let $\boldsymbol{T_G} : V^3_{III} \to V$ be the function that returns for a given node $v \in V^3_{III}$ the node adjacent to v via the third edge of v. We let $\boldsymbol{H_G} : V_I \to V$ be the function that returns for a given node $v \in V_I$ the node adjacent to v via the heaviest edge incident to v. The heaviest edge incident to $v \in V_I$ is called the **heaviest edge of** v. Finally, we let $\boldsymbol{R_G} : V_I \cup V^3_{III} \to V$ be the function that returns for a given node $v \in V_I \cup V^3_{III}$ the node $H_G(v)$ if $v \in V_I$ and $T_G(v)$ otherwise. If the considered graph is clear from the context then we omit the subscript indicating the graph.*

Comment If a node $v \in V_{III}$ has a third edge e then e is the unique third edge of v since in the case that there is a further edge incident to v with the same weight as e, node v is not of Type III at all.

3.3 P-hardness for Graphs with Nodes of Type I and III

The theorem in this section is mainly based on the following property of a node u of Type III in local optima. Assume that one of the neighbors with influence on u is black in a local optimum. Then u is black if and only if the other two neighbors with influence on u are white. This property can be used to simulate a NOR-gate of a Boolean circuit since the output of a NOR-gate is true if and only if both inputs are false. The propagation of the outputs of a gate to the inputs of other gates is done via nodes of Type I which resemble NOT-gates since they have the opposite color of the node that has influence on them in any local optimum.

Theorem 3.3.1 (Constituting Theorem). *i)* LocalMax-Cut *is P-hard with respect to logspace reduction for graphs that contain only nodes of Type I and III.*

ii) Let $f : \{0,1\}^n \to \{0,1\}^m$ be a function and C be a Boolean circuit with $N \in \mathbb{N}$ gates computing f. Then, using $O(logN)$ space, one can compute a graph $G^C = (V^C, E^C)$ that contains only nodes of Type I and III and nodes $s_1, \ldots, s_n, t_1, \ldots, t_m \in V^C$ of degree one such that for the vectors $s := (s_1, \ldots, s_n), t := (t_1, \ldots, t_m)$ we have $f(c_P(s)) = c_P(t)$ in every local optimum P of G^C.

Proof. i) We reduce from the P-complete problem Circuit-Value [41]. An instance of Circuit-Value is a Boolean circuit C consisting of $N \in \mathbb{N}$ gates $g_N, \ldots, g_1$ and an assignment for the inputs of C. The solution is the output of C for the given input assignment. Without loss of generality we make the following four assumptions. First, a gate of C is either a NOR-gate with a fan-in of two and a fan-out of one or a NOT-gate with a fan-in of one and a fan-out of at most two—a circuit that only contains NOR-gates can be constructed according to Proposition 2.4.3 and the main purpose of the NOT-gates is to distribute the output of the NOR-gates. Second, the gates are ordered topologically such that if g_i is an input of g_j then $i > j$. Third, $g_m, \ldots, g_1$ are NOT-gates with a fan-out of one and the vector of their outputs is the output of C. Fourth, $g_N, \ldots, g_{N-n+1}$ are also NOT-gates and the vector of their inputs is the input of C. We let $I_1(g_i)$ and $I_2(g_i)$ be the input gates of a NOR-gate g_i for $1 \leq i \leq N-n$, $I(g_i)$ be the input gate of a NOT-gate g_i for $1 \leq i \leq N-n$, and $value(g_i)$ be the value of the assignment of the input corresponding to g_i for $N-n+1 \leq i \leq N$.

We construct a graph $G = (V, E)$ with weights $w : E \to \mathbb{N}$ from C as follows. The set of nodes is $V = \{v_1, \ldots, v_{3N+1}\}$. The set E contains for every $1 \leq i \leq 3N$ the following edges of weight 2^i:

a) If g_i is a NOR-gate then $\{v_i, v_{N+2i}\} \in E$ and in addition $\{v_i, v_j\} \in E$ for each $m < j \leq N$ for which there is a $1 \leq k \leq 2$ with $I_k(g_i) = g_j$.

b) If g_i is a NOT-gate for $1 \leq i \leq N-n$ then $\{v_i, v_j\} \in E$ for the unique $1 \leq j \leq N$ for which $I(g_i) = g_j$.

c) For all $N-n+1 \leq i \leq N$: if $value(g_i) = 1$ then $\{v_i, v_{N+2i}\} \in E$ and $\{v_i, v_{N+2i-1}\} \in E$ otherwise.

d) For all $N+1 \leq i \leq 3N$: $\{v_i, v_{i+1}\} \in E$.

Then the type and degree of any $v \in V$ as well as the edges with influence on v can be seen in Table 3.1. The nodes v_i for $N+1 \leq i \leq N+2n$ are not necessary for the proof and are only introduced to simplify the description.

Example 3.3.2. *For the sake of illustration, let the circuit in Figure 3.2 be an instance for C and let* $value(g_4) = 1$ *and* $value(g_3) = 0$*. Then the graph G constructed from the instance of C is as presented in Figure 3.3.*

Range of i	Properties of v_i		
	Type	Degree	Influenced by edge(s)
$i = 3N+1$	I	1	$\{v_{3N+1}, v_{3N}\}$
$N+1 \leq i \leq 3N$	I	≤ 3	$\{v_i, v_{i+1}\}$
$N-n+1 \leq i \leq N$	I	2	edge introduced in (c)
$1 \leq i \leq N-n$	I	≤ 3	edge introduced in (b)
$m+1 \leq i \leq N-n$	III	4	all three edges introduced in (a)

Table 3.1: Degrees, types and influences for all nodes in V.

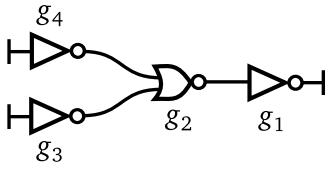


Figure 3.2: An instance for the Boolean circuit C.

Let $P \in \mathscr{P}(V)$ be a local optimum for G. Due to the symmetry of LocalMax-Cut we may assume without loss of generality that $c_P(v_{3N}) = 1$. In the following, we use Observation 2 to deduce the colors of the remaining nodes. First, $c_P(v_i) \neq c_P(v_{i+1})$ for all $N+1 \leq i \leq 3N$. Consequently, $c_P(v_{N+2i}) = 1$ and $c_P(v_{N+2i-1}) = 0$ for all $1 \leq i \leq N$ and $c_P(v_{3N+1}) = 0$. Then, for each $N-n+1 \leq i \leq N$, we have $c_P(v_i) = 0$ if $value(g_i) = 1$ and $c_P(v_i) = 1$ otherwise. Thus, the color of the node v_i for any $N-n+1 \leq i \leq N$ corresponds to the complement of the input assignment for g_i. Now consider the nodes v_i for $1 \leq i \leq N-n$. If g_i is a NOT-gate with $I(g_i) = g_j$ for $m+1 \leq j \leq N$ then $c_P(v_i) \neq c_P(v_j)$. Hence, the color of v_i for any $1 \leq i \leq N-n$

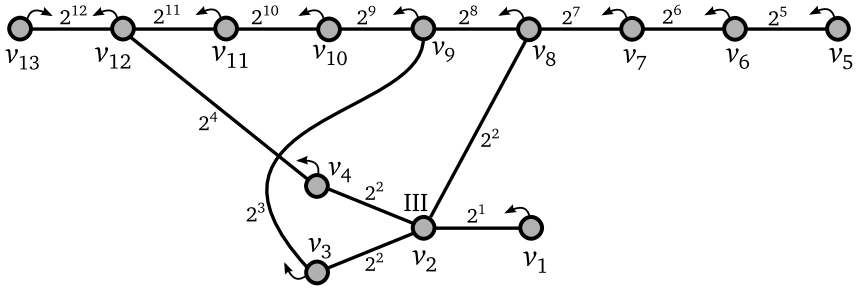


Figure 3.3: The graph G constructed from the instance for C.

corresponds to the output of a NOT-gate with respect to the color of v_j. Finally, if g_i is a NOR-gate with $I_1(g_i) = g_k$ and $I_2(g_i) = g_j$ for $m+1 \leq j < k \leq N$ then $c_P(v_i) = 1$ if and only if $c_P(v_j) = c_P(v_k) = 0$ since v_i is of Type III and its neighbor v_{N+2i} is black in P. Thus, the color of v_i corresponds to the output of a NOR-gate with respect of the colors of v_j and v_k. Altogether, the colors of each node v_i for $1 \leq i \leq N-n$ corresponds to the output of g_i in C for the given input assignment and therefore the colors of the nodes $v_1, \ldots, v_m$ correspond to the output of C.

Now we show that our reduction is in logspace. Notice first that we introduce a constant number of nodes and edges for each gate. The weights of the edges are powers of two. Thus, we only need to store the exponents of the weights. If we write an edge weight to the output tape then we first write the "1" for the most significant bit of the weight and then we write "0" as often as determined by the exponent.

ii) Let $G^C = (V^C, E^C)$ be the graph arising from the graph G that is introduced in (i) by the following three operations. First, we omit the edges introduced by (c). Second, we add nodes s_i for $1 \leq i \leq n$. Third, we add an edge $\{s_i, v_{N-n+i}\}$ with weight 2^{N-n+i} for each $1 \leq i \leq n$. Let $s'_j := v_j$ for $N-n+1 \leq j \leq N$, $s' := (s'_1, \ldots, s'_n)$ and $t_j := v_j$ for $1 \leq j \leq m$. Then the nodes s_i for all $1 \leq i \leq n$ and t_i for all $1 \leq i \leq m$ are of degree one. Moreover, the node s'_i for any $1 \leq i \leq n$ is of Type I and influenced by s_i. Let P be a local optimum for G^C. Then $c_P(s_i) \neq c_P(s'_i)$ for all $1 \leq i \leq n$ due to Observation 2. Let c be the vector of the bitwise complement of $c_P(s')$. As in (i) it follows that $f(c) = c_P(t)$. Thus, $f(c_P(s)) = c_P(t)$. □

Note that G_f can be constructed in logarithmic space and thus in polynomial time for every polynomial-time computable function f. The result and its proof is used in several contexts in the rest of the thesis. To be able to refer to its parts, we introduce the following notations.

Definition 3.3.3. *For a Boolean circuit C we say that $G^C = (V^C, E^C)$ as constructed in the proof of the Constituting Theorem (i.e., Theorem 3.3.1) is the graph that* ***constitutes*** *C. Node $v_i \in V^C$ is said to* ***represent*** *gate g_i. Moreover, we call v_i a* ***NOT-node*** *if g_i is a NOT-gate in C, a* ***NOR-node*** *if g_i is a NOR-gate and a* ***gate-node*** *if it is a NOT-node or a*

NOR-node. The set of NOT-nodes is $\boldsymbol{V^C_{not}}$ *and the set of NOR-nodes is* $\boldsymbol{V^C_{nor}}$*. For a NOT-node* v_i *we let* $\boldsymbol{I(v_i)}$ *be the unique node that has influence on* v_i *and if it* v_i *is a NOR-node then we let* $\boldsymbol{I_1(v_i)}$ *and* $\boldsymbol{I_2(v_i)}$ *be the nodes representing the input gates of* g_i *in* C*. For a partition* $P \in \mathscr{P}(V^C)$ *and a NOT-node* $v_i \in V^C$ *we say that* v_i *is* ***correct*** *in* P *if it has the opposite color of* $I(v_i)$ *in* P*. Similarly, we call a NOR-node* $v_i \in V^C$ *correct in* P *if it is black if and only if the two nodes that represent the inputs of* g_i *in* C *are white. We call* P ***ordinary*** *if each node of* V^C *that represents an input gate of* C *is happy in* P*, the nodes that do not represent a gate are white in* P *if they have an odd index and black otherwise. For a polynomial-time computable function* f *and a Boolean circuit* C *that computes* f *we say that* $G_f := G^C$ ***looks at*** *the input nodes* $s_i \in V^C$ *and* ***biases*** *the output nodes* $t_i \in V^C$ *to the colors induced by* f*.*

Observation 3. *Let* C *be a Boolean circuit computing a function* $f : \{0,1\}^n \to \{0,1\}^m$ *for* $n, m \in \mathbb{N}$*,* $G^C = (V^C, E^C)$ *be the graph that constitutes* C*, and* $s := (s_1, \dots, s_n), t := (t_1, \dots, t_m)$ *for* $s_i, t_j \in V^C$ *for all* $1 \leq i \leq n$*,* $1 \leq j \leq m$ *be the vectors of nodes for which* $f(c_P(s)) = c_P(t)$ *in any local optimum* $P \in \mathscr{P}(V^C)$ *according to Theorem 3.3.1(ii). Then node* t_j *has no influence on the unique node adjacent to* t_j *in* G^C *for all* $1 \leq j \leq m$*.*

3.4 Is-Exp Property for Graphs with Nodes of Type I and III

In this section, we show the is-exp property for graphs with nodes of Type I and III by implementing a counter. The central part of the proof is a subgraph for which we show that it is possible to perform four flips of a certain node of the subgraph for every two flips of a different node of the subgraph. The construction of the proof is inspired by the proof of [27] in which Haken and Luby show the is-exp property for a problem closely related to LocalMax-Cut.

Theorem 3.4.1 (Is-Exp Theorem). LocalMax-Cut *has the is-exp property for graphs that contain only nodes of Type I and III.*

Proof. For $n \in \mathbb{N}_0$ we let C^n be the Boolean circuit depicted in Figure 3.4. We let $G^n = (V^n, E^n)$ be the graph that constitutes C^n. Recall that due to the construction in the proof of the Constituting Theorem (i.e., Theorem 3.3.1), the graph G^n contains a node v_i for every gate g_i of C^n. The graph G^n is depicted in Figure 3.5. Note that we assumed for graphs that constitute Boolean circuits that the output gates of the circuits are NOT-gates—in contrast to C^n. However, if the output link of g_1 is substituted by two NOT-gates linked in series, then the output of the thereby arising circuit is the same as the output of C^n. For the sake of simplicity, we omit these two gates in our description. The initial partition $P^n \in \mathscr{P}(V^n)$ can also be seen in Figure 3.5.

For a sequence $s := (v_{s_1}, v_{s_2}, \dots, v_{s_m})$ of improving flips starting at (G^n, P^n) with $m \in \mathbb{N}$, $1 \leq s_i \leq 12n + 13$ we write s^+ for the sequence $(v_{s'_1}, v_{s'_2}, \dots, v_{s'_m})$ where $s'_i := s_i + 4$ for all $1 \leq i \leq m$. Let $s(0) := (v_1, v_2, v_1)$ in G^0 and $s(n)$ in G^n for $n \geq 1$ be the sequence arising from $s(n-1)^+$ by inserting the following sequence of flips directly after the k-th

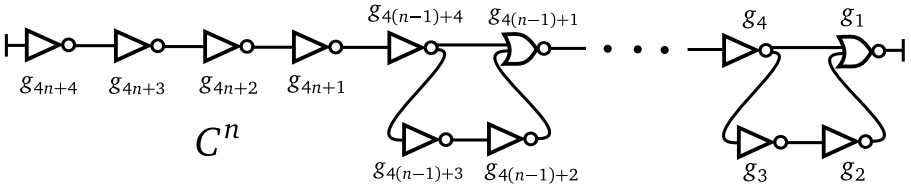


Figure 3.4: The infinite family of Boolean circuits C^n.

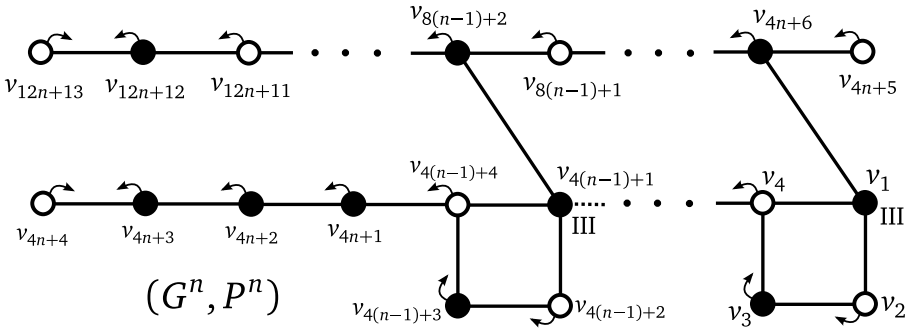


Figure 3.5: The graphs G^n and their initial partitions P^n.

flip of v_5 in $s(n-1)^+$ for all $1 \leq k \leq q$ where $q \in \mathbb{N}$ is the number of flips of v_5 in $s(n-1)^+$:

k is odd: insert $t_1 := (v_4, v_1, v_3)$
k is even: insert $t_2 := (v_4, v_1, v_2, v_1, v_3, v_2, v_1)$

For the sake of illustration, we state the first two sequences $s(1)$ and $s(2)$ in Example 3.4.2:

Example 3.4.2.
$$s(0) = (v_1, \quad v_2, v_1 \quad)$$
$$s(1) = (v_5, v_4, v_1, v_3, v_6, v_5, v_4, v_1, v_2, v_1, v_3, v_2, v_1)$$

In the following, we prove by induction on n that $s(n)$ is an improving sequence starting at (G^n, P^n) and node v_1 flips 2^{n+1} times in $s(n)$. For the induction basis, note that $s(0)$ is an improving sequence starting (G^0, P^0) and node v_1 flips 2^1 times in $s(0)$. Now assume as induction hypothesis that $s(n-1)$ is an improving sequence starting at (G^{n-1}, P^{n-1}) and v_1 flips 2^n times in $s(n-1)$. Notice first that after the first flip of v_5 in $s(n)$ the sequence t_1 is improving and that after the second flip of v_5 the sequence t_2 is improving. Since each node that flips in $t_1 \circ t_2$ flips an even number of times in $t_1 \circ t_2$, we get the following observation.

Observation 4. *Let $n \geq 1$, $W := \{v_1, \ldots, v_4\} \subset V^n$, $Q_0 \in \mathscr{P}(V^n)$, $s = (w_1, \ldots, w_q)$ be a sequence of flips starting at (G^n, Q_0) for $w_i \in V^n$, $q \in \mathbb{N}$ and $1 \leq j \leq q$ be an index for which $s_1^j|_W = t_1 \circ t_2$. Then $c_{Q_0}(v) = c_{Q_j}(v)$ for all $v \in W$.*

Observation 4 guarantees that the flips of t_1 are improving after each flip of v_5 to the white color in $s(n)$—as for its first flip. Thus, $s(n)$ is an improving sequence starting at

(G^n, P^n). Since v_1 flips four times in $t_1 \circ t_2$, it follows that v_1 flips in $s(n)$ twice as often as v_5 in $s(n-1)^+$, i.e., 2^n times. Thus, v_1 flips 2^{n+1} times in $s(n)$.

Since for all $n \in \mathbb{N}_0$ the graph G^n contains $O(n)$ nodes, the claim follows. □

Note that the only nodes of degree four in the graphs G^n of the above proof are the nodes v_{4i+1} for $0 \leq i < n$, i.e., the NOR-nodes. These nodes are of Type III and have, with the single exception of v_1, an incident edge that has no influence on their happiness. If to none of these nodes such an edge was incident, then we would get a graph with a degree of at most three in which only quadratically many flips are possible [49]. Thus, it is the existence of edges of this kind that allows exponentially long flip sequences although the edges do not affect the happiness of nodes of Type III.

Definition 3.4.3. *For $n \in \mathbb{N}_0$ we introduce the following names for objects introduced in the proof of the Is-Exp Theorem. We call the Boolean circuit C^n the **is-exp circuit** of length n. For the graph $G^n = (V^n, E^n)$ that constitutes C^n the initial partition $P^n \in V^n$ is called the **initial is-exp partition** of V^n. The sequence $s(n)$ is called **is-exp sequence** of dimension n. The sequence $s(n)^+$ is called the **shifted is-exp sequence** of dimension n, and the sequences t_1 and t_2 are called the **first** and the **second is-exp module**, respectively.*

3.5 Enforcing Technique for Graphs with Nodes of Type I, II and III

In this section, we develop a technique of enforcing any polynomial-time computable generalized pivot rule in certain partitioned graphs. The technique extends a given graph stepwise by further nodes and edges. In each step, an edge $\{u, v\}$ of a given graph is substituted by nodes and edges that, together with the nodes u and v, build up what is called a *basic* subgraph. At first, we introduce functions that encapsulate the substitution operations.

Then we devise a method that builds up a subgraph called *filter* in place of a heaviest edge of a node of Type I by iteratively using the substitution functions. The purpose of the filter is as follows. Let $G = (V, E)$ be a graph, $u \in V$, $v \in V_I$ and $e := \{u, v\} \in E$ such that e is the heaviest edge of v. If in a partition $P \in \mathscr{P}(V)$ edge e is in the cut and node u flips, then node v is instantly unhappy and could perform an improving flip. In the graph that contains the filter in place of e, node v does not immediately become unhappy after the flip of u. Instead, all nodes of the filter unequal to u and v must flip before v becomes unhappy. The method that builds up the filter allows to make the "walk" of the flips through the filter dependent on the value of an arbitrary Boolean SAT-formula in disjunctive normal form in which the variables of the formula correspond to nodes of V. Then, in certain partitions we are interested in, the flips migrate through the filter towards v if and only if the formula is satisfied with respect to the colors of the nodes that correspond to its variables.

Using the filters, we develop the technique of enforcing any polynomial-time computable generalized pivot rule in certain partitioned graphs. The given graphs constitute circuits and therefore contain only nodes of Type I and III. In our technique, we control

the happiness of the nodes of Type I—recall that these nodes represent the NOT-gates—by means of the filters such that exactly that node of Type I becomes unhappy that is chosen by the given generalized pivot rule according to the given partition. For the given circuit, we assume without loss of generality that the inputs and outputs of each NOR-gate are only NOT-gates. Then we can show that also the nodes of Type III—which represent the NOR-gates—and therefore all nodes of the graph flip exactly when they are chosen by the generalized pivot rule.

Prerequisite: Types of nodes The graphs considered in this section contain only nodes of Type I, II and III. For the purpose of succinctness, we assume for each introduced graph an implicit statement claiming one of the three types for each node.

3.5.1 Basic Subgraphs

The technique makes use of the following functions that extend a given graph by further nodes and edges.

Definition 3.5.1. *Let $G = (V, E)$ be a graph, $v \in V_I$, $u := H_G(v)$ and $e := \{u, v\} \in E$ where $w(e) = a$ for $a \in \mathbb{Q}_{>0}$. We let $\boldsymbol{G^1(G, v)}$ be the graph arising from G by substituting the edge e by the nodes and edges depicted in Figure 3.6. The values of $\epsilon, \delta \in \mathbb{Q}_{>0}$ are chosen small enough such that the following conditions are satisfied:*

- *Node v is of Type I in $G^1(G, v)$ and $H_{G^1(G,v)}(v) = g_2^1(v)$.*
- *Node $g_1^1(v)$ is of Type III in $G^1(G, v)$ and $T_{G^1(G,v)}(g_1^1(v)) = v$.*

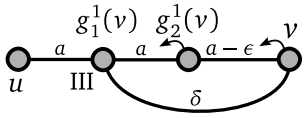


Figure 3.6: The subgraph introduced by the function $G^1(G, v)$.

Note that the degree of v in $G^1(G, v)$ is greater by one than in G.

Comment The purpose of the subgraph $G^1(\cdot)$ is to ensure that there is at most one edge on the path $(g_1^1(v), g_2^1(v), v)$ not in the cut in the partitions we are interested in. In particular, we choose for the initial partition of the three nodes of this path that $c(v) = c(g_1^1(v)) \neq c(g_2^1(v))$. Then, every sequence of improving flips started at the initial partition will retain the property that there is exactly one edge of the cycle $(g_1^1(v), g_2^1(v), v, g_1^1(v))$ not in the cut. The subgraphs that we will introduce below, these subgraphs may substitute the edges $\{g_2^1(v), v\}$ and $\{g_1^1(v), v\}$ by paths, together with their initial partition, will retain the property that in every sequence of improving flips in the subgraph arising from $G^1(\cdot)$ by adding them, there is in each partition induced by the sequence exactly one edge of each of the cycles not in the cut.

Definition 3.5.2. *Let $G = (V, E)$ be a graph, $v \in V_I \cup V_{III}$, $u := R_G(v)$ and $e := \{u, v\} \in E$ where $w(e) = a$ for $a \in \mathbb{Q}_{>0}$. We let $\boldsymbol{G^2(G, v)}$ be the graph arising from G by substituting the edge e by the nodes and edges depicted in Figure 3.7. The value of $\epsilon \in \mathbb{Q}_{>0}$ is chosen small enough such that v has the same type in $G^2(G, v)$ as in G and $R_{G^2(G,v)}(v) = g_2^2(v)$.*

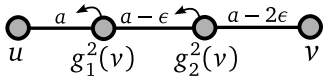


Figure 3.7: The subgraph introduced by the function $G^2(G, v)$.

Comment The purpose of the subgraph $G^2(\cdot)$ is to make the happiness of certain nodes dependent on the color of certain other nodes. On many occasions, this goal can be reached by drawing an edge of appropriate weight between the nodes. However, since we want to construct graphs with maximum degree four, we cannot draw edges between the nodes as often as it might be desirable. The subgraph $G^2(\cdot)$ is to overcome this obstacle in the following way. Let u be the node on whose color the happiness of some other node w is supposed to be made dependent and let G be a graph with an edge $e := \{u, v\}$. We use the subgraph $G^2(\cdot)$ to substitute e. For the partitions we will be interested in, we will ensure that all edges of $G^2(\cdot)$ are in the cut. Then node $g_2^2(v)$ has the same color as u. Thus, to reach the desired goal, we can draw an edge of appropriate weight between w and $g_2^2(v)$ instead of an edge between w and u—the weight of the added edge will be chosen to be smaller than ϵ to retain the property that the heaviest edge of $g_2^2(v)$ is $\{g_1^2(v), g_2^2(v)\}$. Moreover, if we want to make the happiness of w dependent on the opposite color of u then we can draw an edge between w and $g_1^2(v)$, and if we want to make the happiness of more than one or two nodes dependent on the color of u then we can even substitute the edge $\{g_2^2(v), v\}$ by $G^2(H, v)$ where H is the graph arising from G by the substitution of e by $G^2(G, v)$.

Definition 3.5.3. *Let $G = (V, E)$ be a graph, $v \in V_I$, $u := H_G(v)$ and $e := \{u, v\} \in E$ where $w(e) = a$ for $a \in \mathbb{Q}_{>0}$. We let $\boldsymbol{G^3(G, v)}$ be the graph arising from G by substituting the edge e by the nodes and edges depicted in Figure 3.8. The value of $\epsilon \in \mathbb{Q}_{>0}$ is chosen small enough such that v is of Type I in $G^3(G, v)$ and $H_{G^3(G,v)}(v) = g_5^3(v)$.*

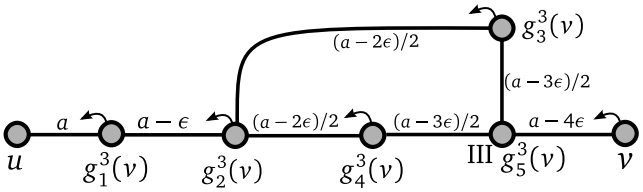


Figure 3.8: The subgraph introduced by the function $G^3(G, v)$.

Comment The purpose of the subgraph $G^3(\cdot)$ is to split an edge between u and v up in two paths. For the initial partition for the subgraph we will choose all edges to be in the cut. Then, if u flips and does not flip a second time before v flips, then the flips migrate along the two paths from u to v. Note that node $g_5^3(v)$ only becomes unhappy and flips when both $g_3^3(v)$ and $g_4^3(v)$ flipped before.

Definition 3.5.4. *Let $G = (V, E)$ be a graph, $v, w_1, w_2 \in V_I$, $u := H_G(v)$ and $e := \{u, v\} \in E$ where $w(e) = a$ for $a \in \mathbb{Q}_{>0}$. We let $\boldsymbol{G^4(G, v, w_1, w_2)}$ be the graph arising from G by substituting the edge e by the nodes and edges depicted in Figure 3.9. The value of $\epsilon \in \mathbb{Q}_{>0}$ is chosen small enough such that the following conditions are satisfied:*

- *Node v is of Type I in $G^4(G, v, w_1, w_2)$.*
- *$H_{G^4(G,v,w_1,w_2)}(v) = g_2^4(v)$.*
- *Node w_i is of type I in $G^4(G, v, w_1, w_2)$ for all $1 \leq i \leq 2$.*
- *$H_{G^4(G,v,w_1,w_2)}(w_i) = H_G(w_i)$ for all $1 \leq i \leq 2$.*

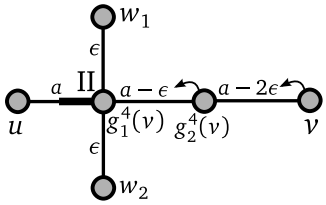


Figure 3.9: The subgraph introduced by the function $G^4(G, v, w_1, w_2)$.

Note that due to the weights of their incident edges, node $g_5^3(v)$ in $G^3(G, v)$ is of Type III and $g_1^4(v)$ in $G^1(G, v, w_1, w_2)$ is of Type II.

Comment The purpose of the subgraph $G^3(G, v, w_1, w_2)$ is to hinder the flips from migrating from u to v unless at least one of the nodes w_1, w_2 has, after the flip of u, the same color as u. More concretely, suppose that the edges on the simple path from u to v in $G^3(G, v, w_1, w_2)$ are in the cut. Suppose furthermore that u flips then and does not flip a second time before v flips for the first time. If at least one of the nodes w_1, w_2 has, after the flip of u, the same color as node u and neither w_1 nor w_2 flips prior to the first flip of v, then the there will be consecutive flips of the nodes $g_1^4(v)$, $g_2^4(v)$ and v in every sequence of improving flips started at the partition after the flip of u. On the other hand, if both $g_1^4(v)$ and $g_2^4(v)$ have the opposite color as u after the flip of u, then at least one of the two nodes must flip before $g_1^4(v)$ and then $g_2^4(v)$ and then v can flip.

Definition 3.5.5. *Let $G = (V, E)$ be a graph. We let $\boldsymbol{G^5(G)}$ be the graph arising from G by introducing two nodes g_1^5, g_2^5 and an edge $\{g_1^5, g_2^5\}$ with weight 1.*

For the identification of certain nodes and edges, we introduce the following notations.

Definition 3.5.6. *Let $G = (V, E)$ be a graph, $v \in V_I \cup V_{III}^3$ and $w_1, w_2 \in V_I$. Let $S_i(v)$ for $1 \leq i \leq 3$ be the subgraph of $G^i(v)$ induced by the nodes v, $R_G(v)$ and $g_j^i(v)$ for all j and $S_4(v)$ be the subgraph of $G^4(v, w_1, w_2)$ induced by the nodes v, $H_G(v)$, $g_1^4(v)$ and $g_2^4(v)$. For each node w of $S_i(v)$ for any $1 \leq i \leq 4$ we call every edge incident to w that is on a simple path from $R_G(v)$ to v in S_i a* ***heavy edge*** *of w.*

Definition 3.5.7. *For all $n \in \mathbb{N}$ we let*

- $\boldsymbol{r(n)} := \lceil n/2 \rceil$
- $\boldsymbol{p_n^+(i)} := (i \mod n) + 1$ *for* $1 \leq i \leq n$
- $\boldsymbol{p_n^-(i)} := ((i-2) \mod n) + 1$ *for* $1 \leq i \leq n$
- $\boldsymbol{pr_n^+(i)} := r(p_n^+(i))$ *for* $1 \leq i \leq n$
- $\boldsymbol{pr_n^-(i)} := r(p_n^-(i))$ *for* $1 \leq i \leq n$.

Definition 3.5.8. *Let $G = (V, E)$ be a graph and $\phi \in \Phi(V)$. We let* $\boldsymbol{Nodes(\phi)}$ *be the set of all nodes $v \in V$ contained in ϕ. For a subset $W \subseteq V$ and a function $\varphi : W \to \Phi(W)$ we let* $\boldsymbol{D(\varphi)} := \{v \in W | \varphi(v) \neq \emptyset\} \cup \bigcup_{v \in W} Nodes(\varphi(v))$ *and for a literal l over a variable v we let* $\boldsymbol{nod(l)} := v$.

In the description of the way in which the subgraphs are combined, we make use of the following conventions.

Prerequisite: In the rest of the chapter, we treat nodes of a graph $G = (V, E)$ also as Boolean variables of Boolean formulas and let the values of the variables be induced by the colors of the nodes in a given partition $P \in \mathscr{P}(V)$. Moreover, for a Boolean formula $\phi \in \Phi(V)$ we let $\boldsymbol{val_P(\phi)} := val_t(\phi)$ where t is the truth assignment induced by assigning the value *true* to a variable $v \in V$ if and only if $c_P(v) = 1$.

Properties of the subgraphs

Observation 5. *Let $G = (V, E)$ be a graph, $v \in V_I \cup V_{III}^3$, $w_1, w_2 \in V_I$, $u := R_G(v)$, $G_i = (V_i, E_i) := G^i(G, v)$ for $1 \leq i \leq 3$, $G_4 = (V_4, E_4) := G^4(G, v, w_1, w_2)$ and $G_5 = (V_5, E_5) := G^5(G)$. Then the following conditions are satisfied:*

i) $deg_{G_1}(v) = deg_G(v) + 1$.

ii) $deg_{G_i}(v) = deg_G(v)$ *for all* $2 \leq i \leq 4$.

iii) $deg_{G_4}(w_i) = deg_G(w_i) + 1$ *for all* $1 \leq i \leq 2$.

iv) $deg_{G_i}(w) = deg_G(w)$ *for all* $1 \leq i \leq 5$ *and* $w \in V \setminus \{v\}$.

v) $deg_{G_i}(w) \leq 4$ *for all* $1 \leq i \leq 5$ *and* $w \in V_i \setminus V$.

vi) Node $w \in V_i \setminus V$ for any $1 \leq i \leq 4$ is happy in a partition $P \in \mathscr{P}(V_i)$ if a heavy edge of w with greatest weight among the heavy edges of w and one further heavy edge of w are in the cut in P.

3.5.2 Combining the Subgraphs

In this section, we describe the function enf_5 whose purpose is as follows. If for a graph $G = (V, E)$ and a node $v \in V_I$ the node $u := H_G(v)$ flips to the same color as v and thereby turns v unhappy, then v could flip directly after the flip of u. The aim of the function enf_5 is to hinder v from becoming unhappy, and thereby from flipping, as long as a given condition is not satisfied. In particular, the condition is formulated as a Boolean SAT-formula $\varphi(v)$ for $\varphi : V_I \to \Phi(V_I)$ in disjunctive normal form. To reach the desired goal, we substitute the edge $\{u, v\} \in E$ by a subgraph which is iteratively built up by the subgraphs introduced in the previous section. In fact, the subgraph is called the **filter** of v and is the subgraph induced by the nodes u, v and all nodes that are introduced by the function enf_5 and substitute either the edge $\{u, v\}$ or an edge of the subgraph that substituted the edge $\{u, v\}$ (a formal definition of the filter will be given in Definition 3.5.9). The substitutions performed by the function enf_5 are divided into five parts.

A rough overview of the filter is illustrated in Figure 3.10. It shows a mapping of subgraphs of the filter of a Type I node v to the parts of the function enf_5 in which they are added. The main purpose of the filter is to split the edge $\{u, v\}$ up into several paths between u to v. There is one path set aside from the others. We call this path the **braid** of v. The subgraph induced by the other paths is called the **head** of v (a formal definition of the head and the braid will be given in Definition 3.5.9). The braid can be seen in Figure 3.10 as the path between u and v containing only nodes that are added in the parts one and four. Each path of the head contains a node that is added in part i for all $1 \leq i \leq 5$. The purpose of the paths that make up the head is as follows. If in a given partition the SAT-formula corresponding to v is satisfied, then all nodes of the head paths can flip their colors consecutively. However, if the formula is not satisfied then on each head path there is a node that remains happy. Altogether, after a flip of u, assuming that neither u nor the nodes of the SAT-formula corresponding to v change their colors, the flips pass the head paths towards v and thereby make v unhappy if and only if the value of the formula is true.

The description of the five parts is done via the functions enf_i for $1 \leq i \leq 5$. For each $1 < i \leq 5$ the function enf_i first calls as subroutine the function enf_{i-1} that performs the parts $1, \ldots, i-1$ and then it adds further subgraphs in place of edges.

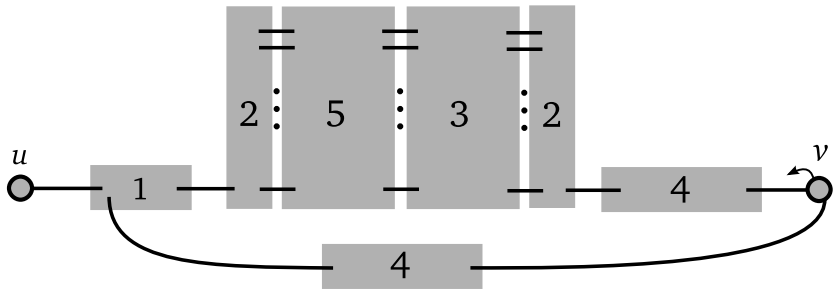


Figure 3.10: Parts of the filter of v labelled by the part of enf_5 in which they are added.

Part 1 The pseudo-code for the function enf_1, which makes up the first part, is shown in Algorithm 3.1. In it, we substitute for each $v \in D(\varphi)$ the edge $\{H_G(v), v\}$ by the subgraph introduced by the function $G^1(\cdot)$ and let $\alpha_1(v)$ and $\alpha_2(v)$ be the nodes introduced by this function—see line 6 of Algorithm 3.1. We call the set of nodes $\{\alpha_1(v), \alpha_2(v)\}$ the **capsule** of v.

Input: graph $G = (V, E)$, function $\varphi : V_I \to \Phi(V_I)$
Output: graph $G' = (V', E')$

1: $k \leftarrow 1$
2: $H^0 \leftarrow G$
3: **for all** $v \in D(\varphi)$ **do**
4: $\quad H^k \leftarrow G^1(H^{k-1}, v)$ ▷ Add capsule
5: $\quad k \leftarrow k + 1$
6: $\quad \alpha_1(v) \leftarrow g_1^1(v); \alpha_2(v) \leftarrow g_2^1(v)$ ▷ Rename added nodes
7: **return** H^{k-1}

Algorithm 3.1: The function enf_1.

Part 2 The function enf_2 (see Algorithm 3.2) first calls as a subroutine the function enf_1 and then it substitutes for all $v \in D(\varphi)$ the edge $\{v, \alpha_2(v)\}$ by a subgraph that is built up by iteratively introducing subgraphs according to the function $G^3(\cdot)$ twice as often as there are monomials in the formula $\varphi(v)$. The nodes of the thereby introduced subgraphs are called $\beta_j^i(v)$ for $1 \leq j \leq 5$ and $1 \leq i \leq 2n$ where n is the number of monomials of $\varphi(v)$—see line 9 of Algorithm 3.2. The subgraph that contains the nodes and edges that substitute the edge $\{v, H_G(v)\}$ after the first two parts of the function enf_5 is depicted in Figure 3.11. We call the nodes introduced in the iteration of the for-loop in lines 3–9 corresponding to v the **splitters** of v.

Comment The nodes and edges introduced in the second part substitute the edge $\{v, \alpha_2(v)\}$ by $2n + 1$ simple paths between $\alpha_2(v)$ and v.

Part 3 The third part (see Algorithm 3.3) substitutes the heaviest edge of node $\beta_4^i(v)$ for all $1 \leq i \leq 2n$ introduced in the previous part by a subgraph that is built up by iteratively adding subgraphs according to $G^2(\cdot)$ once more as twice as often as there are literals in the monomial M_r of $\varphi(v)$ for $r := pr_{2n}^-(i)$ and call the new nodes $\gamma_{j,k}^i(v)$ for $1 \leq j \leq 2r + 1$, $1 \leq k \leq 2$—see line 12 of Algorithm 3.3. Moreover, we call the nodes introduced in the iteration of the for-loop in lines 3–12 corresponding to v the **internal informers** of v. The subgraphs are built up in a way such that instead of the edge $\{H_G(\beta_4^i(v)), \beta_4^i(v)\}$ for $1 \leq i \leq 2n + 1$ a path is introduced—note that $H_G(\beta_4^i(v)) = \beta_2^i(v)$ for all $1 \leq i \leq 2n$. In case of $i = 2n + 1$ the edge is not substituted, i.e., the path only consists of a single edge. For each $1 \leq i \leq 2n + 1$ we let $\boldsymbol{p_i^\varphi(v)}$ be the path that substitutes the edge $\{H_G(\beta_4^i(v)), \beta_4^i(v)\}$.

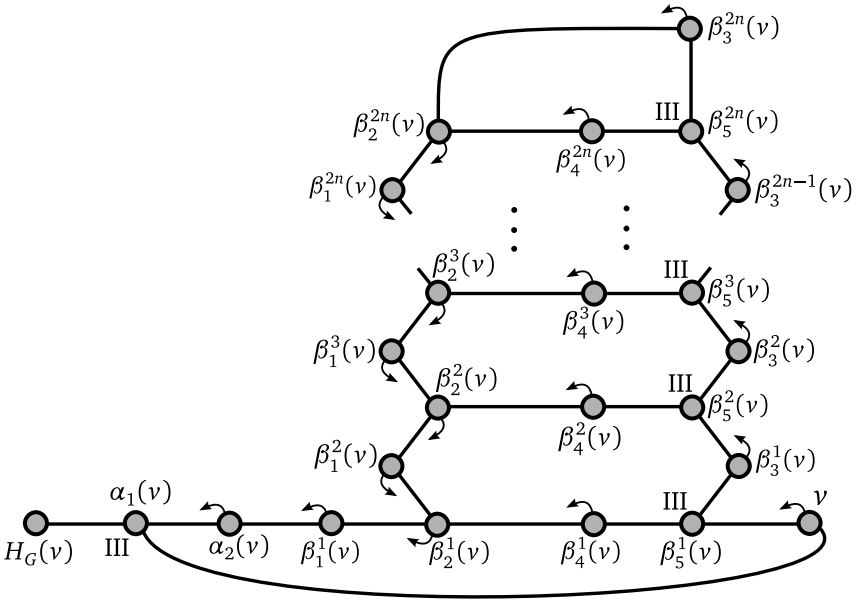


Figure 3.11: The subgraph containing the capsule and the splitters of v.

Comment The purpose of the nodes on the path $p_i^{\varphi}(v)$ for $1 \leq i \leq 2n$ introduced in this part is to reflect whether $p_i^{\varphi}(v)$ was already passed by the flips that migrate towards v.

Part 4 In the fourth part, we introduce four subgraphs for each literal $l_{r,j}$ of $\varphi(v)$ of any $v \in D(\varphi)$ for $1 \leq r \leq n, 1 \leq j \leq m_r$ where m_r is the number of literals of monomial M_r—see Algorithm 3.4. We call the nodes introduced in the iteration of the for-loop in lines 3–13 corresponding to v the **external informers** of v. Two of the four subgraphs substitute the heaviest edge of u and together build up a path in place of that edge. We call the external informers $\delta_{j,k}^r(v)$ for any r, j, k as introduced in line 9 **anterior** and the external informers $\eta_{j,k}^r(v)$ for any r, j, k as introduced in line 10 **posterior**.

Comment The purpose of the external informers is to reflect the color of the node $u := nod(l_{r,j})$, in a way that is examined closer in the description of the next part, to the nodes of the filter of v.

Part 5 In the fifth part we again introduce four subgraphs for each literal $l_{r,j}$ for $1 \leq r \leq n, 1 \leq j \leq m_r$ of $\varphi(v)$ of any $v \in D(\varphi)$ (see lines 13 and 15 of Algorithm 3.5) and for each monomial M_r two further subgraphs (see line 20). We call the nodes introduced in the lines 13 and 15 of the iteration of the for-loop in lines 3–13 corresponding to v the **delayers** of v. The nodes added in line 20 are called the **constants** of v. For $1 \leq r \leq n$, $1 \leq j \leq m_r$, $w := nod(l_{r,j})$ and $p := p_{2n}^-(2r-1)$ the delayers that correspond to the literal $l_{r,j}$ and the nodes they are adjacent to are presented in Figure 3.12. Before describing the purpose of the nodes added in

Input: graph $G = (V, E)$, function $\varphi : V_I \to \Phi(V_I)$
Output: a graph $G' = (V', E')$

1: $k \leftarrow 1$
2: $H^0 \leftarrow enf_1(G)$ ▷ Execute first step
3: **for all** $v \in D(\varphi)$ **do**
4: $\quad \beta_3^0(v) \leftarrow v$
5: $\quad$ **for** $i \leftarrow 1$ **to** $2 \cdot |Mons(\varphi(v))|$ **do**
6: $\quad\quad H^k \leftarrow G^3(H^{k-1}, \beta_3^{i-1}(v))$ ▷ Add splitters
7: $\quad\quad k \leftarrow k + 1$
8: $\quad\quad$ **for** $j \leftarrow 1$ **to** 5 **do**
9: $\quad\quad\quad \beta_j^i(v) \leftarrow g_j^3(\beta_3^{i-1}(v))$ ▷ Rename added nodes
10: **return** H^{k-1}

Algorithm 3.2: The function enf_2.

the fifth part, we first introduce some notations. For future reference we give the pair of prerequisite and definition the name given below:

Filter Definitions (First Part) For the sake of succinctness, we make the following assumption:

Prerequisite: In all remaining Definitions, Observations and Lemmas unequal to the last Lemma, namely the Filtering Lemma (i.e., Lemma 3.5.21), we let $G = (V, E)$ be a graph, $\varphi : V_I \to \Phi(V_I)$, $G^\varphi = (V^\varphi, E^\varphi) := enf_5(G, \varphi)$, $v \in D(\varphi)$, $n^v := |Mons(\varphi(v))|$, M_r^v for $1 \leq r \leq n^v$ be the monomials of $\varphi(v)$, $m_r^v = |Lits(M_r^v)|$ for $1 \leq r \leq n^v$ and $l_{r,j}^v$ for $1 \leq j \leq m_r^v$ be the literals of M_r^v, i.e., the elements of the set $Lits(M_r^v)$. If the considered node is clear from the context then we may omit the superscript that indicates the node.

Definition 3.5.9. *We denote by* $\boldsymbol{F^\varphi(v)}$ *the subset of* V^φ *containing the following nodes:*

F1) v

F2) $I_G(v)$

F3) $\alpha_1(v)$, $\alpha_2(v)$

F4) $\beta_j^i(v)$ *for all* $1 \leq i \leq 2n^v$, $1 \leq j \leq 5$

F5) $\gamma_{j,k}^i(v)$ *for all* $1 \leq i \leq 2n^v$, $1 \leq j \leq 2m_p^v + 1$ *for* $p := pr_{2n^v}^-(i)$ *and* $1 \leq k \leq 2$

F6) $\delta_{j,k}^i(w)$ *for all* $w \in D(\varphi)$, $1 \leq i \leq 2n^w$, $1 \leq j \leq m_r^w$ *for* $r := r(i)$, $1 \leq k \leq 2$ *for which* $nod(l_{i,j}^w) = v$

Input: graph $G = (V, E)$, function $\varphi : V_I \to \Phi(V_I)$
Output: a graph $G' = (V', E')$

1: $k \leftarrow 1$
2: $H^0 \leftarrow enf_2(G, \varphi)$ ▷ Execute first two steps
3: **for all** $v \in D(\varphi)$ **do**
4: $n \leftarrow |Mons(\varphi(v))|$
5: **for** $i \leftarrow 1$ **to** $2 \cdot n$ **do**
6: $\beta \leftarrow \beta_4^i(v)$
7: $r \leftarrow pr_{2n}^+(i)$
8: $M \leftarrow r$-th monomial of $\varphi(v)$
9: **for** $j \leftarrow 1$ **to** $2 \cdot |Lits(M)| + 1$ **do**
10: $H^k \leftarrow G^2(H^{k-1}, \beta)$ ▷ Add internal informers
11: $k \leftarrow k + 1$
12: $\gamma_{j,1}^i(v) \leftarrow g_1^2(\beta); \gamma_{j,2}^i(v) \leftarrow g_2^2(\beta)$ ▷ Rename added nodes
13: **return** H^{k-1}

Algorithm 3.3: The function enf_3.

F7) $\eta_{j,k}^i(w)$ *for all* $w \in D(\varphi)$, $1 \le i \le 2n^w$, $1 \le j \le m_r^w$ *for* $r := r(i)$, $1 \le k \le 2$ *for which* $nod(l_{i,j}^w) = v$

F8) $\phi_{j,k}^i(v)$ *for all* $1 \le i \le 2n^v$, $1 \le j \le 2m_r^v + 1$ *for* $r := r(i)$ *and* $1 \le k \le 2$.

The subgraph of G^φ *induced by the nodes of* $F^\varphi(v)$ *is called the* ***filter*** *of* v *with respect to* φ. *We write* $\boldsymbol{T^\varphi(v)}$ *for the set of nodes that contains all nodes of (F6) and* v *itself and we write* $\boldsymbol{B^\varphi(v)}$ *for the set of nodes that contains all nodes of (F7) and* v *itself. Moreover, we let* $\boldsymbol{H^\varphi(v)} := (F^\varphi(v) \setminus B^\varphi(v)) \cup \{v\}$. *We call the subgraph of* G^φ *induced by the nodes of* $T^\varphi(v)$ *the* ***throat*** *of* v *with respect to* φ, *the subgraph induced by the nodes of* $B^\varphi(v)$ *the* ***braid*** *of* v *with respect to* φ, *and the subgraph induced by the nodes of* $H^\varphi(v)$ *the* ***head*** *of* v *with respect to* φ. *Finally, we let* $\boldsymbol{R^\varphi(v)}$ *be the set containing the nodes of* $F^\varphi(v)$ *without the nodes* $I_G(v)$, $\alpha_1(v)$, $\phi_{j,k}^i(v)$ *for all* i, j, k *and* $\beta_5^i(v)$ *for all* i.

Comment Note that since the nodes of $T^\varphi(v) \setminus \{v\}$ are added via calls of the function $G^2(\cdot)$ in line 9 of Algorithm 3.4 that always substitute the heaviest edge incident to the node v, the throat of v is a path in $F^\varphi(v)$. Similarly, since the nodes of $B^\varphi(v) \setminus \{v\}$ are added via calls of the function $G^2(\cdot)$ in line 10 that always substitute the third edge of the node $\alpha_1(v)$, the braid of v is also a path.

We now continue with the description of the purpose of the nodes added in the fifth part. Recall that the idea of the filter of v is to split the edge $\{u, v\}$ up in paths and delay the flips on their migration from u to v depending on whether the corresponding formula is satisfied. The first four parts provided the split-up of the edge $\{u, v\}$, they provided nodes whose colors are supposed to indicate the colors

Input: graph $G = (V, E)$, function $\varphi : V_I \to \Phi(V_I)$
Output: a graph $G' = (V', E')$

1: $k \leftarrow 1$
2: $H^0 \leftarrow enf_3(G, \varphi)$ ▷ Execute first three steps
3: **for all** $v \in D(\varphi)$ **do**
4: $n \leftarrow |Mons(\varphi(v))|$
5: **for** $i \leftarrow 1$ **to** 2n **do**
6: $r \leftarrow r(i)$
7: $M \leftarrow r$-th monomial of $\varphi(v)$
8: **for** $j \leftarrow 1$ **to** $|Lits(M)|$ **do**
9: $H^k \leftarrow G^2(H^{k-1}, nod(l_{r,j}))$ ▷ Add anterior external informers
10: $H^{k+1} \leftarrow G^2(H^k, \alpha_1(nod(l_{r,j})))$ ▷ Add posterior external informers
11: $k \leftarrow k + 2$
12: $\delta^i_{j,1}(v) \leftarrow g^2_1(nod(l_{r,j}))$; $\delta^i_{j,2}(v) \leftarrow g^2_2(nod(l_{r,j}))$ ▷ Rename added nodes
13: $\eta^i_{j,1}(v) \leftarrow g^2_1(\alpha_1(nod(l_{r,j})))$; $\eta^i_{j,2}(v) \leftarrow g^2_2(\alpha_1(nod(l_{r,j})))$
14: **return** H^{k-1}

Algorithm 3.4: The function enf_4.

of the nodes that correspond to the literals of $\varphi(v)$ and they provided nodes whose colors are supposed to indicate whether one of the paths $p^\varphi_i(v)$ for $1 \leq i \leq 2n$ was passed by the flips that migrate through the filter along the head paths from u to v. In the following, we explain the purposes of the delayers by means of their supposed functionality with respect to the nodes they are adjacent to.

First, we consider the interaction between the delayers and the constants. In the partitions of the filter we will be interested in, all edges on the $2n+1$ simple paths from $\alpha_1(v)$ to v containing a node $\beta^i_4(v)$ along heavy edges, i.e., the head paths, are in the cut. Then the colors of the nodes on the head paths are determined by the color $\kappa \in \{0, 1\}$ of v. However, the satisfaction of the formula $\varphi(v)$ depends on the colors of the nodes of $Nodes(\varphi(v))$ and does not necessarily depend on the color of v. Therefore, we introduced two paths $p^\varphi_{2r-1}(v)$ and $p^\varphi_{2r}(v)$ for each monomial M_r for $1 \leq r \leq n$ of $\varphi(v)$ in the third part and introduce in part five delayers that let the flips pass towards the paths $p^\varphi_j(v)$ for odd $1 \leq j \leq 2n-1$ if $\kappa = 0$ and furthermore introduce delayers in the same part that let the flips pass towards the paths $p^\varphi_j(v)$ for even $2 \leq j \leq 2n$ if $\kappa = 1$. The delayers for this purpose are added in line 20 and the corresponding constants in line 18. We will later assign colors to the constants such that $c^i_1(v)$ is and remains white for odd i and black for even i.

Second, we explain the interaction between the delayers of v and the external informers of v. The idea of the external informers of v is to reflect the colors of the nodes of $Nodes(\varphi(v))$ to the delayers introduced in lines 13 and 15. However, in

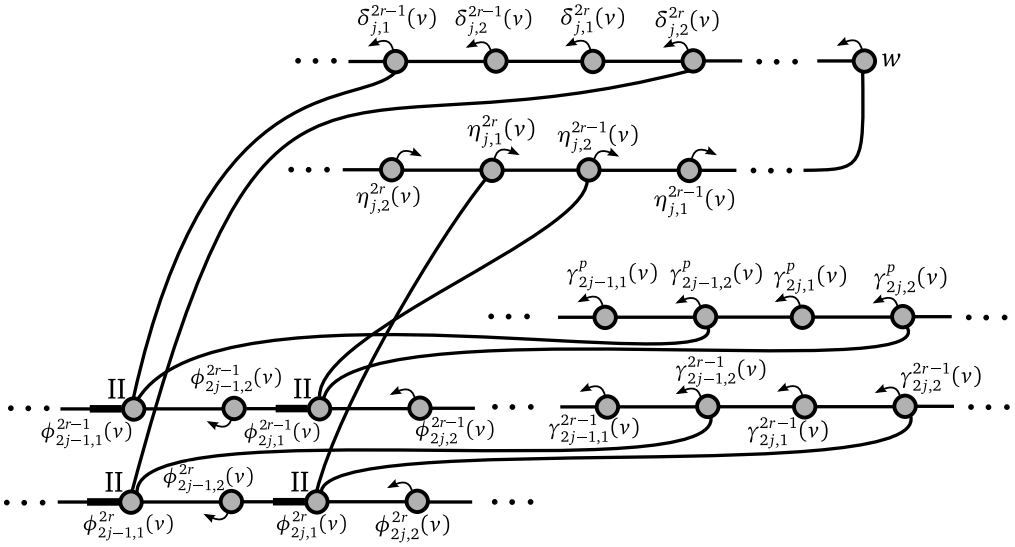


Figure 3.12: The adjacent nodes of the delayers of v.

some partitions that we will later deal with, we cannot deduce the color of a node $w \in Nodes(\varphi(v))$ from the color of an arbitrary anterior external informer of v with respect to w. Therefore, we also introduce the posterior external informers. We will later see that if an edge between the nodes of $T^{\varphi}(w)$ is not in the cut, then all edges between the nodes of $B^{\varphi}(w)$ are in the cut. Then we can conclude that of two nodes w_1, w_2 for $w_1 \in T^{\varphi}(w)$ and $w_2 \in B^{\varphi}(w)$ with equal distance in $F^{\varphi}(w)$ from w modulo 2 at least one has the same color as w. Therefore, we introduce in the lines 13 and 15 one delayer that is adjacent to an anterior external informer and one delayer that is adjacent to a posterior external delayer of the node $nod(l_{r,j})$ for the corresponding literal $l_{r,j}$ for $1 \leq r \leq n$, $1 \leq j \leq m_r$. Depending on whether the literal $l_{r,j}$ is negated or not, we alternate between the distance modulo 2 of the corresponding external informers from $nod(l_{r,j})$ in $F^{\varphi}(v)$—see lines 13 and 15 again.

Third and finally, we describe the interaction between the delayers of v and the internal informers of v. According to the function $G^4(\cdot)$, the nodes $\phi^i_{j,1}(v)$ for $1 \leq i \leq 2n$, $1 \leq j \leq 2m_r$ for $r := r(i)$ are, in addition to the two nodes to which they are adjacent via their heavy edges, adjacent to two further nodes. For one of the two further nodes, which is either a constant or an external informer, we already know the purpose. The other node adjacent to $\phi^i_{j,1}(v)$ is an internal informer on the path $p_{i'}$ for $i' := p^-_{2n^v}(i)$ according to lines 13 and 15. In some of the partitions we will be interested in, these edges between the delayers and the internal informers will be in the cut. Now assume that after such a partition arises in a sequence of flips, the flips pass the path $p^{\varphi}_{i'}(v)$. Then all internal informers on that path change their colors. But then one non-heavy edge of the delayers $\phi^i_{j,1}(v)$ for all $1 \leq j \leq 2m_r + 1$ for $r := r(i)$ is not in the cut. Then, the flips can also pass

towards the path $p_i^{\varphi}(v)$. Altogether, all paths can be passed and the flips migrate towards v. This finishes the description of the function enf_5.

Input: graph $G = (V, E)$, function $\varphi : V_I \to \Phi(V_I)$
Output: a graph $G' = (V', E')$

$k \leftarrow 1$
$H^0 \leftarrow enf_4(G, \varphi)$ ▷ Execute first four steps
for all $v \in D(\varphi)$ **do**
  $n \leftarrow |Mons(\varphi(v))|$
  for $i \leftarrow 1$ **to** 2n **do**
    $\gamma \leftarrow \gamma^i_{1,1}(v)$
    $p \leftarrow p^-_{2n}(i)$
    $r \leftarrow r(i)$
    $M \leftarrow r$-th monomial of $\varphi(v)$
    $t \leftarrow |Lits(M)|$
    for $j \leftarrow 1$ **to** t **do**
      $q \leftarrow (pos(l_{r,j}) + i) \mod 2$
      $H^k \leftarrow G^4(H^{k-1}, \gamma, \delta^i_{j,2-q}(v), \gamma^p_{2j-1,2}(v))$ ▷ Add delayers
      $\phi^i_{2j-1,1}(v) \leftarrow g^4_1(\gamma); \phi^i_{2j-1,2}(v) \leftarrow g^4_2(\gamma)$ ▷ Rename added nodes
      $H^{k+1} \leftarrow G^4(H^k, \gamma, \eta^i_{j,1+q}(v), \gamma^p_{2j,2}(v))$ ▷ Add delayers
      $\phi^i_{2j,1}(v) \leftarrow g^4_1(\gamma); \phi^i_{2j,2}(v) \leftarrow g^4_2(\gamma)$ ▷ Rename added nodes
      $k \leftarrow k + 2$
    $H^k \leftarrow G^5(H^{k-1})$ ▷ Add constants
    $c^i_1(v) \leftarrow g^5_1; c^i_2(v) \leftarrow g^5_2$ ▷ Rename constants
    $H^{k+1} \leftarrow G^4(H^k, \gamma, c^i_1(v), \gamma^p_{2t+1,2}(v))$ ▷ Add delayers for constants
    $k \leftarrow k + 2$
    $\phi^i_{2t+1,1}(v) \leftarrow g^4_1(\gamma); \phi^i_{2t+1,2}(v) \leftarrow g^4_2(\gamma)$ ▷ Rename added nodes
$G' \leftarrow H^{k-1}$

Algorithm 3.5: The function enf_5.

Now we introduce some notations that we need for the description of some properties of the graph arising by the call of the function enf_5. For future reference we give the following block of definitions the name given below:

Filter Definitions (Second Part)

Definition 3.5.10. *We let $\boldsymbol{cy^{\varphi}_v(i)}$ for $1 \le i \le 2n^v$ be the unique cycle along nodes of $F^{\varphi}(v)$ containing the nodes v, $\alpha_1(v)$ and $\beta^i_4(v)$. Similarly, we let $cy^{\varphi}_v(2n^v+1)$ be the unique cycle in $F^{\varphi}(v)$ containing the nodes v, $\alpha_1(v)$ and $\beta^{2n^v}_3(v)$. For $1 \le i \le 2n^v+1$ we let $\boldsymbol{sp^{\varphi}_v(i)}$ be the subpath of $cy^{\varphi}_v(i)$ starting at the unique node of $B^{\varphi}(v) \setminus \{v\}$ incident to v, containing $\alpha_1(v)$ and ending at v.*

Definition 3.5.11. *Let $P \in \mathscr{P}(V^\varphi)$. We call $T^\varphi(v)$ **flat** in P if all edges of the cycle $cy_v^\varphi(1)$ incident to nodes of $T^\varphi(v) \setminus \{v\}$ are in the cut in P and, similarly, we call $B^\varphi(v)$ flat in P if all edges of the cycle $cy_v^\varphi(1)$ incident to nodes of $B^\varphi(v) \setminus \{v\}$ are in the cut in P. If the considered partition is clear from the context then we omit for any of the aforementioned definitions the expression "in P" for the corresponding partition P.*

*We say that $F^\varphi(v)$ is **straight** in P if $c_P(c_1^{2r}(v)) = c_P(c_2^{2r-1}(v)) = 1$ and $c_P(c_2^{2r}(v)) = c_P(c_1^{2r-1}(v)) = 0$ for all $1 \leq r \leq n^\nu$. Suppose that $F^\varphi(v)$ is straight in P. Then we call $F^\varphi(v)$ **canonical** in P if on each cycle $cy_v^\varphi(i)$ for $1 \leq i \leq 2n^\nu + 1$ there is exactly one edge not in the cut. Now suppose that $F^\varphi(v)$ is canonical in P. Then we call $F^\varphi(v)$ **enterable** in P if exactly one edge between the nodes of $B^\varphi(v) \cup \{v, \alpha_1(v)\}$ is not in the cut and the edge $\{I(v), \alpha_1(v)\}$ is in the cut. Furthermore, we call $F^\varphi(v)$ **just entered** in P if P can be reached from an enterable partition by a flip of $I(v)$ and we call $F^\varphi(v)$ **awaiting** in P if it is enterable or just entered in P.*

Definition 3.5.12. *Let $P \in \mathscr{P}(V^\varphi)$ such that $F^\varphi(v)$ is canonical in P, $y \in B^\varphi(v)$ be the unique node for which $\{y, v\} \in E^\varphi$ and $1 \leq i \leq 2n^\nu + 1$. We denote by $\boldsymbol{e_v^\varphi(P,i)}$ the unique edge of $cy_v^\varphi(i)$ that is not in the cut in P. For a node w on the cycle $cy_v^\varphi(i)$ we let $\boldsymbol{dis_v^\varphi(w,i)}$ be the number of edges of the subpath of $sp_v^\varphi(i)$ starting at w and ending at v. We let the function $\boldsymbol{n_v^\varphi(P,i)}$ return a node of $F^\varphi(v)$ in the following way. If $e_v^\varphi(P,i) = \{y, v\}$ then it returns y. Otherwise it returns the node u_i for $\{u_i, v_i\} := e_v^\varphi(P,i)$, $u_i, v_i \in V^\varphi$ for which $dis_v^\varphi(u_i,i) < dis_v^\varphi(v_i,i)$. Furthermore, we denote by $\boldsymbol{t_v^\varphi(P,i)}$ the node adjacent to $n_v^\varphi(P,i)$ via the edge $e_v^\varphi(P,i)$, i.e., $e_v^\varphi(P,i) = \{n_v^\varphi(P,i), t_v^\varphi(P,i)\}$. Finally, we call $\boldsymbol{d_v^\varphi(P)} := \sum_{1 \leq i \leq 2n^\nu+1} dis_v^\varphi(n_v(P,i),i)$ the **potential** of v in P.*

Definition 3.5.13. *Let $P \in \mathscr{P}(V^\varphi)$ and $P' \in \mathscr{P}(V^\varphi)$ be the partition arising from P by choosing the colors of the nodes of $F^\varphi(v) \setminus \{u\}$ such that $F^\varphi(v)$ is enterable in P' where the edge of the braid of v incident to v is not in the cut. For a node $x \in F^\varphi(v) \setminus \{u\}$ we call $c_{P'}(x)$ the **natural color** of x in P and the opposite color its **unnatural color** in P.*

Definition 3.5.14. *Let $1 \leq i \leq 2n^\nu + 1$, $P \in \mathscr{P}(V^\varphi)$ such that $F^\varphi(v)$ is canonical in P and $y \in B^\varphi(v)$ be the unique node for which $\{v, y\} \in E^\varphi$. We call the subpath of $sp_v^\varphi(i)$ starting at $n_v^\varphi(P,i)$ and ending at v the **i-th upper path** of $F^\varphi(v)$ in P. The subpath of $cy_v^\varphi(i)$ starting at $t_v^\varphi(P,i)$ and ending at y along edges that are not in the i-th upper path is called the **i-th lower path** of $F^\varphi(v)$ in P.*

Comment The i-th upper path of $F^\varphi(v)$ in P is equal to $sp_v^\varphi(v)$ if $n_v^\varphi(P,i) = y$ and contains only the node v if $n_v^\varphi(P,i) = v$, i.e., the path has length zero in this case. On the other hand, the i-th lower path of $F^\varphi(v)$ in P contains for $n_v^\varphi(P,i) = y$ no node and for $n_v^\varphi(P,i) = v$ it contains only the node y.

Properties of the filters In the following, we consider properties of the filters.

Observation 6. *Let $1 \leq i \leq 2n$, $r := r(i)$, $1 \leq j \leq 2m_r$ and $k := r(j)$. Then node $\phi_{j,1}^i(v)$ is according to lines 12–16 of Algorithm 3.5 adjacent to the unique external informer w as presented in Table 3.2—see also Figure 3.12.*

Lemma 3.5.15. *The following conditions are satisfied:*

i) $deg_{G^\varphi}(v) = deg_G(v) + 1$ *for all* $v \in D(\varphi)$

ii) $deg_{G^\varphi}(v) = deg_G(v)$ *for all* $v \in V \setminus D(\varphi)$

iii) $deg_{G^\varphi}(v) \leq 4$ *for all* $v \in V^\varphi \setminus V$.

Proof. The graph G^φ arises from G by consecutive calls of the functions $G^i(\cdot)$ for $1 \leq i \leq 5$. Due to Observation 5 only the functions $G^1(\cdot)$ and $G^4(\cdot)$ increase the degree of a node with respect to its degree in the input graph.

i) There is exactly one call of $G^1(\cdot, v)$ in the five parts of enf_5. This call is in line 4 of Algorithm 3.1. In no call of $G^4(\cdot)$ in any of the five parts node v is an input—all calls of this function are in the lines 13 and 15 and 20 of Algorithm 3.5. The calls of $G^2(\cdot)$ and $G^3(\cdot)$ in which v is an input do not increase the degree of v due to Observation 3.5.8 (ii).

ii) None of the calls of the functions $G^i(\cdot)$ for $1 \leq i \leq 5$ in the five parts has a node $v \in V \setminus D(\varphi)$ as input. Thus, the claim follows from Observation 3.5.8 (iv).

iii) The only nodes added by the functions $G^i(\cdot)$ for $1 \leq i \leq 5$ that are itself input for a subsequent call of a function $G^j(\cdot)$ for $j \in \{1, 4\}$ are the informers and the constants. None of them is input of a call of $G^1(\cdot)$. The constants are only input for the calls of $G^4(\cdot)$ in line 20 of Algorithm 3.5. Each constant is input in exactly one call of $G^4(\cdot)$—in fact, after a constant is added in line 18 it is an input of the subsequent call of $G^4(\cdot)$ in line 20 and only of this call. For each pair of anterior external informers added in line 9 of Algorithm 3.4 there is exactly one call of $G^4(\cdot)$ in line 13 of Algorithm 3.5 in which one of the two anterior external informers is an input. Moreover, for each pair of posterior external informers added in line 10 of Algorithm 3.4 there is exactly one call of $G^4(\cdot)$ in line 15 of Algorithm 3.5 in which one of the two posterior external informers is an input. Finally, for each pair of internal informers added in line 10 of Algorithm 3.3 there is exactly one call of $G^4(\cdot)$ in lines 13 and 15 of Algorithm 3.5 in which one of the internal informers is an input. Thus, the informers have a degree of at most three and the constants a degree of at most two in G^φ. □

i	odd				even			
j	odd		even		odd		even	
$pos(l_{r,k})$	0	1	0	1	0	1	0	1
w	$\delta^i_{k,1}(v)$	$\delta^i_{k,2}(v)$	$\eta^i_{k,2}(v)$	$\eta^i_{k,1}(v)$	$\delta^i_{k,2}(v)$	$\delta^i_{k,1}(v)$	$\eta^i_{k,1}(v)$	$\eta^i_{k,2}(v)$

Table 3.2: Node $\phi^i_{j,1}(v)$ is adjacent to external informer w.

Lemma 3.5.16. *Let $P \in \mathscr{P}(V^\varphi)$ and $\kappa := c_P(v)$. Then the following conditions hold:*

i) Let $w \in D(\varphi)$ and $l^w_{r,j} \in \varphi(w)$ for $1 \leq r \leq n^w$, $1 \leq j \leq m^w_r$ be a literal for which $var(l^w_{r,j}) = v$. Then the following conditions are satisfied:

a) Suppose that $T^\varphi(v)$ is flat in P. Then $c_P(\delta^{2r-1}_{j,1}(w)) = c_P(\delta^{2r}_{j,1}(w)) = \kappa$ and $c_P(\delta^{2r-1}_{j,2}(w)) = c_P(\delta^{2r}_{j,2}(w)) = \overline{\kappa}$.

b) Suppose that $B^\varphi(v)$ is flat in P. Then $c_P(\eta^{2r-1}_{j,1}(w)) = c_P(\eta^{2r}_{j,1}(w)) = \overline{\kappa}$ and $c_P(\eta^{2r-1}_{j,2}(w)) = c_P(\eta^{2r}_{j,2}(w)) = \kappa$.

ii) Suppose that $F^\varphi(v)$ is awaiting in P. Then the following equations hold:

a) $c_P(\beta^i_5(v)) = \overline{\kappa}$ and $c_P(\beta^i_3(v)) = c_P(\beta^i_4(v)) = \kappa$ for all $1 \leq i \leq 2n^v$.

b) $c_P(\gamma^i_{j,1}(v)) = \kappa$ and $c_P(\gamma^i_{j,2}(v)) = \overline{\kappa}$ for all $1 \leq i \leq 2n^v, 1 \leq j \leq 2m^v_p + 1$ where $p := pr^+_{2n^v}(i)$.

c) $c_P(\phi^i_{j,1}(v)) = \kappa$ and $c_P(\phi^i_{j,2}(v)) = \overline{\kappa}$ for all $1 \leq i \leq 2n^v$, $1 \leq j \leq 2m^v_i + 1$.

d) $c_P(c^{2r-\kappa}_1(v)) = c_P(c^{2r-\overline{\kappa}}_2(v)) = \overline{\kappa}$ and $c_P(c^{2r-\overline{\kappa}}_1(v)) = c_P(c^{2r-\kappa}_2(v)) = \kappa$ for all $1 \leq r \leq n^v$.

Proof. i) a) The set $T^\varphi(v)$ consists of the anterior external informers of the nodes $z \in D(\varphi)$ for which $z \in Nodes(\varphi(v))$ and are added in line 9 of Algorithm 3.4. Recall that the nodes of $T^\varphi(v)$ and the edges between them form a path in $F^\varphi(v)$, i.e., the throat of v. The nodes $\delta^{2r-1}_{j,1}(w)$ and $\delta^{2r}_{j,1}(w)$ have even distance and $\delta^{2r-1}_{j,2}(w)$ and $\delta^{2r}_{j,2}(w)$ odd distance from v along edges of the throat. All edges of the throat are in the cut in P due to the emptiness of $T^\varphi(v)$. Thus, the claim follows.

b) The set $B^\varphi(v)$ is made up by the posterior external informers of the nodes $z \in D(\varphi)$ for which $z \in Nodes(\varphi(v))$ and are added in line 10 of Algorithm 3.4. The nodes of $B^\varphi(v)$ and the edges between them form a path in $F^\varphi(v)$, i.e., the braid of v. The nodes $\eta^{2r-1}_{j,1}(w)$ and $\eta^{2r}_{j,1}(w)$ have odd distance and $\eta^{2r-1}_{j,2}(w)$ and $\eta^{2r}_{j,2}(w)$ even distance from v along edges of the braid. All edges of the braid are in the cut in P due to the emptiness of $B^\varphi(v)$. Thus, the claim follows.

ii) a) Since $F^\varphi(v)$ is awaiting, $T^\varphi(v)$ is flat. Due to (i)(a) and since the unique edge incident to $\beta^1_5(v)$ connecting $\beta^1_5(v)$ with a node of $T^\varphi(v)$ is in the cut in awaiting partitions, we have $c_P(\beta^1_5(v)) = \overline{\kappa}$. Moreover, the edges $\{\beta^i_3(v), \beta^i_5(v)\}$ and $\{\beta^i_4(v), \beta^i_5(v)\}$ are in the cut in P for all $1 \leq i \leq 2n^v$ which implies $c_P(\beta^i_4(v)) = c_P(\beta^i_3(v)) \neq c_P(\beta^i_5(v))$ for all $1 \leq i \leq 2n^v$. Finally, since node $\beta^{i+1}_5(v)$ is adjacent to $\beta^i_3(v)$ for all $1 \leq i < 2n^v$ and the edge between them is in the cut since $F^\varphi(v)$ is awaiting in P, we get $c_P(\beta^{i+1}_5(v)) \neq c_P(\beta^i_3(v))$.

b) The internal informers of v are added in line 10 of Algorithm 3.3. As the subgraphs induced by the nodes of $T^{\varphi}(v)$, the internal informers are added via the function $G^2(\cdot)$ and form the paths $p_i^{\varphi}(v)$ for $1 \leq i \leq 2n^v$ in $F^{\varphi}(v)$. Let $1 \leq i \leq 2n^v$ and $1 \leq j \leq 2m^v_{p(i)} + 1$. Then, according to the definition of the function $G^2(\cdot)$, node $\gamma^i_{j,1}(v)$ has even and node $\gamma^i_{j,2}(v)$ odd distance from $\beta^i_4(v)$ along edges of the path $p_i^{\varphi}(v)$. Since all edges on the path $p_i^{\varphi}(v)$ are in the cut in awaiting partitions and $c_P(\beta^i_4(v)) = \kappa$ due to (ii)(a), the claim follows.

c) Analogously to (ii)(b) it follows that the delayers added in lines 13 and 15 and 20 of Algorithm 3.5 form paths in $F^{\varphi}(v)$. Let $1 \leq i \leq 2n^v$ and $1 \leq j \leq 2m^v_i + 1$. Then node $\phi^i_{j,1}(v)$ has even and node $\phi^i_{j,2}(v)$ odd distance from $\gamma^i_{1,1}(v)$ along edges of its corresponding path. Since all edges on the path are in the cut in awaiting partitions and $c_P(\gamma^i_{1,1}(v)) = \kappa$ due to (ii)(b), the claim follows.

d) Since $F^{\varphi}(v)$ is awaiting in P, it is also straight. Therefore, $c_P(c_1^{2r}(v)) = c_P(c_2^{2r-1}(v)) = 1$ and $c_P(c_1^{2r-1}(v)) = c_P(c_2^{2r}(v)) = 0$ for all $1 \leq r \leq n^v$. □

Lemma 3.5.17. *Let $1 \leq r \leq n^v$, $1 \leq j \leq m^v_r$, $w := nod(l^v_{r,j})$, $P \in \mathscr{P}(V^{\varphi})$ such that $F^{\varphi}(v)$ is awaiting in P and $\kappa := c_P(v)$. Then the following conditions are satisfied:*

i) Suppose that $T^{\varphi}(w)$ is flat in P. Then the edge between $\phi^{2r-\overline{\kappa}}_{2j-1,1}(v)$ and the unique node of $T^{\varphi}(w)$ it is adjacent to is in the cut if and only if $val_P(l^v_{r,j}) = false$.

ii) Suppose that $B^{\varphi}(w)$ is flat in P. Then the edge between $\phi^{2r-\overline{\kappa}}_{2j,1}(v)$ and the unique node of $B^{\varphi}(w)$ it is adjacent to is in the cut if and only if $val_P(l^v_{r,j}) = false$.

Proof. Let $\pi := 2r - \overline{\kappa}$, $\sigma := 2j$ and $\tau := c_P(w)$. Since $F^{\varphi}(v)$ is awaiting in P, we have $c_P(\phi^{\pi}_{\sigma-1,1}(v)) = c_P(\phi^{\pi}_{\sigma,1}(v)) = \kappa$ due to Lemma 3.5.16 (ii)(c).

i) The emptiness of $T^{\varphi}(w)$ in P implies $c_P(\delta^{\pi}_{j,1}(v)) = \tau$ and $c_P(\delta^{\pi}_{j,2}(v)) = \overline{\tau}$ due to Lemma 3.5.16 (i)(a). Since $\sigma - 1$ is odd and π is odd if and only if $\kappa = 0$, it follows by Observation 6 that $\phi^{\pi}_{\sigma-1,1}(v)$ is adjacent to $\delta^{\pi}_{j,1}(v)$ if $\kappa = pos(l^v_{r,j})$ and to $\delta^{\pi}_{j,2}(v)$ if $\kappa \neq pos(l^v_{r,j})$.

Assume first that $val_P(l^v_{r,j}) = false$. Then $\tau = 0$ if $pos(l^v_{r,j}) = 1$ and $\tau = 1$ otherwise, i.e., $pos(l^v_{r,j}) \neq \tau$. Thus, if $\phi^{\pi}_{\sigma-1,1}$ is adjacent to $\delta^{\pi}_{j,1}(v)$ then $\kappa = pos(l^v_{r,j}) \neq \tau = c_P(\delta^{\pi}_{j,1}(v))$ and therefore $c_P(\delta^{\pi}_{j,1}(v)) = \overline{\kappa}$ and if it is adjacent to $\delta^{\pi}_{j,2}(v)$ then $\kappa \neq pos(l^v_{r,j}) \neq \tau \neq c_P(\delta^{\pi}_{j,2}(v))$ which implies $c_P(\delta^{\pi}_{j,2}(v)) = \overline{\kappa}$.

Now assume that $val_P(l^v_{r,j}) = true$. Then $\tau = 0$ if $pos(l^v_{r,j}) = 0$ and $\tau = 1$ otherwise, i.e., $pos(l^v_{r,j}) = \tau$. Hence, if $\phi^{\pi}_{\sigma-1,1}$ is adjacent to $\delta^{\pi}_{j,1}(v)$ then $\kappa = pos(l^v_{r,j}) = \tau = c_P(\delta^{\pi}_{j,1}(v))$ and therefore $c_P(\delta^{\pi}_{j,1}(v)) = \kappa$ and if it is adjacent to $\delta^{\pi}_{j,2}(v)$ then $\kappa \neq pos(l^v_{r,j}) = \tau \neq c_P(\delta^{\pi}_{j,2}(v))$ which implies $c_P(\delta^{\pi}_{j,2}(v)) = \kappa$.

ii) The emptiness of $B^{\varphi}(w)$ in P implies $c_P(\eta^{\pi}_{j,1}(v)) = \overline{\tau}$ and $c_P(\eta^{\pi}_{j,2}(v)) = \tau$ due to Lemma 3.5.16 (i)(b). Since σ is even and π is odd if and only if $\kappa = 0$, it follows by Observation 6 that $\phi^{\pi}_{\sigma,1}(v)$ is adjacent to $\eta^{\pi}_{j,1}(v)$ if $\kappa \neq pos(l^v_{r,j})$ and to $\eta^{\pi}_{j,2}(v)$ if $\kappa = pos(l^v_{r,j})$.

Assume first that $val_P(l^v_{r,j}) = false$. Then $\tau = 0$ if $pos(l^v_{r,j}) = 1$ and $\tau = 1$ otherwise, i.e., $pos(l^v_{r,j}) \neq \tau$. Thus, if $\phi^{\pi}_{\sigma,1}$ is adjacent to $\eta^{\pi}_{j,2}(v)$ then $\kappa = pos(l^v_{r,j}) \neq \tau = c_P(\eta^{\pi}_{j,2}(v))$ and therefore $c_P(\eta^{\pi}_{j,2}(v)) = \overline{\kappa}$ and if it is adjacent to $\eta^{\pi}_{j,1}(v)$ then $\kappa \neq pos(l^v_{r,j}) \neq \tau \neq c_P(\eta^{\pi}_{j,1}(v))$ which implies $c_P(\eta^{\pi}_{j,1}(v)) = \overline{\kappa}$.

Now assume that $val_P(l^v_{r,j}) = true$. Then $\tau = 0$ if $pos(l^v_{r,j}) = 0$ and $\tau = 1$ otherwise, i.e., $pos(l^v_{r,j}) = \tau$. Hence, if $\phi^{\pi}_{\sigma,1}$ is adjacent to $\eta^{\pi}_{j,2}(v)$ then $\kappa = pos(l^v_{r,j}) = \tau = c_P(\eta^{\pi}_{j,2}(v))$ and therefore $c_P(\eta^{\pi}_{j,2}(v)) = \kappa$ and if it is adjacent to $\eta^{\pi}_{j,1}(v)$ then $\kappa \neq pos(l^v_{r,j}) = \tau \neq c_P(\eta^{\pi}_{j,1}(v))$ which implies $c_P(\eta^{\pi}_{j,1}(v)) = \kappa$.

Lemma 3.5.18. *Let $P_0 \in \mathscr{P}(V^{\varphi})$ such that $F^{\varphi}(v)$ is canonical in P_0, $w \in R^{\varphi}(v)$ and $s := (w_1, \ldots, w_q)$ for $q \in \mathbb{N}$, $w_i \in V^{\varphi}$ for $1 \leq i \leq q$ be a final sequence starting at (G, P_0^{φ}). Then the following conditions are satisfied:*

i) Node w is unhappy in P_0 if and only if there is an index $1 \leq i \leq 2n^v + 1$ such that $w = n^{\varphi}_v(P_0, i)$.

ii) Suppose that $w = w_1$. Then $F^{\varphi}(v)$ is canonical in P_1 and the following properties are satisfied for all $1 \leq i \leq 2n^v + 1$ for which w is a node of the cycle $cy^{\varphi}_v(i)$:

a) If $w \neq v$ then $d^{\varphi}_v(P_1) < d^{\varphi}_v(P_0)$.

b) If $w = v$ then $d^{\varphi}_v(P_1) > d^{\varphi}_v(P_0)$.

iii) Suppose that w is unhappy in P_0. Then there is an index $1 \leq i \leq q$ such that $w = w_i$.

Proof. The following cases for $w \in R^{\varphi}(v)$ are possible—see Figure 3.11 and Figure 3.12:

1) $w = \alpha_2(v)$.

2) $w = \beta^i_j(v)$ for any $1 \leq i \leq 2n^v$, $1 \leq j \leq 4$.

3) $w = \gamma^i_{j,k}(v)$ for any i, j, k.

4) $w = \delta^i_{j,k}(z)$ for any i, j, k and $z \in D(\varphi)$ such that $nod(l^z_{i,j}) = v$.

5) $w = \eta^i_{j,k}(z)$ for any i, j, k and $z \in D(\varphi)$ such that $nod(l^z_{i,j}) = v$.

6) $w = v$.

In each of the above cases node w is of Type I. Let $1 \leq i \leq 2n^v + 1$ be such that w is a node of the cycle $cy^{\varphi}_v(i)$ and $x \in B^{\varphi}(v)$ be the unique node for which $\{x, v\} \in E^{\varphi}$.

i) Since w is of Type I, it is according to Observation 2 unhappy if and only if its heaviest edge is not in the cut. In the following, we show that its heaviest edge $\{H_{G^\varphi}(w), w\}$ is not in the cut in P_0 if and only if there is an index $1 \leq i \leq 2n^v + 1$ such that $w = n_v^\varphi(P_0, i)$ which implies the claim.

In the cases (1)–(4) and (6)—for an orientation, see Figure 3.11 and Figure 3.12—we have $dis_v^\varphi(H_{G^\varphi}(w), i) > dis_v^\varphi(w, i)$ and therefore $w = n_v^\varphi(P_0, i)$ by definition of the function $n_v^\varphi(\cdot)$ if and only if $\{H_{G^\varphi}(w), w\}$ is not in the cut. Now we show the claim for case (5). If $w = x$ then, also by definition of $n_v^\varphi(\cdot)$, we have $w = n_v^\varphi(P_0, i)$ if and only if $\{H_{G^\varphi}(w), w\}$ is not in the cut. And if $w \in B^\varphi(v) \setminus \{x\}$ then we again have $dis_v^\varphi(H_{G^\varphi}(w), i) > dis_v^\varphi(w, i)$ and therefore $w = n_v^\varphi(P_0, i)$ if and only if $\{H_{G^\varphi}(w), w\}$ is not in the cut.

ii) Since w is unhappy in P_0, we have $w = n_v^\varphi(P_0, i)$ for some $1 \leq i \leq 2n^v+1$ according to (i). We prove the claim by considering the possible cases for $w \in R^\varphi(v)$:

- Cases (1), (3), (4), $w = \beta_j^i(v)$ for $1 \leq i \leq 2n^v$, $j \in \{1, 3, 4\}$ or $w = \eta_{j,k}^i(z)$ for any i, j, k and $z \in D(\varphi)$ such that $nod(l_{i,j}^z) = v$ but $w \neq x$:
  Since w is of Type I and unhappy in P_0, its heaviest edge e_1 is not in the cut in P_0. Since $F^\varphi(v)$ is canonical in P_0, the lighter edge e_2 of the two heavy edges incident to w is in the cut in P_0. Therefore, after the flip of w, the edge e_1 is in the cut and e_2 not. Both e_1 and e_2 are edges of $sp_v^\varphi(i)$ which implies that $F^\varphi(v)$ is canonical in P_1. Since the node adjacent to w via e_2 has a lower distance from v along edges of $sp_v^\varphi(i)$, we get $d_v^\varphi(P_1) < d_v^\varphi(P_0)$.

- Case $w = \beta_2^i(v)$ for any $1 \leq i \leq 2n^v$:
  Since w is of Type I and unhappy in P_0, its heaviest edge e_1 is not in the cut in P_0. Since $F^\varphi(v)$ is canonical in P_0, the two non-heaviest edges e_2, e_3 incident to w are in the cut in P_0. Thus, after the flip of w, edge e_1 is in the cut and e_2 and e_3 are not in the cut. The edges e_2 and e_3 are edges of $sp_v^\varphi(i)$ and $sp_v^\varphi(i+1)$. However, there is no path $sp_v^\varphi(j)$ for $1 \leq j \leq 2n^v + 1$ such that both e_2 and e_3 are edges of $sp_v^\varphi(j)$. Thus, $F^\varphi(v)$ is canonical in P_1. Moreover, since the nodes adjacent to w via e_2 and e_3 have a lower distance to v with respect to their corresponding path $sp_v^\varphi(j)$ for $1 \leq j \leq 2n^v + 1$ than w, we get $d_v^\varphi(P_1) < d_v^\varphi(P_0)$.

- Case $w = x$:
  Since w is unhappy in P_0, the heaviest edge $e_1 = \{w, v\}$ of w is not in the cut in P_0. Since $F^\varphi(v)$ is canonical in P_0, the unique edge $e_2 = \{w, w'\}$ for $w' \in B^\varphi(w) \setminus \{v\}$ is in the cut in P_0. Thus, after the flip of w, edge e_1 is in the cut and e_2 is not. The edges e_1 and e_2 are both edges of $sp_v^\varphi(i)$ for $1 \leq i \leq 2n^v + 1$. Thus, $F^\varphi(v)$ is canonical in P_1. Moreover, w' has a lower distance from v along the path $sp_v^\varphi(i)$ than w which implies $d_v^\varphi(P_1) < d_v^\varphi(P_0)$.

- Case (6), i.e., $w = v$:
  Since w is unhappy in P_0, the heaviest edge e_1 of w is not in the cut. Since $F^\varphi(v)$ is canonical in P_0, the edge $e_2 = (w, x)$ is in the cut. Thus, after the

flip of w, edge e_1 is in the cut and e_2 is not. The edges e_1 and e_2 are edges of $sp_v^\varphi(i)$. Thus, $F^\varphi(v)$ is canonical in P_1. Finally, the distance of x from v along edges of $sp_v^\varphi(i)$ is strictly positive which implies $d_v^\varphi(P_1) > d_v^\varphi(P_0)$ due to $d_v^\varphi(P_0) = 0$.

iii) Let $1 \leq i \leq 2n^\nu + 1$ be an index such that $w = n_v^\varphi(P,i)$—from (i) we know that there is such an index. Since $F^\varphi(v)$ is canonical in P_0, all edges of the cycle $cy_v^\varphi(i)$ unequal to the heaviest edge incident to w are in the cut in P_0. The node $H_{G^\varphi}(v)$ is also of Type I—see Figure 3.11 and Figure 3.12—and w has no influence on it. Thus, $H_{G^\varphi}(v)$ is happy in P_0. For the remaining nodes of the cycle $cy_v^\varphi(i)$ Observation 3.5.8 (vi) implies that they are happy in P_0. Moreover, the nodes of the said cycle unequal to w remain happy as long as no node of the cycle flips. Then, since s is final, there is a flip of w in s. □

Lemma 3.5.19. *Let $Q_0 \in \mathscr{P}(V^\varphi)$ such that $F^\varphi(v)$ is canonical in Q_0 and let $s = (w_1,\dots,w_q)$ starting at (G^φ, Q_0) for $q \in \mathbb{N}$ and $w_i \in V^\varphi$ for all $1 \leq i \leq q$ be an improving sequence of flips. Then $F^\varphi(v)$ is canonical in Q_i for all $0 \leq i \leq q$.*

Proof. Since $F^\varphi(v)$ is canonical in Q_0, it is also straight in Q_0 by definition. For every $1 \leq i \leq 2n^\nu$, node $c_1^i(v)$ and $c_2^i(v)$ are of Type I, $H_{G^\varphi}(c_1^i(v)) = c_2^i(v)$, $H_{G^\varphi}(c_2^i(v)) = c_1^i(v)$ and $c_{Q_0}(c_1^i(v)) \neq c_{Q_0}(c_2^i(v))$. Thus, node $c_1^i(v)$ and $c_2^i(v)$ for all $1 \leq i \leq 2n^\nu$ are happy in partition Q_j for all $0 \leq j \leq q$. Consequently, no constant node $c_1^i(v)$, $c_2^i(v)$ for $1 \leq i \leq 2n^\nu$, $v \in D(\varphi)$ flips in s and therefore $F^\varphi(v)$ is straight in Q_j for all $0 \leq j \leq q$.

Now we show by induction on i that $F^\varphi(v)$ is canonical in Q_i for all $0 \leq i \leq q$. As induction basis, note that it is canonical in Q_0. Now assume as induction hypothesis that $F^\varphi(v)$ is canonical in Q_i for an arbitrary $0 \leq i < q$. Let x be a node of $F^\varphi(v) \setminus \{H_G(v)\}$ such that $x \neq n_v^\varphi(Q_i, j)$ for all $1 \leq j \leq 2n^\nu + 1$. Then all heavy edges of x are in the cut in Q_i since $F^\varphi(v)$ is canonical in Q_i and therefore x is happy in Q_i due to Observation 3.5.8 (vi). Thus, for each $1 \leq i \leq q$ there is a $1 \leq k \leq 2n^\nu + 1$ such that $w_i = n_v^\varphi(Q_i, k)$.

Now we consider the possible cases for w_i and show for each of them that $F^\varphi(v)$ is canonical in Q_{i+1}. If $w_j \in R^\varphi(v)$ then Lemma 3.5.18 (ii) implies the claim for this case. Now consider the case that $w_i = \alpha_1(v)$. Then, exactly one of the two edges $\{\alpha_1(v), \alpha_2(v)\}$ and $e := \{\alpha_1(v), T_{G^\varphi}(\alpha_1(v))\}$ is in the cut in Q_i. Thus, after the flip of w_i there is still exactly one of them in the cut which implies the claim for this case. Now we consider the case that $w_i = \phi_{k,m}^j(v)$ for any j,k,m. Then exactly one of the two heavy edges incident to w_i is in the cut in Q_i. Thus, in Q_{i+1} there is still exactly one of them in the cut whereafter the claim follows for this case. Finally, we consider the case that $w_i = \beta_5^j(v)$ for an arbitrary $1 \leq j \leq 2n^\nu$—for an overview see Figure 3.11. Since w_i is of type III, it is only unhappy if at least two edges incident to it are not in the cut. If the edges $\{\beta_3^j, \beta_5^j\}$ and $\{\beta_4^j, \beta_5^j\}$ are not in the cut in Q_i then the remaining edge e incident to $\beta_5^j(v)$ is in the cut in Q_i since $F^\varphi(v)$ is canonical in Q_i. Thus, after the flip of w_i the edges $\{\beta_3^j, \beta_5^j\}$ and $\{\beta_4^j, \beta_5^j\}$ are in the cut and e is not in the cut anymore, which

implies that $F^{\varphi}(v)$ is still canonical. If e is not in the cut in Q_i then the remaining edges incident to w_i are in the cut since $F^{\varphi}(v)$ is canonical in Q_i, but then w_i is happy, which is a contradiction. Thus, this case is not possible. □

Lemma 3.5.20. *Let $w \in D(\varphi)$ with $v \neq w$, $Q_0 \in \mathscr{P}(V^{\varphi})$ such that $F^{\varphi}(x)$ is canonical in Q_0 for all $x \in D(\varphi)$ and assume $\{v\} <_{Q_0} \{w\}$. Then $\{v\} <_{Q_0} T^{\varphi}(w)$.*

Proof. Suppose for the sake of contradiction that there is a sequence $s = (w_1, \ldots, w_q)$ for $w_i \in V^{\varphi}$, $1 \leq i \leq q$, $q \in \mathbb{N}$ starting at (G^{φ}, Q_0) such that there is a node $x_1 \in T^{\varphi}(w)$ that flips prior to the first flip of v in s, i.e., there is an index $1 \leq j \leq q$ such that $x_1 = w_j$ and $w_i \neq v$ for all $1 \leq i \leq j$. Let $t := (x_1, x_2, \ldots, x_k)$ for $k \in \mathbb{N}$ where $x_k = w$ and x_i for $1 < i \leq k$ be the unique node on which x_{i-1} has influence in G^{φ}—see Figure 3.12. Since $F^{\varphi}(w)$ is canonical in Q_0 it is also canonical in Q_{j-1} due to Lemma 3.5.19. Therefore, the sequence $s_1^{j-1} \circ t$ started at (G^{φ}, Q_0) is improving which contradicts the assumption $\{v\} <_{Q_0} \{w\}$. Thus, $\{v\} <_{Q_0} T^{\varphi}(w)$. □

After characterizing basic properties of the filters, we are ready to formulate the main tool that we use in the enforcing technique. The statement of the lemma makes use of notations introduced in the first and in the second part of the definitions for the filter and the proof also makes use of the definitions of the Basic Subgraphs.

Lemma 3.5.21 (Filtering Lemma). *Let $G = (V, E)$ be a graph containing only nodes of Type I and III, $P \in \mathscr{P}(V)$ and $\varphi : V_I \to \Phi(V_I)$ be such that for each $w \in D(\varphi)$ node w is of degree at most three, w has no influence on $H_G(w)$ and w is happy in (G, P). Then one can compute in time $O(poly(|V|, |\varphi|))$ a graph $G^{\varphi} = (V^{\varphi}, E^{\varphi})$ and a partition $Q_0 \in \mathscr{P}(V^{\varphi})$ with the following properties:*

FL1) G^{φ} is of maximum degree four.

FL2) $V \subset V^{\varphi}$.

FL3) Each $w \notin D(\varphi)$ is influenced in G^{φ} by the same nodes via edges of the same weights as in G.

FL4) $Q_0|_V = P$.

FL5) For each final sequence $s = (w_1, \ldots, w_q)$ starting at (G^{φ}, Q_0) for $q \in \mathbb{N}$ and $w_j \in V^{\varphi}$ for all $1 \leq j \leq q$ and each index $0 \leq i \leq q$, for which s_1^i does not contain two flips of $H_G(w)$ for any $w \in D(\varphi)$ without an intermediate flip of w, the following properties hold for all $v \in D(\varphi)$ with $u := H_G(v)$:

- *i) If $c_{Q_i}(u) \neq c_{Q_i}(v)$ then $\{u\} <_{Q_i} \{v\}$.*
- *ii) Suppose $c_{Q_i}(u) = c_{Q_i}(v)$.*
  - *a) If*
    - *no node of $Nodes(\varphi(v))$ flips in s_1^i after the last flip of u*
    - *$val_{Q_i}(\varphi(v)) = false$*

- $\{v\} <_{Q_i} \{u\}$

then $Nodes(\varphi(v)) <_{Q_i} \{v\}$.

b) If

- $val_{Q_i}(\varphi(v)) = true$
- $\{v\} <_{Q_i} Nodes(\varphi(v)) \cup \{u\}$

then there is a flip of v *in* s^q_{i+1}.

Comment Recall that the aim of the extension of the graph G by further nodes and edges is as follows. If a node $v \in D(\varphi)$ is happy in a partition of G then it is not supposed to flip prior to the first flip of $u := H_G(v)$ in any sequence starting at the corresponding partition of the extended graph G^φ. This property is encapsulated in Lemma 3.5.21 (FL5)(i). On the other hand, for the case that v is unhappy in a partition of G, a flip of v is supposed to depend in the extended graph on the satisfaction of the formula $\varphi(v)$ with respect to the colors of the nodes of $Nodes(\varphi(v))$. The case in which $\varphi(v)$ is not satisfied is encapsulated in Lemma 3.5.21 (FL5)(ii)(a) and the opposite case in Lemma 3.5.21 (FL5)(ii)(b).

Proof. We let $G^\varphi := enf_5(G, \varphi)$—for an orientation, see Figure 3.11 and Figure 3.12. The graph G^φ is computable in time $O(poly(|V|, |\varphi|))$ for the following two reasons. First, each operation performed in $enf_5(G, \varphi)$—including the operations performed during the execution of the functions $enf_1(G, \varphi), \ldots, enf_4(G, \varphi)$ called by $enf_5(G, \varphi)$—that substitutes an edge by a subgraph, namely the functions $G^4(\cdot), \ldots, G^1(\cdot)$, require constant time—see Definition 3.5.1–Definition 3.5.5. Second, the number of passes of each for-loop of $enf_5(G, \varphi), \ldots, enf_1(G, \varphi)$ can be upper bounded linearly in either $|V|$ or $|\varphi|$.

Since each node $v \in D(\varphi)$ has by assumption a degree of at most three, the property (FL1) follows from Lemma 3.5.15. The function $enf_5(\cdot)$ substitutes no nodes which implies (FL2). Moreover, it only substitutes the heaviest edges of the nodes of $D(\varphi)$. By assumption, no node $v \in D(\varphi)$ has influence on $H_G(v)$ in G. Therefore, we get (FL3).

The partition Q_0 is computed from P in the following way. We choose $c_{Q_0}(w) = c_P(w)$ for all $w \in V$. This implies (FL4). The colors of the remaining nodes are chosen such that for all $v \in D(\varphi)$ all heavy edges of $F^\varphi(v)$ but one arbitrary edge of the braid of v are in the cut in Q_0. In the following, we prove (FL5).

FL5i) Since $F^\varphi(v)$ is canonical in Q_0, it is also canonical in Q_i due to Lemma 3.5.19. Moreover, due to $c_{Q_i}(u) \neq c_{Q_i}(v)$ it follows that $F^\varphi(v)$ is enterable in Q_i. Consequently, the following properties are satisfied:

E1) The heaviest edge of v is in the cut in Q_0.

E2) The edges $\{u, \alpha_1(v)\}$ and $\{\alpha_1(v), \alpha_2(v)\}$ are in the cut in Q_0.

E3) All heavy edges of the nodes of $H^\varphi(v) \setminus \{u, v, \alpha_1(v)\}$ are in the cut in Q_0.

Since v is of Type I, property (E1) implies that node v is happy in Q_0. Property (E2) implies that $\alpha_1(v)$ is happy due to Observation 3.5.8 (vi). Finally, (E3) implies the happiness for the remaining nodes of $H^{\varphi}(v) \setminus \{u\}$ due to, again, Observation 3.5.8 (vi). Thus, no node of $H^{\varphi}(v) \setminus \{u\}$, in particular node v, flips in s^q_{i+1} prior to the first flip of u.

FL5ii) Let $\kappa := c_{Q_0}(v)$. Since v is happy in (G,P) by assumption, (FL4) implies $c_{Q_0}(u) = \overline{\kappa}$. Then, (FL5)(i) implies that there is a flip of u in s^i_1. Let $1 \leq k \leq i$ be the greatest index for which $w_k = u$.

a) Suppose that no node of $Nodes(\varphi(v))$ flips in s^i_{k+1}, $val_{Q_i}(\varphi(v)) = \mathit{false}$ and $\{v\} <_{Q_i} \{u\}$. Since $F^{\varphi}(v)$ is canonical in Q_0 by assumption and s is improving, Lemma 3.5.19 implies that $F^{\varphi}(v)$ is canonical in Q_k. Then $w_k = u$ implies that $F^{\varphi}(v)$ is just entered in Q_k whereafter we get the following properties for all $1 \leq r \leq n^v$, $1 \leq j \leq 2m^v_r + 1$—for an overview of the corresponding nodes see Figure 3.12:

JE1) $c_{Q_k}(\phi^{2r-1}_{j,1}(v)) = c_{Q_k}(\phi^{2r}_{j,1}(v)) = \kappa$ and $c_{Q_k}(\phi^{2r-1}_{j,2}(v)) = c_{Q_k}(\phi^{2r}_{j,2}(v)) = \overline{\kappa}$ due to Lemma 3.5.16 (ii)(c).

JE2) $c_{Q_k}(\gamma^{p^-_{2n^v}(2r-1)}_{j,2}(v)) = c_{Q_k}(\gamma^{p^-_{2n^v}(2r)}_{j,2}(v)) = \overline{\kappa}$ due to Lemma 3.5.16 (ii)(b).

Moreover, for all $1 \leq r \leq n^v$ we have

JE3) $c_{Q_k}(c^{2r-\kappa}_1(v)) = \overline{\kappa}$ due to Lemma 3.5.16 (ii)(d).

In the rest of the proof, we make use of a set $M \subset F^{\varphi}(v)$ which is determined as follows. For each $1 \leq j \leq 2n^v$ we name a path whose nodes are in M. The path is a subpath of $sp^{\varphi}_v(j)$ beginning at some node $z_j := \phi^j_{i(j),1}(v)$ for an index $1 \leq i(j) \leq 2m^v_{r(j)}$ that is specified later and ends at v. We let $M^- := M \setminus \bigcup_j \{z_j\}$. The purpose of M is to show that $Nodes(\varphi(v)) <_{Q_k} M$ which directly implies $Nodes(\varphi(v)) <_{Q_k} \{v\}$ due to $v \in M$. Then, since by assumption no node of $Nodes(\varphi(v))$ flips in s^i_{k+1}, we also get $Nodes(\varphi(v)) <_{Q_i} \{v\}$ as postulated by the theorem.

To show $Nodes(\varphi(v)) <_{Q_k} M$ we first prove some properties of the nodes in M. In Q_k all heavy edges between the nodes of M are in the cut since $F^{\varphi}(v)$ is just entered in Q_k. Thus, all nodes of M^- are happy in Q_k according to Observation 3.5.8 (vi) and remain happy as long as no node of M flips. Node z_j for any j is of Type II and is therefore happy according to Observation 2 if the three non-heaviest edges incident to z_j are in the cut. In Q_k two of these three edges are in the cut for each z_j: the heavy edge $\{z_j, \phi^j_{i(j),2}(v)\}$ according to (JE1) and the edge between z_j and the internal informer adjacent to z_j according to (JE1) and (JE2).

Now we specify the indices $i(j)$ and consider the happiness of the corresponding nodes z_j. The specification and consideration of the happiness is divided into two parts.

In the first part, we specify $i(j)$ for the indices $1 \leq j \leq 2n^v$ for which $j = 2r(j) - \kappa$. For these j, we choose $i(j) := 2m^v_{r(j)} + 1$. Then z_j is adjacent to the constant c^j_1—see line 20 of Algorithm 3.5—and the edge between them is in the cut due to (JE1) and (JE3). Thus, all non-heaviest edges incident to z_j are in the cut in Q_k which implies that it is happy. The constant c^j_1 does not flip in s since $F^\varphi(v)$ is canonical and therefore straight in Q_b for all $0 \leq b \leq q$. The other two nodes adjacent to z_j via non-heaviest edges incident to z_i are in M^- and remain therefore happy as long as no node of M flips. Thus, z_j also remains happy as long as no node of M flips. Consequently, if a node of M flips in s^q_{k+1} then the first node of M that flips in s^q_{k+1} is a node z_j for $j \not\equiv \kappa \mod 2$.

In the second part, we consider the remaining indices $1 \leq j \leq 2n^v$. These are the indices for which $j = 2r(j) - \overline{\kappa}$. Let $1 \leq t \leq m^v_{r(j)}$ be such that $val_{Q_k}(l^v_{r(j),t}) = \mathit{false}$—such a literal exists in $M^v_{r(j)}(v)$ since, by assumption, $val_{Q_i}(\varphi(v)) = \mathit{false}$ and no node of $Nodes(\varphi(v))$ flips in s^i_{k+1}—and let $w := nod(l^v_{r(j),t})$. We choose $i(j) := 2t$. Let $1 \leq j \leq 2n^v$ with $j = 2r(j) - \overline{\kappa}$ and assume that z_j is the first node of M that flips in s^q_{k+1}. In the following, we show that prior to the first flip of z_j in s^q_{k+1} there is a flip of w. Then, it follows, as required, that $Nodes(\varphi(v)) <_{Q_k} M$.

Since $F^\varphi(v)$ is just entered in Q_k the edge between z_j and the node $y'_j := \phi^j_{i(j)-1,2}(v)$ adjacent to it via the heaviest edge incident to z_j is in the cut, and the edge between y'_j and the node $y_j := \phi^j_{i(j)-1,1}(v)$ adjacent to y'_j via the heaviest edge incident to y'_j is also in the cut in Q_k. Node y'_j is of Type I. Thus, both z_j and y'_j are happy in s^q_{k+1} as long as y_j does not flip. Moreover, it follows that both y_j and y'_j flip before z_j becomes unhappy in s^q_{k+1}.

Now we show that between the flip of y_j and the flip of z_j there is a flip of w. Node y_j is of Type II and is therefore happy if the three non-heaviest edges incident to y_j are in the cut. Due to (JE1) and (JE2) the edge $\{y_j, y'_j\}$ is in the cut in Q_k and the edge between y_j and the internal informer adjacent to y_j is also in the cut. A further node adjacent y_j to via a non-heaviest edge is an anterior external informer $p_1 \in T^\varphi(w)$. Since $F^\varphi(w)$ is canonical in Q_0, it is also canonical in Q_b for all $0 \leq b \leq q$ due to Lemma 3.5.19. Thus, on the path from p_1 to w via the edges of the throat of w there is at most one edge not in the cut in Q_b for all $0 \leq b \leq q$. Let $k+1 < d \leq q$ be such that $w_d = y_j$ and $w_{d'} \neq y_j$ for all $k+1 < d' < d$. Assume for the sake of contradiction that p_1 has in Q_{d-1} the same color as in the partition Q' that arises from Q_{d-1} by choosing the colors of the nodes of $T^\varphi(w) \setminus \{w\}$ such that $T^\varphi(w)$ is flat. Then the edge $\{y_j, p_1\}$ is in the cut in Q' according to Lemma 3.5.17 (i). But then y_j is happy in Q' since the three non-heaviest edges incident to it are in the cut, which is a contradiction. Thus, p_1 has in Q_{d-1} the opposite color as in Q'.

Consequently, $T^{\varphi}(w)$ is not flat in Q_{d-1}. But then $B^{\varphi}(w)$ is flat in Q_{d-1}, which implies that the edge between z_j and the posterior external informer p_2 adjacent to z_j is in the cut in Q_{d-1} due to Lemma 3.5.17 (ii). Therefore, the three non-heaviest edges incident to z_j are in the cut in Q_{d-1}. Since the two non-heaviest edges incident to z_j unequal to $\{z_j, p_2\}$ are in the cut as long as no node of M flips, it follows that $B^{\varphi}(w)$ is not flat when z_j flips. By definition of the potential function $d_w^{\varphi}(\cdot)$, a canonical partition in which $B^{\varphi}(w)$ is not flat has a higher potential than a canonical partition in which $T^{\varphi}(w)$ is not flat. Then, since $T^{\varphi}(w)$ is not flat in Q_{j-1}, it follows by Lemma 3.5.18 (ii) that there is a flip of w between the flip of y_j and the flip of z_j.

b) Suppose that $val_{Q_i}(\varphi(v)) = \mathit{true}$ and $\{v\} <_{Q_i} Nodes(\varphi(v)) \cup \{u\}$. Let λ be the greatest index for $i \leq \lambda \leq q$ such that $w_j \neq v$ for all $i < j \leq \lambda$, π be an index for which $i \leq \pi \leq \lambda$ and $1 \leq r \leq n^v$ be such that for monomial M_r of $\varphi(v)$ we have $val_{Q_i}(M_r) = \mathit{true}$—the formula $\varphi(v)$ contains such a monomial since $val_{Q_i}(\varphi(v)) = \mathit{true}$—and let $\rho := 2r - \overline{\kappa}$.

We first show for any $1 \leq \sigma \leq 2n^v + 1$ a property of the σ-th upper path and one of the σ-th lower path. Since $F^{\varphi}(v)$ is canonical in Q_π due to Lemma 3.5.19, the nodes of the σ-th upper path, in particular node $n_v^{\varphi}(Q_\pi, \sigma)$, have their unnatural colors with respect to Q_π. Moreover, all edges of the σ-th upper path are in the cut in Q_π. On the other hand, all nodes of the σ-th *lower* path have their natural values with respect to Q_π and all edges of the σ-th lower path are in the cut in Q_π.

In the following, we distinguish several cases for $w \in F^{\varphi}(v) \setminus \{u, v\}$. For all but the last case, we consider individually the consequences of each of the following two assumptions:

A1) $w = n_v^{\varphi}(Q_\pi, \sigma)$ and $\{w\} <_{Q_\pi} \{t_v^{\varphi}(Q_\pi, \sigma)\}$ for all $1 \leq \sigma \leq 2n^v + 1$ for which w is a node of $cy_v^{\varphi}(\sigma)$.

A2) w is a node of the σ-th lower path with respect to Q_π for all $1 \leq \sigma \leq 2n^v + 1$ for which w is a node of $cy_v^{\varphi}(\sigma)$.

The above assumptions have the following direct implications that we use throughout the consideration of the cases:

T1) If (A1) is satisfied then $\{w\} <_{Q_\pi} \{v\}$, since all edges of the σ-th upper path with respect to Q_π are in the cut in Q_π.

T2) If (A2) is satisfied then w has its natural color with respect to Q_π.

With the help of these two observations, we show for all but the last of the considered cases for $w \in F^{\varphi}(v) \setminus \{u, v\}$ that the following two implications hold:

F1) *Suppose that* (A1) *is satisfied. Then there is a flip of* w *in* $s_{\pi+1}^{\lambda}$ *to its natural color with respect to* Q_π.

F2) *Suppose that* (A2) *is satisfied. Then* $\{v\} <_{Q_\pi} \{w\}$.

The cases for w are the following:

C1) Case $w = \alpha_1(v)$:
(F1): Due to $w = n_v^\varphi(Q_\pi, \sigma)$ the edge $e_v^\varphi(Q_\pi, \sigma) = \{w, t_v^\varphi(Q_\pi, \sigma)\} = \{w, T_{G^\varphi}(w)\}$ is not in the cut in Q_π. By assumption, node v does not flip in s_{i+1}^π. Moreover, due to the assumption $\{v\} <_{Q_i} \{u\}$, node u does also not flip in s_{i+1}^π. Since $c_{Q_i}(u) = c_{Q_i}(v)$ and $w = n_v^\varphi(Q_\pi, \sigma)$ has its unnatural color in Q_π, the edge $\{u, w\}$ is not in the cut in Q_π, which implies that w is unhappy in Q_π. Due to the assumption $\{w\} <_{Q_\pi} \{t_v^\varphi(Q_\pi, \sigma)\}$ and the property $\{w\} <_{Q_\pi} \{v\}$ following by (T1), node w remains unhappy as long as it does not flip. Then, since s is final, it follows that there is a flip of w in $s_{\pi+1}^\lambda$.

(F2): Due to (A2) all edges of the braid of v are in the cut in Q_π, which implies $\{v\} <_{Q_\pi} \{T_{G^\varphi}(w)\}$ according to Observation 3.5.8 (vi). Then the assumption $\{v\} <_{Q_i} \{u\}$ implies that neither u nor $T_{G^\varphi}(w)$ flips prior to the first flip of v in $s_{\pi+1}^q$, which in turn implies that w does not flip prior to the first flip of v in $s_{\pi+1}^q$. Thus, $\{v\} <_{Q_\pi} \{w\}$.

C2) Case $w \in R^\varphi(v) \setminus \{v\}$:
(F1): Then w is of Type I and its heaviest edge is not in the cut in Q_π. Thus, Lemma 3.5.18 (iii) implies that there is a flip of w in $s_{\pi+1}^\lambda$.

(F2): All nodes of the σ-th lower path with respect to Q_π unequal to $t_v^\varphi(Q_\pi, \sigma)$ are happy in Q_π due to Observation 3.5.8 (vi). Moreover, w is happy since it is of Type I and its heaviest edge is in the cut. Thus, all nodes of the σ-th lower path remain happy as long as node v does not flip.

C3) Case $w = \phi_{j,2}^\sigma(v)$ for $1 \leq \sigma \leq 2n^v$, $1 \leq j \leq 2m_{r(\sigma)}^v + 1$:
(F1): Since w is of Type I and it is influenced by $t_v^\varphi(Q_\pi, \sigma)$, it remains unhappy as long as neither itself nor $t_v^\varphi(Q_\pi, \sigma)$ flips. Consequently, the assumption $\{w\} <_{Q_\pi} \{t_v^\varphi(Q_\pi, \sigma)\}$ and (T1) together imply that there is a flip of w in $s_{\pi+1}^\lambda$.

(F2): Since the node that has influence on the Type I node w is a node of the σ-th lower path and all edges of this path are in the cut in Q_π, Observation 3.5.8 (vi) implies $\{v\} <_{Q_\pi} \{w\}$.

C4) Case $w = \phi_{2\omega-1,1}^\rho(v)$ for $1 \leq \omega \leq m_r^v$:
Let x be the unique anterior external informer adjacent to w and let $y \in Nodes(\varphi(v))$ such that $x \in T^\varphi(y)$. By assumption, we have $\{v\} <_{Q_i} \{y\}$ and therefore $\{v\} <_{Q_i} T^\varphi(y)$ according to Lemma 3.5.20. In particular, we get $\{v\} <_{Q_i} \{x\}$ since $x \in T^\varphi(y)$. Moreover, $T^\varphi(y)$ is flat in Q_i, since otherwise there was an unhappy node of $T^\varphi(y)$ in Q_i which would contradict $\{v\} <_{Q_i} T^\varphi(y)$. We let $Q \in \mathscr{P}(V^\varphi)$ be the partition arising from Q_π by choosing the colors of the nodes of $F^\varphi(v) \setminus \{u, v\}$ such that

$F^{\varphi}(v)$ is awaiting in Q.
(F1): Since $w = n_v^{\varphi}(Q_\pi, \sigma)$, node w has its unnatural color in Q_π. In Q, node w also has its unnatural color and the edge $\{w, x\}$ is not in the cut in Q due to Lemma 3.5.17 (i) which implies that it is also not in the cut in Q_π. By assumption, we have $\{w\} <_{Q_\pi} \{t_v^{\varphi}(Q_\pi, \sigma)\}$ and $\{v\} <_{Q_i} \{y, u\}$. Thus, $\{v\} <_{Q_i} T^{\varphi}(y) \cup \{u\}$ according to Lemma 3.5.20 and therefore $\{v\} <_{Q_i} \{x\}$. Since v does not flip in s_{i+1}^{π}, we get $\{v\} <_{Q_\pi} \{x\}$ which implies that w remains unhappy as long as it does not flip. Hence, there is a flip of w in $s_{\pi+1}^{\lambda}$.

(F2): The Type I node z adjacent to w via the heaviest edge incident to w is on the σ-th lower path and the heaviest edge incident to z is also on this path. Since all edges of this path are in the cut in Q_π, Observation 3.5.8 (vi) implies $\{v\} <_{Q_\pi} \{z\}$. Since w is on the σ-th lower path, node w has its natural color in Q_π. In Q, node w has its unnatural color and the edge $\{w, x\}$ is not in the cut in Q due to Lemma 3.5.17 (i) which implies that $\{w, x\}$ is in the cut in Q_π. Then, by the assumption $\{v\} <_{Q_i} \{y\}$ we get $\{v\} <_{Q_\pi} \{x\}$ due to Lemma 3.5.20. Consequently, the two properties $\{v\} <_{Q_\pi} \{z\}$ and $\{v\} <_{Q_\pi} \{x\}$ together imply for the Type II node w that $\{v\} <_{Q_\pi} \{w\}$.

C5) Case $w = \phi_{2m_r^v+1,1}^{\rho}(v)$:
(F1): Node w has its unnatural color in Q_π and has therefore the same color as in the partition Q arising from Q_π by choosing the colors of the nodes of $F^{\varphi}(v) \setminus \{u, v\}$ such that $F^{\varphi}(v)$ is awaiting in Q. According to Lemma 3.5.16 (ii)(d) the constant c_1^{ρ}—which is adjacent to w—has in Q and therefore also in Q_π the same color as w, i.e., the color κ. Consequently, since w is of Type II, it is unhappy in Q_π. Then, due to the assumption $\{w\} <_{Q_\pi} \{t_v^{\varphi}(Q_\pi, \sigma)\}$ and since c_1^{ρ} does not flip at all in s, there is a flip of w in $s_{\pi+1}^{q}$.

(F2): Node w remains happy as long as neither $\phi_{2m_r^v,2}^{\rho}(v)$ nor c_1^{ρ} flips. Since all edges of the σ-th lower path are in the cut in Q_π, we have $\{v\} <_{Q_\pi} \{\phi_{2m_r^v,2}^{\rho}(v)\}$ according to Observation 3.5.8 (vi). Then, since w is of Type II, we get $\{v\} <_{Q_\pi} \{w\}$.

C6) Case $w = \phi_{\omega,1}^{\sigma}(v)$ for any $1 \leq \sigma \leq 2n^v + 1$, $1 \leq \omega \leq 2m_{r(\sigma)}^v + 1$ with the assumption that $c_{Q_\pi}(\gamma_{\omega,2}^{p}) = \kappa$ and $\{v\} <_{Q_\pi} \{\gamma_{\omega,2}^{p}\}$ for $p := pr_{2m_{r(\sigma)}^v+1}^{-}(\sigma)$:
(F1): Node w is of Type II and is adjacent to $t_v^{\varphi}(Q_\pi, \sigma)$ and $\gamma_{\omega,2}^{p}$. Both of them have the color κ in Q_π. Since w has its unnatural color in Q_π, i.e., κ, it remains unhappy as long as neither any of these two neighbors nor w itself flips. However, by assumption, we have $\{v\} <_{Q_\pi} \{t_v^{\varphi}(Q_\pi, \sigma)\}$ and $\{v\} <_{Q_\pi} \{\gamma_{\omega,2}^{p}\}$. Thus, (T1) implies that there is a flip of w in $s_{\pi+1}^{\lambda}$.

(F2): Since w is of Type II, it remains happy as long as neither $\gamma_{\omega,2}^{p}$ nor

the node x adjacent to w via the heaviest edge incident to w flip. Since x is on the σ-th lower path and all edges of the σ-th lower path are in the cut in Q_π we have $\{v\} <_{Q_\pi} \{x\}$ according to Observation 3.5.8 (vi). Then the assumption $\{v\} <_{Q_\pi} \{\gamma^p_{\omega,2}\}$ implies $\{v\} <_{Q_\pi} \{w\}$.

C7) Case $w = \beta^\sigma_5(v)$ for $1 \leq \sigma \leq 2n^v$:
(F1): Then the two edges $\{w, \beta^\sigma_3(v)\}$ and $\{w, \beta^\sigma_4(v)\}$ are not in the cut which implies that w is unhappy in Q_π. Since both nodes $\beta^\sigma_3(v)$ and $\beta^\sigma_4(v)$ do by assumption not flip prior to the first flip of v in $s^q_{\pi+1}$, node w remains unhappy as long as none of the two nodes flips. Thus, there is a flip of w in s^q_π.

(F2): Node w remains happy as long as neither $\beta^\sigma_3(v)$ nor $\beta^\sigma_4(v)$ flips. Since all edges of the σ-th lower path are in the cut in Q_π for each σ for which w is a node of the σ-th lower path, we have $\{v\} <_{Q_\pi} \{\beta^\sigma_3(v)\}$ and $\{v\} <_{Q_\pi} \{\beta^\sigma_4(v)\}$ according to Observation 3.5.8 (vi). Then, since w is of Type III, we get $\{v\} <_{Q_\pi} \{w\}$.

Now we consider the last case for w. In it, we prove two properties that are slightly different compared to (F1) and (F2), respectively. The first property has the same premise as (F1) but two extra implications. We will denote the property by (F1*). The second property has the same implication as (F2) but two extra premises—the two extra premises correspond with the two extra implications of (F1*). We will denote the second property by (F2*). The properties are specified within the case which is as follows:

C8) Case $w = \phi^\rho_{2\omega,1}(v)$ for $1 \leq \omega \leq m^v_r$:
We let x be the posterior external informer adjacent to w, $y \in Nodes(\varphi(v))$ be such that $x \in B^\varphi(y)$ and $Q \in \mathscr{P}(V^\varphi)$ be the partition arising from Q_π by choosing the colors of the nodes of $F^\varphi(v) \setminus \{u, v\}$ such that $F^\varphi(v)$ is awaiting in Q.

F1*) *If* (A1) *is satisfied then there is an index* $i < j \leq \lambda$ *for which* $w_j = w$, $c_{Q_j}(x) = \kappa$ *and* $\{y\} <_{Q_j} \{x\}$:

Proof. Since w is a node of the σ-th upper path, it has its unnatural in Q_π. Thus, it has the same color in Q_π as in Q, namely κ.

At first we consider the case that $e_1 := \{w, x\}$ is in the cut in Q_π. The set $T^\varphi(y)$ is flat in Q_π since otherwise one of its nodes could flip, which contradicts the assumption $\{v\} <_{Q_i} \{y\}$ according to Lemma 3.5.20. Consequently, $B^\varphi(y)$ is not flat in Q_π since otherwise e_1 is not in the cut according to Lemma 3.5.17 (ii). Thus, there is exactly one edge e_2 on the path t from y to x along nodes of $B^\varphi(y)$ not in the cut. According to Lemma 3.5.18 (i) the node $x' \in B^\varphi(y)$ whose heaviest edge e_2 is not in the cut is unhappy then and can flip. No other node of $F^\varphi(y) \setminus \{H_G(y)\}$ is unhappy since $F^\varphi(y)$ is canonical in Q_π according to Lemma 3.5.19. Thus, by induction it

follows that the nodes on the path from x' to x flip consecutively. After their flips, the edge e_1 is not in the cut. Since w is of Type II, it is unhappy then. Let $\pi < \pi' \leq \lambda$ be the index for which $x = w_{\pi'}$. Then all edges on the path t are in the cut in $Q_{\pi'+1}$ and therefore we have $\{y\} <_{Q_{\pi'+1}} \{x\}$ according to Observation 3.5.8 (vi). Due to the assumption $\{v\} <_{Q_i} \{y\}$ we get $\{v\} <_{Q_{\pi'+1}} \{x\}$. Then the assumption $\{w\} <_{Q_\pi} \{t_v^\varphi(Q_\pi, \sigma)\}$ implies that there is a flip of w in $s^\lambda_{\pi+1}$. Due to (T1), node x does not flip prior to the first flip of w in $s^\lambda_{\pi'+1}$. Hence, x still has the color κ when w flips. Moreover, since neither y nor x flip in $s^\lambda_{\pi'+1}$ prior to the first flip of w, the property $\{y\} <_{Q_{\pi'+1}} \{x\}$ implies $\{y\} <_{Q_j} \{x\}$ for the index $i < j \leq \lambda$ with $w_j = w$.

Now consider the case that e_1 is not in the cut in Q_π, i.e., $c_{Q_\pi}(x) = \kappa$. Let Q' be the partition arising from Q_π by choosing the colors of the nodes of $B^\varphi(y) \setminus \{y\}$ such that $B^\varphi(y)$ is flat in Q'. Then Lemma 3.5.17 (ii) implies that e_1 is not in the cut in Q'. Consequently, we have $c_{Q'}(x) = c_{Q_\pi}(x)$. Since $F^\varphi(y)$ is according to Lemma 3.5.19 canonical in Q_π, there is at most one edge of t not in the cut. Thus, it follows that all edges of t are in the cut which implies $\{y\} <_{Q_\pi} \{x\}$. Thus, there is a flip of w in $s^\lambda_{\pi+1}$. Due to $\{y\} <_{Q_\pi} \{x\}$ and the assumption $\{v\} <_{Q_i} \{y\}$, node x does not flip prior to the first flip of w in $s^\lambda_{\pi+1}$. Thus, x still has the color κ when w flips. Moreover, since neither y nor x flip in $s^\lambda_{\pi+1}$ prior to the first flip of w, the property $\{y\} <_{Q_\pi} \{x\}$ implies $\{y\} <_{Q_j} \{x\}$ for $i < j \leq \lambda$ with $w_j = w$.

F2*) *If* (A2) *is satisfied,* $c_{Q_\pi}(x) = \kappa$ *and* $\{y\} <_{Q_\pi} \{x\}$ *then* $\{v\} <_{Q_\pi} \{w\}$:

Proof. Since w is on the σ-th lower path, it has its natural color, i.e., $\overline{\kappa}$. Node z adjacent to w via the heaviest edge incident to w is on the σ-th lower path and all edges of this path are in the cut in Q_π. Thus, Observation 3.5.8 (vi) implies $\{v\} <_{Q_\pi} \{z\}$. Then the properties $c_{Q_\pi}(x) = \kappa$ and $\{y\} <_{Q_\pi} \{x\}$ together imply $\{v\} <_{Q_\pi} \{w\}$.

Now we distinguish between the possible cases for $w = n_v^\varphi(Q_\pi, \sigma)$ if $w \in F^\varphi(v) \setminus \{u, v\}$ and show for each of them that there is an index $\pi < \tau < \lambda$ such that the following properties hold:

S1) $w = w_\tau$.

S2) $\{v\} <_{Q_\tau} \{w\}$.

Then, for the node $z \in F^\varphi(v) \setminus \{u, v\}$ adjacent to v via the heaviest edge incident to v the properties (S1) and (S2) imply that z flips to its natural color in $s^\lambda_{\pi+1}$ and keeps its natural color until v flips. Thus, after the flip of z there is a flip of v. After the flip of v we get that $F^\varphi(v)$ is enterable as postulated by the lemma—recall that $F^\varphi(v)$ is canonical in Q_k for all $0 \leq k \leq q$ according to 3.5.19 and that u does by assumption not flip prior to

the first flip of v in s^q_{i+1}. The consideration of the cases is divided into four parts (P1)–(P4):

P1) Case $w \in B := B^\varphi(v) \cup \{\alpha_1(v), \alpha_2(v)\} \cup \bigcup_{1 \le j \le 2n^v+1} \{\beta_1^j, \beta_2^j\}$:
At first, we consider the case $w \in B^\varphi(v)$ for which w is adjacent to v. Then we get $\{v\} <_{Q_\pi} \{w\}$ since all edges of the σ-th lower path are in the cut and therefore all nodes of the path unequal to w are happy in Q_π due to Observation 3.5.8 (vi). Then property (F1) of (C2) implies (S1), whereafter property (F1) implies (S2) for this case.

We show the remaining cases via induction. In particular, the hypothesis that $\{w\} <_{Q_\pi} \{t_v^\varphi(Q_\pi, \sigma)\}$ for all σ for which w is a node of $c_v^\varphi(\sigma)$ implies due to (F1) of (C2) the property (S1), which in turn implies (S2) due to (F2) of (C2). Thus, the two properties (S1) and (S2) follow for all nodes of B by induction.

P2) Case $w = \phi^k_{i,j}(v)$ or $w = \gamma^k_{i,j}(v)$ for any i, j, k:
The two properties (S1) and (S2) are shown via induction on k. For the induction basis we consider the case $k = \rho$ and show by induction on i that the properties (S1) and (S2) hold for $w = \phi^\rho_{i,j}(v)$ for all i, j and then, also by induction on i, that the two properties also hold for $w = \gamma^\rho_{i,j}(v)$ for all i, j, k.

As induction basis, we consider the case $w = \phi^\rho_{1,1}(v)$. Then property (S2) of (P1) and (F1) of (C4) together imply property (S1) for w whereafter we get (S2) from (F2) of (C4). Then (F1) and (F2) of (C2) imply the properties (S1) and (S2) also for $w = \phi^\rho_{1,2}(v)$. Assume as induction hypothesis for an arbitrary $1 < i \le 2m^v_{r(\rho)} + 1$ that for $\phi^\rho_{i-1,2}(v)$ the properties (S1) and (S2) hold. If $i = 2t - 1$ for an index $1 \le t \le m^v_{r(\rho)}$ then (F1) and (F2) of (C4) imply the properties (S1) and (S2) and if $i = 2t$ then (F1*) and (F2*) of (C8) imply the properties (S1) and (S2). Moreover, if $i = 2m^v_{r(\rho)} + 1$ then the two properties follow from (F1) and (F2) of (C5). Then, analogously to the case $i = 1$, (F1) and (F2) of (C2) imply (S1) and (S2) for $\phi^\rho_{i,2}(v)$. Consequently, the property (S2) of $\phi^\rho_{2m^v_{r(\rho)}+1,2}(v)$ implies (S1) for $\gamma^\rho_{1,1}(v)$ according to (F1) of (C2) whereafter we get (S2) for the same node. Then (C2) implies the two properties inductively for the remaining nodes $\gamma^\rho_{i,j}(v)$ for any i, j.

Now assume as induction hypothesis that for an arbitrary $1 \le k \le 2n^v$ the properties (S1) and (S2) hold for all $\phi^k_{i,j}(v)$ and $\gamma^k_{i',j'}(v)$ for all i, j, i', j'. We show that the two properties then also hold for all $\phi^{k'}_{a,b}(v)$ and $\gamma^{k'}_{a',b'}(v)$ for all a, b, a', b' where $k' := (k \mod 2n^v) + 1$. We prove this statement also by induction on i. As induction basis, we consider the case $w = \phi^{k'}_{1,1}(v)$. Then the induction hypothesis and property (F1) of (C6) together imply property (S1), which in turn implies (S2) due to property (F2) of (C6). Then (F1) and (F2) of (C2) imply the properties

(S1) and (S2) also for $w = \phi^{k'}_{1,2}(v)$. Assume as induction hypothesis for an arbitrary $1 < i \leq 2m^{v}_{r(k')} + 1$ that for $\phi^{k'}_{i-1,2}(v)$ the conditions (S1) and (S2) are satisfied. Then the properties (F1) and (F2) of (C6) imply the properties (S1) and (S2) for $\phi^{k'}_{i,1}(v)$. Then, analogously to the case $k = \rho$, the properties (F1) and (F2) of (C2) imply the properties (S1) and (S2) for $\phi^{k'}_{i,2}(v)$.

P3) Case $w \in \bigcup_{1 \leq i \leq 2n^{v}} \{\beta^{i}_{3}, \beta^{i}_{4}, \beta^{i}_{5}\}$:
Again, we show the two properties (S1) and (S2) by induction on i. For the induction basis we consider the case $i = 2n^{v}$. According to (S2) of (P1) we have $\{v\} <_{Q_\pi} \{\beta^{2n^{v}}_{2}(v)\}$. Thus, (F1) of (C2) implies (S1) for $\beta^{2n^{v}}_{3}(v)$, whereafter (F2) of (C2) implies (S2) for the same node. Property (F2) of (P2) implies for the node z adjacent to $\beta^{2n^{v}}_{4}(v)$ via the heaviest edge incident to $\beta^{2n^{v}}_{4}(v)$ that $\{v\} <_{Q_\pi} \{z\}$. Then (F1) of (C2) implies (S1) for $\beta^{2n^{v}}_{4}(v)$, whereafter (F2) of (C2) implies (S2) for it. Then we get property (S1) for $\beta^{2n^{v}}_{5}(v)$ due to (F1) of (C7) whereafter (F2) of (C7) implies (S2) for it.

As induction hypothesis we assume for an arbitrary $1 \leq i < 2n^{v}$ that the properties (S1) and (S2) are satisfied for $\beta^{i+1}_{5}(v)$. Then property (S1) follows for the node $\beta^{i}_{3}(v)$ according to (F1) of (C2) whereafter we get (S2) due to (F2) of (C2). Then, analogously to the induction basis, the properties (S1) and (S2) follow for the nodes $\beta^{i}_{4}(v)$ and $\beta^{i}_{5}(v)$.

P4) Case $w \in T^{\varphi}(v) \setminus \{v\}$:
If w is adjacent to $\beta^{1}_{5}(v)$ then property (S1) follows from (S2), (P3), and (F1) of (C2) whereafter (F2) of (C2) implies (S2). For the remaining cases the hypothesis that (S2) is satisfied for $t^{\varphi}_{v}(Q_\pi, \sigma)$ implies due to (F1) of (C2) the property (S1) for w whereafter (F2) of (C2) implies (S2). Thus, by induction the properties (S1) and (S2) follow for all nodes of $T^{\varphi}(v) \setminus \{v\}$.

This finishes the proof of the Filtering Lemma. □

3.5.3 Enforcing Pivot-Rules with Combined Subgraphs

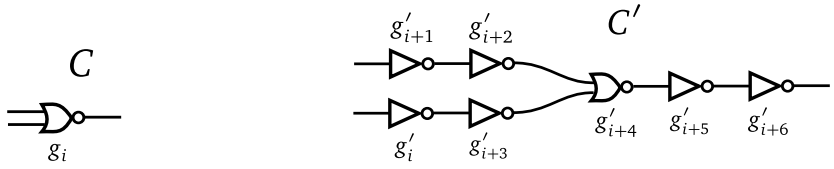
Theorem 3.5.22 (Enforcing Theorem). *Let C be a Boolean circuit, $G^{C} = (V^{C}, E^{C})$ be the graph that constitutes C, n be the number of gates in C, P^{C} be an ordinary partition of V^{C}, h be a generalized pivot rule in G^{C} which is computable in $O(poly(n))$ time and t be the sequence starting at (G^{C}, P^{C}) induced by h. Then one can compute in $O(poly(n))$ time a graph $G^{h} = (V^{h}, E^{h})$ with $V^{C} \subseteq V^{h}$ and a partition $P_0 \in \mathscr{P}(V^{h})$ with $P_0|_{V^{C}} = P^{C}$ such that for each final sequence $s = (w_1, \ldots, w_q)$ starting at (G^{h}, P_0) we get*

$$s|_{V^{C}} = t.$$

Proof. For the sake of simplicity, we use the same names for the gates in C and the nodes that represent the gates in G^C—recall that each gate is represented by exactly one node in the graph that constitutes the circuit. Without loss of generality, we make the following five assumptions. First, a gate of C is either a NOT-gate with a fan-in of one and a fan-out of at most two or a NOR-gate with a fan-in of two and a fan-out of one—this assumption can be made due to Proposition 2.4.3. The NOT-gates are used to distribute the output of the NOR-gates without violating the condition that all nodes of the resulting graph have maximum degree four. Second, C contains the gates $g_1, g_2, \dots, g_n$ for $n \in \mathbb{N}$ which are topologically sorted such that if g_i is input to g_j then $i < j$. The proof that the following three assumptions can be made without loss of generality is given after their statement. Third, for any NOR-gate g_i of C the inputs of g_i in C are $g_{i-2} =: I_1(g_i)$ and $g_{i-1} =: I_2(g_i)$, gate g_{i+1} is the gate whose input is g_i, all three gates g_{i+1}, g_{i-1} and g_{i-2} are NOT-gates and the opposite color of g_i in P^C. Fourth, for each partition $P \in \mathscr{P}(V^C)$ in which a NOR-node v is the unique unhappy NOR-node of G^C we have $h(P) = v$. Fifth, for any NOR-node $g_i \in V^C$ and any partition $P \in \mathscr{P}(V^C)$ we have $h(P') = g_{i+1}$ if and only if $h(P) = g_i$ for the partition $P' \in \mathscr{P}(V^C)$ arising from P by flipping g_i.

Now we show that the last three of the above assumptions can be made without loss of generality. For each NOR-gate g_i of C we can construct a circuit C' from C which computes the same function as C by iteratively adding NOT-gates and renaming the nodes as depicted in Figure 3.13. From P we can construct a partition P' of the nodes of the graph $G^{C'} = (V^{C'}, E^{C'})$ that constitutes C' by assigning to the nodes g'_i, g'_{i+1} and g'_{i+6} the color $c_P(g_i) =: c_{P'}(g'_{i+4})$ and the opposite color to the nodes g'_{i+2}, g'_{i+3} and g'_{i+5}. The colors of the remaining NOT-nodes of $G^{C'}$ are chosen such that they correspond to the colors of their corresponding NOT-nodes in G^C. From the generalized pivot rule h we construct a pivot rule h' in the following way. Let $Q' \in \mathscr{P}(V^{C'})$ and $Q \in \mathscr{P}(V^C)$ such that $Q'|_{V^C} = Q$. If $h(Q) \in V^C_{not}$ then we let $h'(Q')$ return the node of $V^{C'}_{not}$ that corresponds to $h(Q)$. On the other hand, if $h(Q) = g_i$ for $g_i \in V^C_{nor}$ then we let $h'(Q')$ return a node of $V^{C'}$ in the following way. If the NOR-node of $V^{C'}$ that corresponds to g_i—for convenience, we let g'_{i+4} be this node—is unhappy in Q' then $h'(\cdot)$ returns g'_{i+4}, otherwise it returns the unhappy node of the set $\{g'_i, \dots, g'_{i+6}\} \setminus \{g'_{i+4}\}$ with the highest index—note that one of the nodes of the set $\{g'_i, \dots, g'_{i+3}\}$ is necessarily unhappy if g_i is unhappy in Q but g'_{i+4} is happy in Q'. Then a NOR-node of $V^{C'}$ is returned by $h'(\cdot)$ if it is the unique unhappy NOR-node of the given partition and node g'_{i+5} is chosen by $h'(\cdot)$ directly after $h'(\cdot)$ chooses g'_{i+4}. Since the flips of the added NOT-gates do not occur in $s|_{V^C}$, the last three assumptions can be made without loss of generality.

The proof in a nutshell We extend the graph G^C by extra nodes and edges and name the initial colors of the added nodes. The purpose of the added nodes is, depending on the given colors of the nodes of V^C, to allow only that node of V^C to flip which is chosen by the generalized pivot rule h. The proof consists of four parts. First, we extend G^C by further nodes and edges and call the hereby arising graph $G^h = (V^h, E^h)$. The nodes and edges added in this step consist of two subsets of nodes and edges. One subset



(a) Gate g_i of circuit C

(b) Gate g'_{i+4} in the extended circuit C' corresponds to gate g_i in C

Figure 3.13: Adding NOT-gates allows desired numbering and behavior.

constitutes a Boolean circuit C^h whose main purpose is to compute the generalized pivot rule h. The other one contains nodes and edges which are supposed to hinder that nodes of V^C from flipping which are not chosen by the generalized pivot rule h for the next flip. Second, we name the colors of the nodes of G^h in the initial partition P_0. Third, we introduce a function $\varphi : V_I^h \to \Phi(V_I^h)$ whose purpose is, beside some technical purposes, to ensure by means of the Filtering Lemma (i.e., Lemma 3.5.21) that each node that represents a gate of C^h only switches to its correct color with respect to the colors of its inputs when its input nodes already have their correct colors with respect to the colors of their corresponding inputs. The graph and the corresponding partition induced by the Filtering Lemma will be called $G^\varphi = (V^\varphi, E^\varphi)$ and R_0, respectively. Fourth, we show for all sequences starting at (G^φ, R_0) that the nodes of V^C in fact flip in the order that is induced by h.

1) Extend G^C In this part, we add nodes and edges to the graph G^C whereby we get the graph G^h. The description of G^h is divided into three steps. In the first two steps we add gates to the circuit C and call the resulting circuit C^1. In the third step we substitute edges in the graph G^1 that constitutes C^1 by further nodes and edges. Let V^C_{core} be the set of nodes of V^C_{not} without the nodes that represent input nodes of C. For a partition $P \in \mathscr{P}(V^C_{core})$ let $P' \in \mathscr{P}(V^C)$ in the following be the partition arising from P by assigning that color to each input node and to each NOR-node such that it is happy.

In the first step we add a separate Boolean circuit C^h. The circuit C^h takes as input the bitwise complement of the bit vector encoding the colors of the nodes of V^C_{core} in a partition $P \in \mathscr{P}(V^C_{core})$ and returns the bit vector that differs from the input in that and only that component of the bit vector which corresponds to the node $h(P')$—if $h(P') = nil$ then each output bit equals its corresponding input bit. For C^h we make the following five assumptions without loss of generality. First, it only contains NOT-gates with a fan-in of one and a fan-out of at most two and NOR-gates with a fan-in of two and a fan-out of one—this assumption can be made according to Proposition 2.4.3. Second, we denote the gates of C^h by $\gamma_1, \ldots, \gamma_m$ where m is the number of gates of C^h and assume that the gates are topologically sorted such that $i < j$ if γ_i is input to γ_j. Third, similarly to the circuit C, we assume that the inputs of a NOR-gate γ_j are NOT-gates $\gamma_{i-2} =: I_1(\gamma_i)$ and $\gamma_{i-1} =: I_2(\gamma_i)$ and each of them has a fan-out of one and as input also a NOT-gate. Fourth, for each NOR-gate γ_j the unique gate for which γ_j is an input is the NOT-gate

γ_{j+1} and γ_{j+1} is not an output gate of C^h. Fifth, for each $g_i \in V^C_{not}$ the gate γ_i is the gate of C^h that takes as input the bit of the input assignment that corresponds to g_i and $\gamma_{m-n+i} =: \tau_i$ the corresponding output gate—this assumption can be made since one can substitute links by two NOT-gates linked in series without changing the output of the circuit.

In the second step we add the gates with the white fillings as depicted in Figure 3.14 and the corresponding links as shown in the figure. The added gates connect gates of C with gates of C^h. The gates with the gray filling are gates of C and C^h, they are redrawn to determine the inputs and outputs of some of the added gates. The gray rectangles are to indicate the gates whose corresponding nodes in G^h make up the sets of nodes V_i^0, V_i^1 and V_i^2. We call the resulting circuit C^1 and let white be the **natural** color of $\mu_i^{\kappa,\omega}$ for all $0 \leq \kappa \leq 1$ and all even $0 \leq \omega \leq 2m$ and black be their **unnatural** color. For $\mu_i^{\kappa,\omega}$, all $0 \leq \kappa \leq 1$ and all odd $1 \leq \omega \leq 2m-1$ we let black be their natural color and white be their unnatural color.

Comment The idea of the gates $\mu_i^{\kappa,\omega}$ for all i, κ, ω is to control, by means of the Filtering Lemma (i.e., Lemma 3.5.21) and a function φ which is specified later, the stepwise adaption of the correct colors of the nodes representing the gates of C^h with respect to the colors of their input nodes in ascending order with respect to their topological order. The insistence on the correction of the gates according to their topological order is necessary, since if some gate γ_j does not take its correct color before the gates it is an input to take their correct color with respect their inputs, then the output of the circuit C^h can be updated incompletely, which can lead to a flip of a node that is unequal to the one indicated by the generalized pivot rule. The gates will perform their aim in the following way: Suppose that node $g_i \in V^C_{not}$ flips its color. In the partitions we will be interested in, the edge $\{g_i, \lambda_i^{0,1}\}$ and edges between the nodes of V_i^2 are in the cut, as well as the edges between the nodes of V_i^0 and the ones between the nodes of V_i^1. Then, after the flip of g_i, the nodes in V_i^2 change their colors and in exactly one of the sets V_i^0 and V_i^1 all nodes take their unnatural colors, as we will see later,—the function φ together with the Filtering Lemma will ensure that all nodes of this set in fact take their unnatural colors. In particular, if g_i flips to black then the nodes in V_i^0 take their unnatural colors, and if it flips to white then the nodes in V_i^1 take their unnatural colors. Let $\kappa \in \{0,1\}$ be such that the nodes of V_i^κ flipped to their unnatural colors. Then we let the nodes $\mu_i^{\kappa,\omega}$ for all $0 \leq \omega \leq 2m$ switch back to their natural colors consecutively in ascending order with respect to ω. However, some of the nodes $\mu_i^{\kappa,\omega}$ are hindered via the function φ and the Filtering Lemma from flipping back to their natural colors unless some corresponding gate node γ_k for $1 \leq k \leq m$ has its correct color with respect to the colors of its inputs. Furthermore, each NOT-node of V^{C^h} is hindered from flipping to its correct color before for a certain $0 \leq \omega \leq 2m$ the nodes $\mu_i^{\kappa,\omega}$ for all $g_i \in V^C_{not}$ have their natural colors. Then the gates of C^h take their correct colors consecutively according to their topological order, with a single exception that is due to a technical reason.

In the third step we substitute in the graph G^1 that constitutes the circuit C^1 the edge

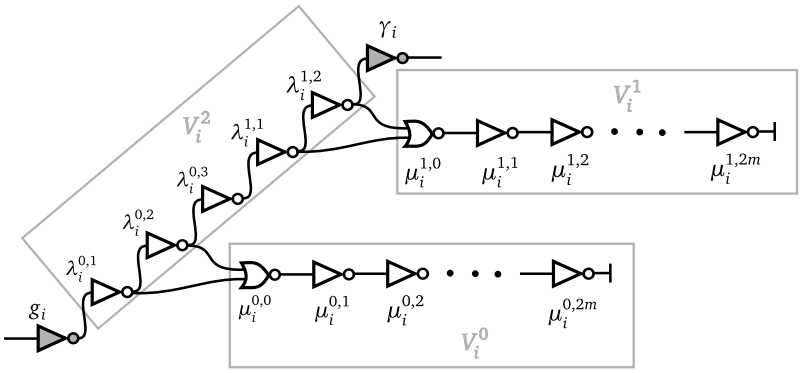


Figure 3.14: The gates that connect a NOT-gate g_i of C^1 with gate γ_i of C^h.

$\{I(g_i), g_i\}$ for each $g_i \in V_{not}^C$ for which g_i does not represent an input gate of C with weight $a \in \mathbb{Q}_{>0}$ by the nodes and edges presented in Figure 3.15. The nodes in the figure that have gray circumcircles were already introduced and are redrawn to determine the edges of the added nodes. The value of $\epsilon > 0$ is chosen small enough such that the types of g_i and τ_i remain the same, π_i is of Type III and has no influence on τ_i and ρ_i has influence on g_i, i.e., the heaviest edge of g_i is $\{\rho_i, g_i\}$. This finishes the description of $G^h = (V^h, E^h)$. For an overview of the graph G^h with the subgraphs G^C and G^{C^h} that constitute the circuits C and C^h see Figure 3.16.

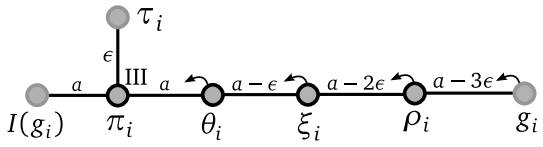


Figure 3.15: Nodes $\pi_i, \theta_i, \xi_i, \rho_i$ and incident edges substitute the edge $\{I(g_i), g_i\}$ of G^1.

Comment The purpose of the nodes π_i, θ_i, ξ_i and ρ_i is as follows. In certain partitions we will be interested in, node π_i and ξ_i will have the same color as g_i whereas θ_i and ρ_i have the opposite color. If the function h chooses g_i for the next flip in such a partition and the colors of the nodes that represent the circuit C^h reflect a correct computation of C^h then τ_i will have, as we see later, the same color as $I(g_i)$ and therefore also the same color as g_i. Then π_i flips followed by a sequence of flip of θ_i, ξ_i and ρ_i whereafter g_i becomes unhappy. On the other hand, if h chooses a node unequal to g_i for the next flip then node τ_i will have the opposite color of g_i, which implies that all five nodes $\pi_i, \theta_i, \xi_i, \rho_i$ and g_i are happy. In this way, the added nodes will make the node g_i unhappy if and only if h chooses g_i for the next flip. The nodes θ_i and ξ_i are only added to for a technical reason—the above-mentioned behavior could also be achieved if there was an edge directly connecting π_i and ρ_i.

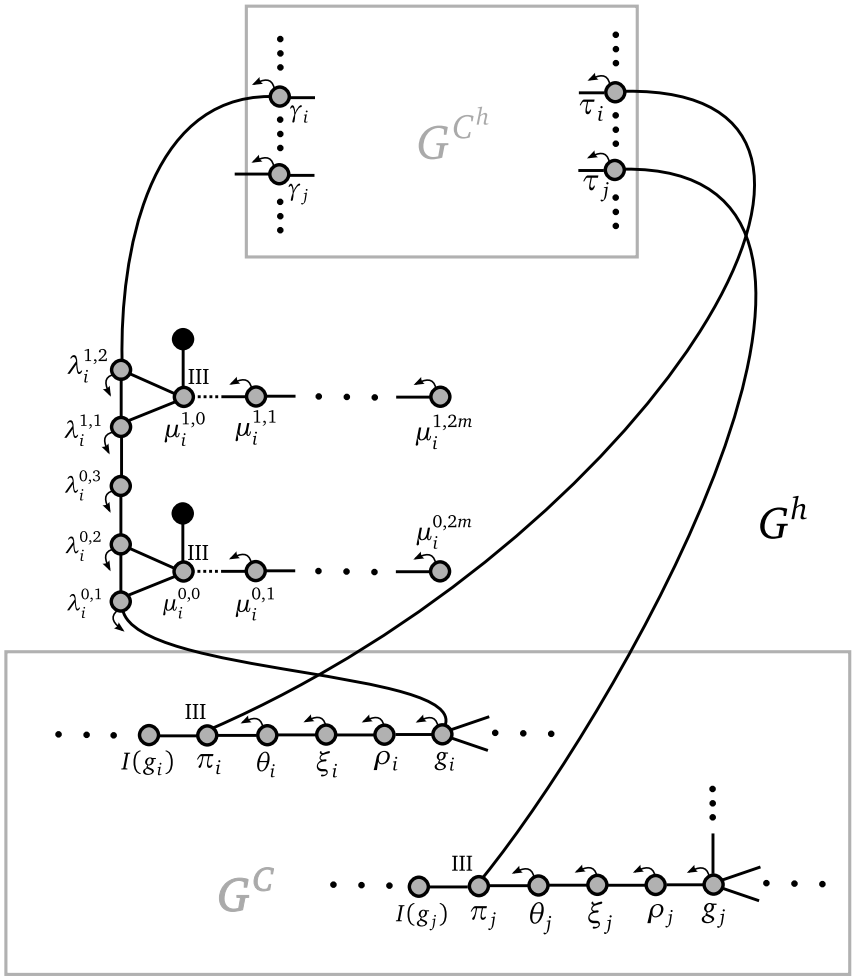


Figure 3.16: Overview of the graph G^h.

2) Assign colors to V^h In the following, we name the colors of the nodes of G^h in the initial solution P_0. We let $P_0|_{V^C} = P^C$ and use for the colors of the remaining nodes the following definition.

Definition 3.5.23. *A partition $P \in \mathscr{P}(V^h)$ is called **recurring** if the following conditions are satisfied:*

R1) $c_P(g_i) = \neg(c_P(g_{i-1}) \vee c_P(g_{i-2}))$ *for any* $g_i \in V^C_{nor}$.

R2) For any $g_i \in V^C_{not}$, $\kappa \in \{0,1\}$ *the following conditions are satisfied:*

R2i) $c_P(g_i) = c_P(\rho_i) \neq c_P(\xi_i) \neq c_P(\theta_i) \neq c_P(\pi_i)$ *if* $h(P|_{V^C_{not}}) = g_i$ *and* $c_P(g_i) \neq c_P(\rho_i) \neq c_P(\xi_i) \neq c_P(\theta_i) \neq c_P(\pi_i)$ *otherwise.*

R2ii) $c_P(\lambda_i^{0,2}) \neq c_P(\lambda_i^{0,3})$ *and* $c_P(\lambda_i^{\kappa,j}) \neq c_P(I(\lambda_i^{\kappa,j}))$ *for all* $1 \leq j \leq 2$.

R2iii) $\mu_i^{\kappa,j}$ *has its natural color in P for each* $0 \leq j \leq 2m$.

R3) $c_P(\gamma_i) \neq c_P(I(\gamma_i))$ *for any* $\gamma_i \in V_{not}^{C^h}$.

R4) $c_P(\gamma_i) = \neg(c_P(I_1(\gamma_i)) \vee c_P(I_2(\gamma_i)))$ *for any* $\gamma_i \in V_{nor}^{C^h}$.

For a given partition of the nodes of V^C a recurring partition of V^h can be computed in polynomial time by choosing the colors of the nodes of $V^h \setminus V^C$ consecutively according to (R2)–(R4). For P_0 we choose the colors of the nodes in $V^h \setminus V^C$ such that P_0 is recurring where the colors of the nodes of V^{C^h} that do not represent gates of C^h are chosen such that $P_0|_{V^{C^h}}$ is ordinary—note that the nodes of V^{C^h} that represent input gates of C^h are happy in P_0 due to (R3).

3) Function φ We describe the function φ by stating a SAT-formula $\varphi(v)$ for each $v \in V_I^h$. For each $v \in V_I^h$ for which we do not explicitly state $\varphi(v)$ we let $\varphi(v) = \emptyset$. The idea of the function φ is to use the Filtering Lemma (i.e., Lemma 3.5.21) to hinder certain nodes of V_I^h from flipping in certain partitions. In the following, we informally describe the supposed functionality of a considered formula, and how it fits in the overall plan for the sequences of flips and formally introduce it. For this, we let Q be a recurring partition of V^h and $h(Q) = g_i$ for $g_i \in V_{not}^C$.

Comment As we will see later, g_i is the only unhappy node of V^h in Q and will therefore flip. After that, there will be a flip of $\lambda_i^{0,1}$ since $\lambda_i^{0,1}$ has the opposite color of g_i in Q according to (R2ii).

We first consider the case that g_i flipped to black. Then $\lambda_i^{0,1}$ flips to white whereafter $\mu_i^{0,0}$ and $\lambda_i^{0,2}$ are unhappy. To guarantee that all nodes $\mu_i^{0,\omega}$ for $0 \leq \omega \leq 2m$ flip to their unnatural colors in ascending order in ω, we hinder $\lambda_i^{0,2}$ from flipping to black unless $\mu_i^{0,2m}$ already has its unnatural color white as opposed to the natural color that it has in Q according to (R2iii).

Now assume that g_i flipped to white. Then, as we will see later, the nodes $\lambda_i^{0,1}$, $\lambda_i^{0,2}$, $\lambda_i^{0,3}$ and $\lambda_i^{1,1}$ flips consecutively—note that $\mu_i^{0,0}$ does not become unhappy after the flip of $\lambda_i^{0,1}$ since $\lambda_i^{0,1}$ flips to black and $\mu_i^{0,0}$ has its natural color, i.e., white. Then, due to the flip of $\lambda_i^{1,1}$ to the white color, the nodes $\mu_i^{1,0}$ and $\lambda_i^{1,2}$ are unhappy and we hinder $\lambda_i^{1,2}$ from flipping to black unless $\mu_i^{1,2m}$ has its unnatural color.

Altogether, we let

$$\varphi(\lambda_i^{\kappa,2}) = \lambda_i^{\kappa,1} \vee \mu_i^{\kappa,2m} \text{ for all } \kappa \in \{0,1\}, g_i \in V_{not}^C. \tag{3.5.24}$$

Comment Note that if for any $\kappa \in \{0,1\}$ node $\mu_i^{\kappa,0}$ was unhappy after the flip of $\lambda_i^{\kappa,1}$, then after the subsequent flip of $\lambda_i^{\kappa,2}$ the node $\mu_i^{\kappa,0}$ is unhappy again and can flip back to its natural color white. Now assume that $c_Q(g_i) = \kappa$ for $\kappa \in \{0,1\}$ and that all nodes of V_i^2 flipped exactly once after the flip of g_i and all nodes of V_i^κ flipped at least once.

In the partitions arising thereby, we want the nodes of V_i^κ to act as a control of the consecutive correction of the gate nodes of V^{C^h} according to their topological order.

To guarantee this, we would like to simply hinder each $\mu_i^{\kappa,2j}$ for $\gamma_j \in V_{not}^{C^h}$ from flipping back to its natural color if γ_j is incorrect with respect to the colors of its input. Unfortunately, in this case the following problem arises. According to our assumption, the indices of the NOT-nodes adjacent to a NOR-node $\gamma_j \in V^{C^h}$ are such that its input nodes are γ_{j-2} and γ_{j-1} and the node for which γ_j is the input is γ_{j+1}. Assume that γ_j has to switch its color if its input nodes take their correct colors and that γ_{j+1} has the opposite color of γ_j. If all NOT-gates flip to their correct colors according to their topological order then there is a flip of γ_j between the flip(s) of its input nodes and the flip of γ_{j+1}. However, if γ_j already *has* its correct color with respect to the correct colors of its input then γ_j might flip *twice*. In particular, if the input nodes of γ_j have unequal colors, γ_j is white and the input node that is black flips to white before the other input node flips, then γ_j becomes unhappy and can flip to black. If the other input subsequently flips to black then γ_j becomes unhappy again and can flip back to white. But if γ_j flips twice without an intermediate flip of γ_{j+1} then we cannot argue via the Filtering Lemma (i.e., Lemma 3.5.21) whether γ_{j+1} is hindered from flipping or even which color γ_{j+1} has.

Therefore, we make a distinction between the NOT-nodes and hinder the input node of a NOR-node γ_j with lower index with respect to the topological order, i.e., γ_{j-2}, from flipping to white if the other input node, i.e., γ_{j-1}, does not yet have its correct color. Only when γ_{j-1} has its correct color, we allow γ_{j-2}—via the Filtering Lemma—to flip to its correct color. Then a double flip of γ_j without an intermediate flip of γ_{j+1} is impossible—in fact, γ_j does not flip at all in this case. For an overview of the classification of the NOT-nodes of C^h see Figure 3.17.

Definition 3.5.25. *The set $\boldsymbol{N_1}$ contains each node γ_j that represents an input gate of C^h. The set $\boldsymbol{N_2}$ contains each node that represents a NOT-gate $\gamma_i \in C^h$ for which γ_{i+2} is a NOR-gate of C^h. The set $\boldsymbol{N_3}$ contains each node that represents a NOT-gate $\gamma_i \in C^h$ for which γ_{i-1} is a NOR-gate of C^h. The set $\boldsymbol{N_4}$ contains all nodes that represent NOT-gates of C^h that are not contained in N_i for any $1 \leq i \leq 3$.*

Comment Note that the sets N_i for $1 \leq i \leq 4$ are pairwise disjoint due to our assumption that in C^h each input of a NOR-gate is a NOT-gate whose input is also a NOT-gate.

For all $\gamma_j \in N_1 \cup N_3 \cup N_4$ we hinder γ_j from flipping unless $\mu_i^{\kappa,2j-1}$ has its natural color, i.e., black, for all $g_i \in V_{not}^C$ and $\kappa \in \{0,1\}$:

$$\varphi(\gamma_j) = \bigwedge_{g_i \in V_{not}^C, \kappa \in \{0,1\}} \mu_i^{\kappa,2j-1} \text{ for all } \gamma_j \in N_1 \cup N_3 \cup N_4. \tag{3.5.26}$$

Let $\gamma_j \in N_2$ and $I(\gamma_j) = \gamma_k$ for $\gamma_k \in V_{not}^{C^h}$. We hinder γ_j from flipping unless $\mu_i^{\kappa,2j-1}$ has its natural color, i.e., black, for all $g_i \in V_{not}^C$, $\kappa \in \{0,1\}$ and γ_k is white or $\mu_i^{\kappa,2j+2}$ has

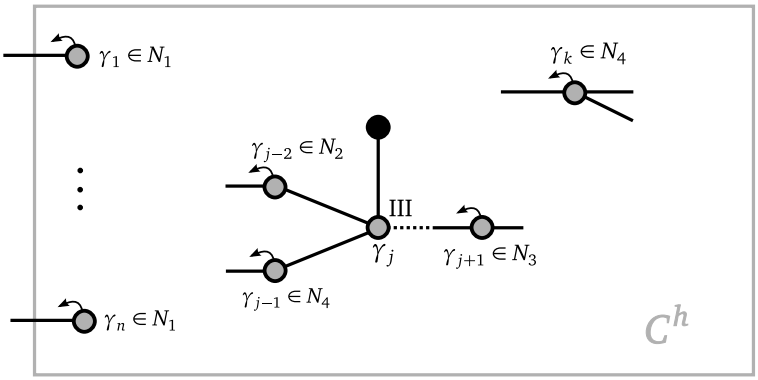


Figure 3.17: The classification of the nodes of C^h.

its natural color, i.e., white, for all $g_i \in V_{not}^C$, $\kappa \in \{0,1\}$ and γ_k is black:

$$\varphi(\gamma_j) = (\overline{\gamma_k} \wedge \bigwedge_{g_i \in V_{not}^C, \kappa \in \{0,1\}} \mu_i^{\kappa,2j-1}) \vee (\gamma_k \wedge \bigwedge_{g_i \in V_{not}^C, \kappa \in \{0,1\}} \overline{\mu_i^{\kappa,2j+2}}) \quad \text{for all } \gamma_j \in N_2 \text{ with } I(\gamma_j) = \gamma_k \text{ for } \gamma_k \in V_{not}^{C^h}. \tag{3.5.27}$$

Comment As in the formula for the nodes of $N_1 \cup N_3 \cup N_4$, the first part of the formula hinders the node γ_j from flipping as long as at least one $\mu_i^{\kappa,2j-1}$ has its unnatural color, i.e., white. The second part is to prevent a double flip of the NOR-node γ_{j+2}—see comment after (3.5.24). For this purpose, the formula $\varphi(\gamma_j)$ hinders γ_j from flipping to white as long as there is a node $\mu_i^{\kappa,2j+2}$ for any $g_i \in V_{not}^C$, $\kappa \in \{0,1\}$ that has its unnatural color, i.e., white. In this way it is ensured that the other input of the NOR-node γ_{j+2} takes its correct color—should it have been incorrect—before γ_j flips to white. Then a double flip of γ_{j+2} is prevented.

Let $g_i \in V_{not}^C$, $\gamma_j \in N_1$ and $\kappa \in \{0,1\}$. We hinder $\mu_i^{\kappa,2j}$ from flipping unless $\mu_i^{\kappa,2j-1}$ has its unnatural color, i.e., white, or γ_j has the opposite color of $\lambda_j^{0,1}$. Formally,

$$\varphi(\mu_i^{\kappa,2j}) = \overline{\mu_i^{\kappa,2j-1}} \vee (\gamma_j \wedge \overline{\lambda_j^{0,1}}) \vee (\overline{\gamma_j} \wedge \lambda_j^{0,1}) \quad \text{for all } g_i \in V_{not}^C, \gamma_j \in N_1, \kappa \in \{0,1\}. \tag{3.5.28}$$

Comment The satisfaction of the formula in the case that $\mu_i^{\kappa,2j-1}$ for $\gamma_j \in N_1$ has its unnatural color, i.e., white, is motivated by the aim to let each node $v \in V_i^\kappa$ take its unnatural color if its corresponding input flipped to the opposite of the unnatural color of v. The satisfaction for the case that γ_j has the opposite color than $\lambda_j^{0,1}$ stems from the aim to let the flips back to the natural colors within the set V_i^κ only pass the node $\mu_i^{\kappa,2j}$ if γ_j has the same color as g_j. However, since g_j possibly has a degree of four in

G^h, it cannot be in $D(\varphi)$ for us to apply the Filtering Lemma (i.e., Lemma 3.5.21)—for this, v had to have a degree of at most three. Thus, we instead choose the formula to be satisfied if γ_j has the opposite color as $\lambda_j^{0,1}$ which, as we see later, will have the opposite color as g_j in the partitions in which $\mu_i^{\kappa,2j}$ has the same color as $\mu_i^{\kappa,2j-1}$.

Let $g_i \in V_{not}^C$, $\gamma_j \in N_2$ with $I(\gamma_j) = \gamma_k$ for $\gamma_k \in V_{not}^{C^h}$ and $\kappa \in \{0,1\}$. We hinder $\mu_i^{\kappa,2j}$ from flipping unless $\mu_i^{\kappa,2j-1}$ is white or γ_k is black or γ_k is white and γ_j is black. Moreover, we hinder $\mu_i^{\kappa,2j+3}$ unless $\mu_i^{\kappa,2j+2}$ is black or γ_k is white or γ_k is black and γ_j is white.

$$\begin{gathered}\varphi(\mu_i^{\kappa,2j}) = \overline{\mu_i^{\kappa,2j-1}} \vee \gamma_k \vee (\overline{\gamma_k} \wedge \gamma_j) \\ \text{for all } g_i \in V_{not}^C, \gamma_j \in N_2 \text{ with } I(\gamma_j) = \gamma_k \text{ for } \gamma_k \in V_{not}^{C^h}, \kappa \in \{0,1\},\end{gathered} \tag{3.5.29}$$

$$\begin{gathered}\varphi(\mu_i^{\kappa,2j+3}) = \mu_i^{\kappa,2j+2} \vee \overline{\gamma_k} \vee (\gamma_k \wedge \overline{\gamma_j}) \\ \text{for all } g_i \in V_{not}^C, \gamma_j \in N_2 \text{ with } I(\gamma_j) = \gamma_k \text{ for } \gamma_k \in V_{not}^{C^h}, \kappa \in \{0,1\}.\end{gathered} \tag{3.5.30}$$

Comment See comment for (3.5.24) and (3.5.27).

Let $g_i \in V_{not}^C$, $\gamma_j \in N_3$ and $\kappa \in \{0,1\}$. We hinder $\mu_i^{\kappa,2j}$ from flipping unless $\mu_i^{\kappa,2j-1}$ is white or γ_j has the color $c(\gamma_{j-2}) \vee c(\gamma_{j-3})$:

$$\begin{gathered}\varphi(\mu_i^{\kappa,2j}) = \overline{\mu_i^{\kappa,2j-1}} \vee (\overline{\gamma_j} \wedge \overline{\gamma_{j-2}} \wedge \overline{\gamma_{j-3}}) \vee (\gamma_j \wedge \gamma_{j-2}) \vee (\gamma_j \wedge \gamma_{j-3}) \\ \text{for all } g_i \in V_{not}^C, \gamma_j \in N_3, \kappa \in \{0,1\}.\end{gathered} \tag{3.5.31}$$

Comment As in the formulas (3.5.28) and (3.5.29) we let the formula of $\mu_i^{\kappa,2j}$ for $\gamma_j \in N_3$ be satisfied if its input node has its unnatural color, i.e., white. The remaining part of the formula is to let the flips back to the natural color within V_i^κ pass $\mu_i^{\kappa,2j}$ only if the input node of γ_j has its correct color with respect to the color of its input node, i.e., the NOR-node γ_{j-1}. However, since γ_{j-1} is a NOR-node, it cannot be in $D(\varphi)$ and therefore it cannot be a variable of $\varphi(\mu_i^{\kappa,2j})$. Instead, we make the flip of $\mu_i^{\kappa,2j}$ dependent on whether γ_j has the opposite color of γ_{j-1} under the assumption that γ_{j-1} has the correct color with respect to the colors of its corresponding inputs, i.e., if γ_j has the color $c(\gamma_{j-2}) \vee c(\gamma_{j-3})$.

Let $g_i \in V_{not}^C$, $\gamma_j \in N_4$ with $I(\gamma_j) = \gamma_k$ for $\gamma_k \in V_{not}^{C^h}$ and $\kappa \in \{0,1\}$. We hinder $\mu_i^{\kappa,2j}$ from flipping unless $\mu_i^{\kappa,2j-1}$ has its unnatural color, i.e., white, or γ_j has the opposite color of γ_k:

$$\begin{aligned}\varphi(\mu_i^{\kappa,2j}) &= \overline{\mu_i^{\kappa,2j-1}} \vee (\gamma_j \wedge \overline{\gamma_k}) \vee (\overline{\gamma_j} \wedge \gamma_k) \\ \text{for all } g_i \in V_{not}^C, \gamma_j \in N_4 \text{ with } I(\gamma_j) = \gamma_k \text{ for } \gamma_k \in V_{not}^{C^h}, \kappa \in \{0,1\}.&\end{aligned} \tag{3.5.32}$$

Comment Analogously to (3.5.28), (3.5.29) and (3.5.31) we design $\varphi(\mu_i^{\kappa,2j})$ for $\gamma_j \in N_4$ to be satisfied if its input node has its unnatural color, i.e., white. The remaining part of the formula is supposed to let the flips to the natural color within V_i^κ pass $\mu_i^{\kappa,2j}$ only if the input node of γ_j has the opposite color as its corresponding input node.

Finally, we hinder ρ_i for all $g_i \in V_{not}^C$ from flipping unless $\mu_j^{\kappa,2m}$ for each $g_j \in V_{not}^C$, $\kappa \in \{0,1\}$ has its natural color, i.e., white:

$$\varphi(\rho_i) = \bigwedge_{g_j \in V_{not}^C, \kappa \in \{0,1\}} \overline{\mu_j^{\kappa,2m}} \text{ for all } g_i \in V_{not}^C. \tag{3.5.33}$$

Comment The aim of the formula $\varphi(\rho_i)$ for $g_i \in V_{not}^C$ is as follows. Assume that node $g_k \in V_{not}^C$ flipped followed by flips of the nodes of V_k^2. Let $g_i \in V^C$ be the NOT-node of V^C chosen by h for the next flip of a NOT-node after the flip of g_k. When all gate-nodes of V^{C^h} took their correct colors after the flip of g_k and the nodes π_i, θ_i and ξ_i flipped, then all nodes of the sets V_j^κ for $\kappa \in \{0,1\}$, $g_j \in V_{not}^C$ are supposed to have their natural color again—as in the initial, recurring, partition. Therefore, we let ρ_i only flip if the nodes $\mu_j^{\kappa,2m}$ for all $\kappa \in \{0,1\}$, $g_j \in V_{not}^C$ have their natural colors again, i.e., white.

This finishes the description of φ. For an overview of the nodes of the formulas of φ see Table 3.3.

In the following, we consider whether the graph G^h, the partition P_0 and the function φ satisfy the conditions of the Filtering Lemma (i.e., Lemma 3.5.21) and show that the degrees of the nodes of $D(\varphi)$ is at most three. By assumption, all NOT-gates of the circuits C and C^h have fan-in one and fan-out at most two. The NOT-gates introduced in Figure 3.14 also have fan-in one and fan-out at most two. Furthermore, all nodes of Type I introduced in Figure 3.15 are of degree two—in particular, the nodes ρ_i which are in $D(\varphi)$. Due to the gates added in Figure 3.14 there might be NOT-gates $g_i \in C^1$ that have a fan-in of one and a fan-out of three—namely the NOT-gates $g_i \in C$ that have a fan-out of two in C. However, no node of V_{not}^C is in $D(\varphi)$. Thus, in G^h all nodes of $D(\varphi)$ have a degree of at most three.

Now we consider the influence of the nodes of $D(\varphi)$ and their happiness in P_0—the nodes of $D(\varphi)$ can be seen in the first and second column of Table 3.3. Node $\mu_i^{\kappa,j}$ for any $g_i \in V_{not}^C, \kappa \in \{0,1\}, 1 \leq j \leq 2m$ does not have influence on $H_{G^h}(\mu_i^{\kappa,j}) = \mu_i^{\kappa,j-1}$ and is happy in P_0 according to (R2iii). Node $\lambda_i^{\kappa,2}$ for any $g_i \in V_{not}^C, \kappa \in \{0,1\}$ has no influence on $H_{G^h}(\lambda_i^{\kappa,2}) = \lambda_i^{\kappa,1}$ and is happy in P_0 according to (R2ii). Similarly, node

$\lambda_i^{\kappa,1}$ for any $g_i \in V_{not}^C, \kappa \in \{0,1\}$ has no influence on $H_{G^h}(\lambda_i^{\kappa,2})$, i.e., node g_i if $\kappa = 0$ and $\lambda_i^{0,3}$ if $\kappa = 1$, and is happy according to, again, (R2ii). Each node $\gamma_j \in V_{not}^{C^h} \setminus N_1$ has no influence on $I(\gamma_j)$ according to the definition of G^{C^h} which constitutes C^h. The happiness of γ_j follows from (R3). Each node $\gamma_j \in N_1$ has no influence on $H_{G^h}(\gamma_j) = \lambda_j^{1,2}$ and is happy due to, again, (R3). Finally, node ρ_i for any $g_i \in V_{not}^C$ has no influence on ξ_i—this property is the reason for the existence of the nodes θ_i and ξ_i: If π_i and ρ_i were adjacent without the intermediate nodes θ_i and ξ_i, then ρ_i had influence on π_i. The happiness of ρ_i in P_0 follows from (R2i). No further nodes are in $D(\varphi)$.

Thus, G^h, P_0 and φ satisfy the conditions of the Filtering Lemma. We let $\boldsymbol{G^\varphi} = (V^\varphi, E^\varphi)$ be the graph and $\boldsymbol{R_0} \in \mathscr{P}(V^\varphi)$ be the partition guaranteed to be polynomial-time computable from G^h, P_0 and φ according to the Filtering Lemma.

4) Phases For the sake of readability, we introduce the following notations.

Definition 3.5.34. *For a partition $P \in \mathscr{P}(V^\varphi)$, we call P **recurring** if $P|_{V^h}$ is recurring. Let r be a sequence starting at (G^φ, R_0). We call r **alternating** if for each $v \in D(\varphi)$ with $u := H_{G^h}(v)$ we have $r|_{\{u,v\}} = (u,v,u,v,u,\ldots)$. If r is not alternating then we call it **irregular**. Furthermore, in a partition $R \in \mathscr{P}(V^\varphi)$ we call v **open** if $c_P(u) = c_P(v)$ and **closed** otherwise. Finally, for a partition $P \in \mathscr{P}(V^\varphi)$ we let $\boldsymbol{P^*} := P|_{V^C}$.*

Comment The main purpose of the denotation "alternating" is to encapsulate in a simple name for a condition that implies a condition of the Filtering Lemma (i.e., Lemma 3.5.21), in particular, the condition of (FL5) that for each node $u \in D(\varphi)$ there are no two flips of the $H_G(u)$ in the corresponding sequence without an intermediate flip of u. Similarly, the purpose of the denotations "open" and "closed" is to create a

Node	Nodes of formula	Conditions	
$\lambda_i^{\kappa,2}$	$\lambda_i^{\kappa,1}, \mu_i^{\kappa,2m}$	$g_i \in V_{not}^C$, $\kappa \in \{0,1\}$	
γ_j	$\mu_i^{\kappa,2j-1}$		$\gamma_j \in N_1 \cup N_3 \cup N_4$
γ_j	$\mu_i^{\kappa,2j-1}, \mu_i^{\kappa,2j+2}, \gamma_k$		$\gamma_j \in N_2, \gamma_k = I(\gamma_j)$
$\mu_i^{\kappa,2j}$	$\mu_i^{\kappa,2j-1}, \gamma_j, \lambda_j^{0,1}$		$\gamma_j \in N_1$
$\mu_i^{\kappa,2j}$	$\mu_i^{\kappa,2j-1}, \gamma_j, \gamma_k$		$\gamma_j \in N_2, \gamma_k = I(\gamma_j)$
$\mu_i^{\kappa,2j+3}$	$\mu_i^{\kappa,2j+2}, \gamma_j, \gamma_k$		
$\mu_i^{\kappa,2j}$	$\mu_i^{\kappa,2j-1}, \gamma_j, \gamma_k$		$\gamma_j \in N_4, \gamma_k = I(\gamma_j)$
$\mu_i^{\kappa,2j}$	$\mu_i^{\kappa,2j-1}, \gamma_j, \gamma_{j-2}, \gamma_{j-3}$		$\gamma_j \in N_3, \gamma_k = I(\gamma_j)$
ρ_i	$\mu_i^{\kappa,2m}$		

Table 3.3: Variables of the formulas of φ.

simple possibility to refer to the two different conditions for Lemma 3.5.21 (FL5)(i) and Lemma 3.5.21 (FL5)(ii).

Definition 3.5.35. *We let* $\sigma : \mathscr{P}(V^C) \to \{1, 2, \ldots, m\}$ *be the following partial function*

$$\boldsymbol{\sigma(P)} = \begin{cases} j, & \text{if } g_j \in V^C_{not} \text{ for } g_j = h(P) \\ j+1, & \text{if } g_j \in V^C_{nor} \text{ for } g_j = h(P). \end{cases}$$

Note that σ is partial since $h(P) = nil$ for some partitions $P \in \mathscr{P}(V^C)$.

Definition 3.5.36. *Let* $P, P' \in \mathscr{P}(V^h)$ *and* $Q \in \mathscr{P}(V^{C^h})$ *be such that* $c_P(g_j) = c_Q(\gamma_j)$ *for each* $g_j \in V^C_{not}$, $c_Q(g_j) \neq c_Q(I(\gamma_j))$ *for each NOT-gate* γ_j *of* C^h *whose input link is not an input link of* C^h, *and* $c_Q(\gamma_j) \neq (c_Q(I_1(\gamma_j)) \vee c_Q(I_2(\gamma_j)))$ *for all NOR-gates of* C^h. *For a node* $\gamma_j \in V^{C^h}$ *we call* $c_Q(\gamma_j)$ *the* P-***correct*** *color of* γ_j *and call* γ_j *itself* P-***correct*** *in partition* P' *if* $c_{P'}(\gamma_j) = c_Q(\gamma_j)$, *otherwise we call it* P-*incorrect in* P'.

Before turning towards the individual phases we characterize frequently used properties of recurring partitions of V^h.

Lemma 3.5.37. *Let* P *be a partition of* V^φ *such that* $P|_{V^h}$ *is recurring. Then* $c_P(\tau_i) \neq c_P(\pi_i)$ *for all* $g_i \in V^C_{not}$.

Proof. Due to (R2ii) we have $c_P(g_i) \neq c_P(\lambda_i^{1,2})$. Thus, (R3) implies $c_P(g_i) = c_P(\gamma_i)$ for all $g_i \in V^C_{not}$. Since in P the color of each node that represents a gate of C^h is correct with respect to the colors of the nodes representing its inputs in C^h, it follows from (R3) and (R4) that $c_P(\tau_j) = c_P(g_j)$ for $g_j = h(P^*)$ and $c_P(\tau_i) \neq c_P(g_i)$ for all $i \neq j$. □

Lemma 3.5.38. *Let* P *be a partition of* V^φ *such that* $P|_{V^h}$ *is recurring. Then the following two conditions are satisfied. First, all nodes of* $D(\varphi)$ *are closed in* P. *Second, if* $h(P^*) = nil$ *then all nodes of* $V^h \setminus D(\varphi)$ *are happy in* P *and if* $h(P^*) \neq nil$ *then* $h(P^*)$ *is unhappy in* P *and all other nodes of* $V^h \setminus D(\varphi)$ *are happy.*

Proof. At first, we consider the nodes of $V^1_I \setminus V^C_{not}$. According to (R2ii)–(R3) for each node $v \in V^1_I \setminus V^C_{not}$ with $u := H_{G^h}(v)$ we have $c_P(u) \neq c_P(v)$. No node of V^C_{not} is in $D(\varphi)$—see first and second column of Table 3.3. The only nodes of $V^h \setminus V^1$ that are in $D(\varphi)$ are the nodes ρ_i for $g_i \in V^C_{not}$ but these nodes are closed due to (R2i). Thus, all nodes of $D(\varphi)$ are closed in P.

Now we consider the remaining nodes of V^h. The NOR-nodes of V^h are happy in P according to (R1) and (R4). The nodes ρ_i, ξ_i and θ_i are happy for all $g_i \in V^C_{not}$ according to (R2i). The nodes π_i for $g_i \in V^C_{not}$ are happy since $c_P(\pi_i) \neq c_P(\theta_i)$ according to (R2i) and $c_P(\pi_i) \neq c_P(\tau_i)$ according to Lemma 3.5.37. Moreover, (R2i) also implies that each node $g_i \in V^C_{not}$ is happy in P if $h(P^*) = nil$. Finally, (R2i) implies that g_i is unhappy if $h(P^*) = g_i$ and that all $g_j \in V^C_{not}$ with $j \neq i$ are happy. □

Lemma 3.5.39. *Let $r = (x_1, \dots, x_q)$ for $q \in \mathbb{N}$ be a final sequence starting at (G^φ, R_0) and $0 \leq j \leq q$ be an index for which $R := R_j|_{V^h}$ is recurring. Then node π_i for any $g_i \in V^C_{not}$ is happy in each partition R_k for $j \leq k \leq q$ for which there is no flip of τ_i in r^k_{j+1}.*

Proof. Since R is recurring, Lemma 3.5.37 implies $c_R(\tau_i) \neq c_R(\pi_i)$ for all $g_i \in V^C_{not}$. Moreover, (R2i) implies $c_R(\theta_i) \neq c_R(\pi_i)$. Since θ_i is of Type I with $H_{G^\varphi}(\theta_i) = \pi_i$—note that $\theta_i \notin D(\varphi)$—there is no flip of θ_i prior to the first flip of π_i in r^q_{j+1}. However, π_i is of Type III and therefore happy as long as τ_i and θ_i do not flip. Thus, the claim follows. □

Now we continue to prove the Enforcing Theorem. At first, we consider the case that $h(P^C) = nil$. Then, by definition of P_0, we have $h(P^*_0) = nil$. Since P_0 is recurring, all nodes of $D(\varphi)$ are closed according to Lemma 3.5.38. Thus, Lemma 3.5.21 (FL5)(i) implies that no node of $D(\varphi)$ flips prior to the first flip of a node of $V^h \setminus D(\varphi)$. Then, also due to Lemma 3.5.38, no node flips in $s|_{V^C}$. For the case $h(P^C) \neq nil$ we use the following invariant.

Lemma 3.5.40. *Let $0 \leq j \leq q$ be such that R_j is recurring, r^j_1 is alternating, $g_i := h(R^*_j)$ for $g_i \in V^C_{not}$, $R' \in \mathscr{P}(V^C_{not})$ be the partition arising from R^*_j by flipping g_i. Then there is an index $j < k \leq q$ such that the following conditions are satisfied:*

i) R_k is recurring.

ii) r^k_1 is alternating.

iii) In r^k_{j+1}, node g_i flips exactly once and no other node of V^C_{not} flips.

iv) If $h(R') \in V^C_{nor}$ then, in r^k_{j+1}, node $h(R')$ flips exactly once and no other node of V^C_{nor} flips otherwise no node of V^C_{nor} flips in r^k_{j+1}.

Proof. When arguing about the flips, we make frequent use of the Filtering Lemma, in particular of Lemma 3.5.21 (FL5)(ii). For the sake of succinctness, we make the following convention.

Prerequisite For a node $v \in D(\varphi)$ and an index $j \leq i \leq k$ we say that v is **blocked** in R_i if we argue by Lemma 3.5.21 (FL5)(ii)(a) that $Nodes(\varphi(v)) <_{R_i} \{v\}$ and, similarly, we say that v is **pushed** in R_i if we argue by Lemma 3.5.21 (FL5)(ii)(b) that there is a flip of v in r^k_{i+1}.

To show that a node $v \in D(\varphi)$ is blocked, we have to show that v is open, that $u := H_{G^h}(v)$ does not flip prior to the first flip of v, that its corresponding formula is unsatisfied and that no variable of $\varphi(v)$ flipped after the flip of u that made v open. To show that v is pushed, we have to show that it is open, that $\varphi(v)$ is satisfied and that neither u nor any variable of $\varphi(v)$ flips prior to the first flip of v.

We divide the consideration of the flips of the nodes of V^h in fourteen phases P1–P14. The phases and their corresponding flips are illustrated in Table 3.4—the variable $\boldsymbol{p} \in \mathbb{N}$

in the fourth and fifth column and rows of index 10–13 is chosen such that $\gamma_p := \tau_{\sigma(R')}$. The first column of the table contains the enumeration of the phases. The second column contains flips of nodes that occur in any case in r, and the following columns contain nodes that flip if the condition in the second row of the corresponding column is satisfied. The horizontal lines that enclose the nodes from above and below are to determine the range of phases in which the flips of the corresponding nodes take place. If, for example, $h(R') \in V^C_{nor}$ then a flip of $h(R')$, as specified in the third column, happens within the phases two to ten. For each set of nodes that is enclosed by such lines there is an absolute order defined for the nodes. More concretely, the nodes representing the gates of C^h are absolutely ordered by their topological ordering and the nodes $\mu_i^{\kappa,j}$ for a given $g_i \in V^C_{not}$, $\kappa \in \{0,1\}$, are absolutely ordered according to the second superscript $0 \leq j \leq 2m$. The upper line marks the moment at which the first node of the enclosed set with respect to the corresponding order becomes unhappy and the lower line marks the moment at which the last node of the set flips at the latest.

We already point out that each node of $V^h \setminus D(\varphi)$ that becomes unhappy in r^k_{j+1} subsequently becomes happy only by its own flip. Similarly, a node of $D(\varphi)$ that becomes open within the phases subsequently becomes closed only by its own flip. Moreover, the nodes of the set V_i^κ for $g_i \in V^C_{not}$, $\kappa \in \{0,1\}$ flip according to their absolute order with respect to the second superscript and the nodes that represent the gates of C^h flip, with a single exception that is pointed out later, according to their topological order.

In the following, we keep track of the flips in $r|_{V^h}$ by considering the set that contains the unhappy nodes of $V^h \setminus D(\varphi)$ and the open nodes of $D(\varphi)$ and how the set changes during the phases. It satisfies to focus on this set of nodes due to the following four properties. First, a happy node of $V^h \setminus D(\varphi)$ can obviously not perform the next flip. Second, if a node $v \in V^h \setminus D(\varphi)$ is happy in a partition P and unhappy in the partition P' arising from P by flipping a node $w \in V^h$ then v is influenced by w—recall that all nodes of $V^h \setminus D(\varphi)$ are influenced by the same nodes in G^φ as in G^h. Third, for an index $j \leq i \leq k$ for which r^i_{j+1} is alternating, a closed node $v \in D(\varphi)$ can also not perform the next flip since Lemma 3.5.21 (FL5)(i) implies $\{H_{G^h}(v)\} <_{R_i} \{v\}$ in this case—the property that r^i_j is alternating can in each case be verified by means of Table 3.4 according to the Filtering Lemma. Fourth, if a node $v \in D(\varphi)$ is closed in P and open in P' then v is influenced by w in G^h. Thus, to keep track of the set of unhappy and open nodes after a flip of a node w we only need to consider the nodes on which w has influence in G^h.

P1 Since R_j is recurring, Lemma 3.5.38 implies that node g_i is the unique unhappy node of $V^h \setminus D(\varphi)$ and that each node of $D(\varphi)$ is closed. Since r^j_1 is alternating, the Filtering Lemma, in particular Lemma 3.5.21 (FL5)(i), implies that no node of $D(\varphi)$ flips prior to the first flip of g_i in r^q_{j+1}. Therefore, g_i flips in r^q_{j+1} and no other node of V^h flips prior to the first flip of g_i. Node g_i has influence on $\lambda_i^{0,1}$ and at most two nodes of V^C.

P2 The flip of g_i in P1 makes $\lambda_i^{0,1}$ unhappy since g_i and $\lambda_i^{0,1}$ have the same color after the flip of g_i according to (R2ii).

P	Flips in case of				
		$h(R') \in V_{nor}^{C}$	$c_{R_j}(g_i) = 0$	$c_{R_j}(g_i) = 1$	$h(R') \neq nil$
1	g_i				
2	$\lambda_i^{0,1}$	$h(R')$			
3			$\mu_i^{0,\omega}$ $\forall 0 \leq \omega \leq 2m$		
4	$\lambda_i^{0,2}$				
5	$\lambda_i^{0,3}$		$\mu_i^{0,\omega}$ $\forall 0 \leq \omega < 2i$		
6	$\lambda_i^{1,1}$				
7				$\mu_i^{1,\omega}$ $\forall 0 \leq \omega \leq 2m$	
8	$\lambda_i^{1,2}$				
9				$\mu_i^{1,\omega}$ $\forall 0 \leq \omega < 2i$	
10	γ_ω for $1 \leq \omega \leq m$ and γ_ω R'-incorrect		$\mu_i^{0,\omega}$ $\forall 2i \leq \omega < 2p$	$\mu_i^{1,\omega}$ $\forall 2i \leq \omega < 2p$	
11			$\mu_i^{0,\omega}$ $\forall 2p \leq \omega \leq 2m$	$\mu_i^{1,\omega}$ $\forall 2p \leq \omega \leq 2m$	$\pi_{\sigma(R')}$
12					$\theta_{\sigma(R')}$
13					$\xi_{\sigma(R')}$
14					$\rho_{\sigma(R')}$

Table 3.4: Flips of the corresponding phases.

If g_i is not an output gate of C then g_i has influence on further nodes. There are two possible cases for further influence of g_i. First, g_i is an input of a NOR-gate $g_{i'}$ for $1 \leq i' \leq n$ in C. Then g_i has influence on $g_{i'}$ in G^h. According to our assumption that if $g_{i'}$ is the unique unhappy NOR-node of a given partition then h returns $g_{i'}$—recall that all NOR-nodes are happy in R_j according to (R4). Consequently, $g_{i'}$ is unhappy after the flip of g_i in P1 if and only if $h(R') = g_{i'}$. Thus, if $g_{i'}$ is unhappy after the flip of g_i then $h(R') = g_{i'}$ and $g_{i'}$ can flip in P2—see third column in Table 3.4. The unique node on which $g_{i'}$ has influence is $\pi_{i'+1}$ and this node remains happy as long as $\tau_{i'+1}$ does not flip due to Lemma 3.5.39. Second, g_i is input of a NOT-gate $g_{i'}$ for $1 \leq i' \leq n$ in C. Then g_i has influence on $\pi_{i'}$ but $\pi_{i'}$ remains happy as long as $\tau_{i'}$ does not flip due to, again, Lemma 3.5.39. Thus, there is a flip of $\lambda_i^{0,1}$ in r_{j+1}^k after the flip of g_i. Node $\lambda_i^{0,1}$ has influence on $\lambda_i^{0,2} \in D(\varphi)$ and on $\mu_i^{0,0} \notin D(\varphi)$.

P3 After the flip of $\lambda_i^{0,1}$ in P2 node $\lambda_i^{0,2}$ is open. We now distinguish between the two possible cases for the color of g_i in R_j.

Assume first $c_{R_j}(g_i) = 0$—see fourth column of Table 3.4. Then g_i flipped to black in P1 and $\lambda_i^{0,1}$ to white in P2. Then $\varphi(\lambda_i^{0,2})$ is unsatisfied according to (3.5.24) and therefore $\lambda_i^{0,2}$ is blocked as long as neither $\lambda_i^{0,1}$ nor $\mu_i^{0,2m}$ flips. The other node on which $\lambda_i^{0,1}$ has influence is $\mu_i^{0,0}$. Since both $\lambda_i^{0,1}$ and $\lambda_i^{0,2}$ are white, node $\mu_i^{0,0}$ is unhappy and will therefore flip to its unnatural color, i.e., black. According to the equations (3.5.28)–(3.5.32) for each node $\mu_i^{0,\omega} \in D(\varphi)$ with $1 \leq \omega \leq 2m$ the formula $\varphi(\mu_i^{0,\omega})$ is satisfied if $\mu_i^{0,\omega-1}$ has its unnatural value. Thus, it follows by induction on ω that the nodes $\mu_i^{0,\omega}$ for $1 \leq \omega \leq 2m$ are consecutively pushed in ascending order in ω and flip to their unnatural colors. The flip of $\mu_i^{0,2m}$ to the black color implies $\varphi(\lambda_i^{0,2})$ is satisfied according to (3.5.24). Since each node of V_i^0 flips to its unnatural color prior to the flip of its corresponding input node, the sequence r_{j+1}^q does not become irregular by a flip of a node of V_i^0 in P3.

Now assume $c_{R_j}(g_i) = 1$—see fifth column of Table 3.4. Then g_i flipped to white in P1 and $\lambda_i^{0,1}$ to black in P2. Since $\lambda_i^{0,2}$ is black and $\mu_i^{0,0}$ is white, node $\mu_i^{0,0}$ is still happy. However, since $\lambda_i^{0,1}$ is black, we have, as in the case $c_{R_j}(g_i) = 0$, the formula $\varphi(\lambda_i^{0,2})$ is satisfied according to (3.5.24).

P4 Since $\lambda_i^{0,2}$ is open after the flip of $\lambda_i^{0,1}$ in P2 and $\varphi(\lambda_i^{0,2})$ is satisfied as shown in P3, node $\lambda_i^{0,2}$ is pushed and flips therefore. Node $\lambda_i^{0,2}$ has influence on $\lambda_i^{0,3}$ and $\mu_i^{0,0}$.

P5 The flip of $\lambda_i^{0,2}$ in P4 makes $\lambda_i^{0,3}$ unhappy and it makes $\mu_i^{0,0}$ unhappy if $\mu_i^{0,0}$ flipped to its unnatural color in P3 which is true if and only if $c_{R_j}(g_i) = 0$—see Table 3.4. In the following, we consider the possible flips of the two nodes $\lambda_i^{0,3}$ and $\mu_i^{0,0}$.

At first, we consider $\lambda_i^{0,3}$. The only node that has influence on $\lambda_i^{0,3}$ is $\lambda_i^{0,2}$. Thus, $\lambda_i^{0,3}$ remains unhappy as long as neither itself nor $\lambda_i^{0,2}$ flips. Node $\mu_i^{0,0}$ is a node of V_i^0 and the nodes of V_i^0 have in G^h no influence on nodes outside of V_i^0. Thus, there is a flip of $\lambda_i^{0,3}$. Node $\lambda_i^{0,3}$ has influence on $\lambda_i^{1,1}$ and the flip of $\lambda_i^{0,3}$ makes $\lambda_i^{1,1}$ unhappy.

Now we consider the possible flips initiated by the unhappiness of $\mu_i^{0,0}$ in case of $c_{R_j}(g_i) = 0$—see fourth column of Table 3.4. All nodes of $V_i^0 \cap D(\varphi)$ are closed and all nodes of $V_i^0 \setminus (D(\varphi) \cup \{\mu_i^{0,0}\})$ are happy after the flip of $\lambda_i^{0,2}$ in P4. Due to Lemma 3.5.21 (FL5)(i) no node of $V_i^0 \cap D(\varphi)$ flips back to its natural color before its corresponding input node in G^h flips back to its natural color. Additionally, no node of $V_i^0 \setminus (D(\varphi) \cup \{\mu_i^{0,0}\})$ flips back to its natural color before its corresponding input node flips back to its natural color. The formula $\varphi(\mu_i^{0,2i})$ is not satisfied according to (3.5.28) as long as γ_i has the same color as $\lambda_i^{0,1}$—recall that γ_i and $\lambda_i^{0,1}$ have opposite colors in R_j due to (R2ii) and (R3) and after the flip of $\lambda_i^{0,1}$ in P2 therefore the same color. Thus, even if there is a flip of the input node of $\mu_i^{0,2i}$ in G^h, namely $\mu_i^{0,2i-1}$, node $\mu_i^{0,2i}$ is blocked.

P6 The flip of $\lambda_i^{0,3}$ in P5 makes $\lambda_i^{1,1}$ unhappy. As shown in P5 none of the nodes of $\mu_i^{0,\omega}$ for $0 \leq \omega \leq 2m$ has influence on any node outside of V_i^0. Thus, there is a flip of $\lambda_i^{1,1}$. Node $\lambda_i^{1,1}$ has influence on $\mu_i^{1,0}$ and $\lambda_i^{1,2}$.

P7 If $c_{R_j}(g_i) = 1$ then $\lambda_i^{0,3}$ flipped to white in P5. Then, as for the nodes $\mu_i^{0,\omega}$ for $0 \leq \omega \leq 2m$ and $\lambda_i^{0,2}$ in P3, it follows that the nodes $\mu_i^{1,\omega}$ for $0 \leq \omega \leq 2m$ flip consecutively in ascending order in ω and $\lambda_i^{1,2}$ does not flip as long as $\mu_i^{1,2m}$ is white.

P8 Analogously to $\lambda_i^{0,2}$ in P4 it follows that $\lambda_i^{1,2}$ flips. Node $\lambda_i^{1,2}$ has influence on $\gamma_i \in D(\varphi)$ and on $\mu_i^{1,0} \notin D(\varphi)$.

P9 Let $\kappa := c_{R_j}(g_i)$. The flip of $\lambda_i^{1,2}$ in P8 makes γ_i open—recall that in the recurring partition R_j node γ_i is closed and neither $\lambda_i^{1,2}$ nor γ_i flip in P1–P7. In the following, we show that the nodes $\mu_i^{\kappa,j}$ for $0 \leq j < 2i$ flip back to their natural colors and that their flips are finished before node γ_i takes the same color to which node g_i flipped in P1.

According to (3.5.26) the formula $\varphi(\gamma_i)$ is unsatisfied if $\mu_i^{1,2i-1}$ has its unnatural color, i.e., white. None of the nodes of V_i^κ has influence on $\lambda_i^{1,2}$ in G^h. Thus, γ_i is blocked as long as $\mu_i^{\kappa,2i-1}$ has its unnatural color. The nodes $\lambda_i^{0,3}$, $\lambda_i^{1,1}$ and $\lambda_i^{1,2}$ have no influence on the nodes of V_i^0 in G^h and neither of them is a variable of a formula $\varphi(v)$ for any $v \in V_i^0$—see Table 3.3. Thus, in the case $\kappa = 0$ the flips of the nodes $\lambda_i^{0,3}$, $\lambda_i^{1,1}$ and $\lambda_i^{1,2}$ in P5–P8 do not affect the happiness or the openness of any of the nodes of V_i^0.

We show by induction on ω that the nodes $\mu_i^{\kappa,\omega}$ for $0 \leq \omega < 2i$ flip back to their natural colors after the flip of $\lambda_i^{\kappa,2}$. As induction basis, note that $\mu_i^{\kappa,0}$ became unhappy after the flip of $\lambda_i^{\kappa,2}$ and that it remains unhappy as long as neither $\lambda_i^{\kappa,1}$ nor $\lambda_i^{\kappa,2}$ flips. In case of $\kappa = 1$ neither of the two nodes flips prior to the first flip of $\mu_i^{0,0}$ since there is no node different from $\mu_i^{\kappa,0}$ that is influenced by $\lambda_i^{\kappa,2}$ and can flip prior to a flip of $\mu_i^{\kappa,0}$—recall that γ_i is blocked as long as $\mu_i^{\kappa,2i-1}$ that has its unnatural color. In case of $\kappa = 0$, the nodes $\lambda_i^{0,3}$, $\lambda_i^{1,1}$ and $\lambda_i^{1,1}$ have no influence on $\lambda_i^{0,1}$ nor $\lambda_i^{0,2}$ and do therefore not affect the happiness of $\mu_i^{0,0}$. Thus, there is a flip of $\mu_i^{0,0}$ after the flip of $\lambda_i^{\kappa,2}$.

The induction hypothesis assumes for an arbitrary $0 \leq \omega < 2i$ that the nodes $\mu_i^{\kappa,\omega'}$ for $0 \leq \omega' < \omega$ flip back to their natural colors after the flip of $\lambda_i^{\kappa,2}$. If $\mu_i^{\kappa,\omega} \in D(\varphi)$, then it is pushed since $\varphi(\mu_i^{\kappa,\omega})$ is satisfied according to (3.5.28)—recall that in the recurring partition R_j node γ_ω has the opposite color as $\lambda_\omega^{0,1}$ due to (R2ii) and (R3) and none of these two nodes flips in P1–P8.

P10 Let $\kappa \in \{0,1\} = c_{R_j}(g_i)$. The flip of $\lambda_i^{1,2}$ in P8 made γ_i open and the flip of $\mu_i^{\kappa,2i-1}$ that took place in P9 at the latest made $\mu_i^{\kappa,2i}$ open. In the following, we show:

- For each $2i \leq \omega < 2p$ node $\mu_i^{\kappa,\omega}$ flips back to its natural color.
- For each $\gamma_\omega \in V^{C^h}$ node γ_ω flips exactly once if it is R'-incorrect in R_j and does not flip otherwise.

The claim is proven via induction on ω. As induction basis, we show that γ_i flips and that the nodes $\mu_i^{\kappa,2i}$ and $\mu_i^{\kappa,2i+1}$ flip back to their natural colors. Notice first that the nodes of V_i^κ only have influence in G^h on nodes of the same set, i.e., V_i^κ itself. The formula $\varphi(\mu_i^{\kappa,2i})$ is not satisfied according to (3.5.28) since γ_i has the same color as $\lambda_i^{0,1}$. Thus, $\mu_i^{\kappa,2i}$ is blocked. On the other hand, the formula $\varphi(\gamma_i)$ is satisfied due to (3.5.26). Consequently, node γ_i is pushed. By assumption, each node $\gamma_{i'}$ to which γ_i is an input in C^h is a NOT-node. According to (3.5.26) and (3.5.27) node $\mu_i^{\kappa,2i'-1}$ must have its natural value for $\varphi(\gamma_{i'})$ to be satisfied. Therefore, $\gamma_{i'}$ is either closed or blocked. The formula $\varphi(\mu_i^{\kappa,2i})$ is satisfied according to (3.5.28) since after the flip of γ_i, node γ_i has the opposite color as $\lambda_i^{0,1}$. Hence, $\mu_i^{\kappa,2i}$ is pushed. After the flip of $\mu_i^{\kappa,2i}$, node $\mu_i^{\kappa,2i+1}$—which is the only node on which $\mu_i^{\kappa,2i}$ has influence in G^h—becomes unhappy. Moreover, node $\mu_i^{\kappa,2i}$ is not a variable of the formulas of any gate for which γ_i is an input. Consequently, node $\mu_i^{\kappa,2i+1}$ flips.

As induction hypothesis we assume for γ_ω with $i \leq \omega \leq p$ that

IH1) Case $\gamma_\omega \in N_1 \cup N_2 \cup N_4$:

IH1i) $I(\gamma_\omega)$ flipped once if γ_ω was R'-incorrect in R_j and the inputs of $I(\gamma_\omega)$, should they be in V^{C^h}, flipped once if they were R'-incorrect in R_j and did not flip otherwise.

IH1ii) $\mu_i^{\kappa,2\omega-1}$ flipped back to its natural color.

IH2) Case $\gamma_\omega \in V_{nor}^{C^h}$:

IH2i) The input nodes of the input nodes of γ_ω flipped once if they were R'-correct and did not flip otherwise.

IH2ii) $\mu_i^{\kappa,2\omega-5}$ flipped back to its natural color.

The induction step is divided into four different cases which are induced by four disjoint sets in which γ_ω can be an element, namely $\gamma_\omega \in N_1$ for $\omega > i$, $\gamma_\omega \in N_2$, $\gamma_\omega \in N_4$ with $\omega \leq i$ and $\gamma_\omega \in V_{nor}^{C^h}$—see Table 3.5.

Case	Condition of formula	Consideration of (possible) flips	
		Gate node	Node of V_i^κ
$\gamma_\omega \in N_1$	$\omega > i$	γ_ω	$\mu_i^{\kappa,2\omega}, \mu_i^{\kappa,2\omega+1}$
$\gamma_\omega \in N_4$	$\omega < p$ and γ_ω has no influence on a NOR-node	γ_ω	$\mu_i^{\kappa,2\omega}, \mu_i^{\kappa,2\omega+1}$
	$\omega = p$ or γ_ω has influence on a NOR-node	γ_ω	$\mu_i^{\kappa,2\omega}$
$\gamma_\omega \in N_2$	$\omega > i$	γ_ω	$\mu_i^{\kappa,2\omega}, \mu_i^{\kappa,2\omega+1}, \mu_i^{\kappa,2\omega+3}$
$\gamma_\omega \in V_{nor}^{C^h}$		$\gamma_\omega, \gamma_{\omega+1}$	$\mu_i^{\kappa,2\omega}, \ldots, \mu_i^{\kappa,2\omega+3}$

Table 3.5: Distribution of the consideration of the (possible) flips among the cases.

The case $\gamma_\omega \in N_1$ for $\omega = i$ is already covered in the induction basis and for $\omega < i$ there is no flip of γ_ω since γ_ω is R'-correct in this case and the flips of the corresponding nodes $\mu_i^{\kappa,2\omega}$ and $\mu_i^{\kappa,2\omega+1}$ are covered in P5–P9 for $c_{R_j}(g_i) = 0$ and in P9 for $c_{R_j}(g_i) = 1$. In case of $\gamma_\omega \in N_4$ for $\omega > p$, node γ_ω is R'-correct in R' and does not flip since its input is also R'-correct in R_j and does not flip due to the induction hypothesis. The flips of the nodes $\mu_i^{\kappa,\omega}$ for $2p \leq \omega \leq 2m$—these nodes correspond to the gates γ_ω for $\omega > p$—are covered in P11–P13. The possible flips of the nodes of N_3 and the flips of the corresponding nodes of V_i^κ are covered in the case for $\gamma_\omega \in V_{nor}^{C^h}$. Due to dependencies among the cases, we consider the cases in the order $\gamma_\omega \in N_1$ for $\omega > i$, $\gamma_\omega \in N_4$ for $\omega \leq p$, $\gamma_\omega \in N_2$ and then $\gamma_\omega \in V_{nor}^{C^h}$. For an overview over the relevant nodes for the last three of these cases see Figure 3.18. In each of the four cases we show that the corresponding gate nodes flip once if they are R'-incorrect in R_j and that they do not flip otherwise. The considerations of the flips of the nodes $\mu_i^{\kappa,\alpha}$ for $0 \leq \alpha \leq 2m$ are distributed among the cases as follows. In the case $\gamma_\omega \in N_1$ and in the case $\gamma_\omega \in N_4$ for $\omega < p$ where γ_ω does not have influence on a NOR-node we show that the nodes $\mu_i^{\kappa,2\omega}$ and $\mu_i^{\kappa,2\omega+1}$ flip. In the case of $\gamma_\omega \in N_4$ where γ_ω has influence on a NOR-node,

we just show that $\mu_i^{\kappa,2\omega}$ flips—the flip of the node $\mu_i^{\kappa,2\omega+1}$ is not covered by this case. However, in the case for $\gamma_\omega \in N_2$, we show that $\mu_i^{\kappa,2\omega}$, $\mu_i^{\kappa,2\omega+1}$ and $\mu_i^{\kappa,2\omega+3}$ flip—the node $\mu_i^{\kappa,2\omega+3}$ is the node that was not covered in the previous case. Finally, in the case of $\gamma_\omega \in V_{nor}^{C^h}$ we show that the nodes $\mu_i^{\kappa,2\omega},\dots,\mu_i^{\kappa,2\omega+3}$ flip.

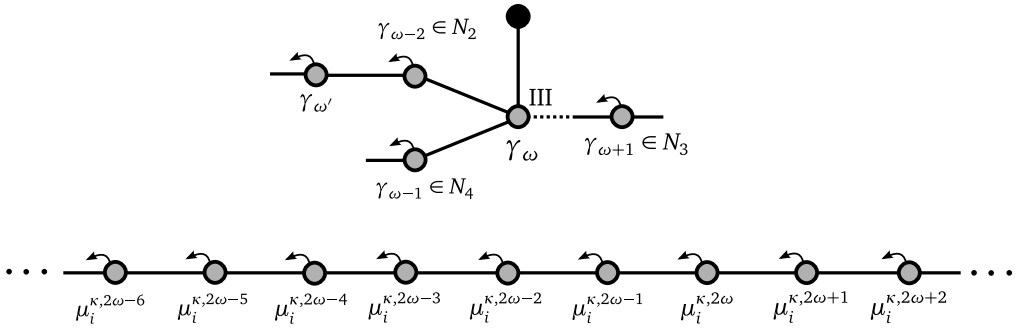


Figure 3.18: The relevant nodes for the induction induction step of P10.

- Case $\gamma_\omega \in N_1$ for $\omega > i$:

  Since R_j is recurring and neither g_ω nor γ_ω flipped in P1–P9 node γ_ω is R'-correct according to (R2ii) and (R3). Since $I(\gamma_\omega) = \lambda_\omega^{1,2}$ also did not flip in P1–P9 node γ_ω is closed and has the opposite color as $\lambda_\omega^{1,2}$ according to, again, (R2ii) and (R3). Thus, the formula $\varphi(\mu_i^{\kappa,2\omega})$ is satisfied due to (3.5.28). According to (IH1ii) node $\mu_i^{\kappa,2\omega-1}$ flipped back to its natural color. Consequently, $\mu_i^{\kappa,2\omega}$ is open and pushed. After its flip, node $\mu_i^{\kappa,2\omega+1}$—which is the only node on which $\mu_i^{\kappa,2\omega}$ has influence in G^h—becomes unhappy. Consequently, node $\mu_i^{\kappa,2\omega+1}$ flips.

- Case $\gamma_\omega \in N_4$ with $\omega \leq i$:

  We distinguish between the two possible cases for the R'-correctness of γ_ω and show that γ_ω flips if it is R'-incorrect and that it does not flip otherwise. At first, we consider the case that γ_ω is R'-incorrect. Then the input gate of γ_ω in C^h flipped to its R'-correct color according to (IH1i) and therefore γ_ω is open and has the same color as its input node. Since it has the same color as its input node, the formula $\varphi(\mu_i^{\kappa,2\omega})$ is not satisfied according to (3.5.32) which implies that $\mu_i^{\kappa,2\omega}$ is blocked. Thus, node γ_ω is pushed.

  Now we consider the case that γ_ω is R'-correct. Then the input gate of γ_ω in C^h did not flip according to (IH1i) which implies that γ_ω is closed. Thus, Lemma 3.5.21 (FL5)(i) implies that there is no flip of γ_ω prior to the first flip of $\mu_i^{\kappa,2\omega}$.

  In both cases we get a partition in which γ_ω has the opposite color as its input. Now we show that when γ_ω has the opposite color as its input node then the node $\mu_i^{\kappa,2\omega}$ flips. The formula $\varphi(\mu_i^{\kappa,2\omega})$ is satisfied according to

(3.5.32) and $\mu_i^{\kappa,2\omega}$ is open due to the flip of $\mu_i^{\kappa,2\omega-1}$ to its natural color. In the following, we show that the nodes on which γ_ω has influence—these nodes might have become unhappy or, if they are in $D(\varphi)$, open by a flip of γ_ω—or the flips of the nodes these nodes themselves have influence on, neither affect the openness of $\mu_i^{\kappa,2\omega}$ nor the satisfaction of $\varphi(\mu_i^{\kappa,2\omega})$. Then, it follows that $\mu_i^{\kappa,2\omega}$ is pushed.

If γ_ω is an input to a NOT-gate $\gamma_{\omega'}$ in C^h then the formula $\varphi(\gamma_{\omega'})$ is not satisfied according to (3.5.27) since the nodes $\mu_i^{\kappa,\alpha}$ for $2\omega' \leq \alpha \leq 2m$ have their unnatural values. Hence, $\gamma_{\omega'}$ is blocked if γ_ω was R'-incorrect and flipped or it is closed if γ_k was R'-correct and did therefore not flip. If γ_k is an input to a NOR-gate γ_{k+1} in C^h then the formula $\varphi(\gamma_{\omega+2})$ is not satisfied according to (3.5.26) since node $\mu_i^{\kappa,2(\omega+1)-1}$ has its unnatural value. Thus, $\gamma_{\omega+2}$ is, depending on whether $\gamma_{\omega+1}$ flipped after the flip of γ_ω, either blocked or closed. In both cases, there is no flip of $\gamma_{\omega+2}$ prior to the first flip of $\mu_i^{\kappa,2\omega}$. Finally, if γ_ω for $\omega < p$ is an output gate of C^h then γ_ω has influence on π_α for some $g_\alpha \in V_{not}^C$. Node π_α has only influence on θ_α in G^h. The sequence of unique influences continues with ξ_α and ρ_α. None of the nodes π_α, θ_α, ξ_α and ρ_α is a variable in a formula of any other node of $D(\varphi)$. Moreover, the formula $\varphi(\rho_\alpha)$ is unsatisfied according to (3.5.33). Thus, ρ_α is blocked if its input node in G^h flips prior to the first flip of $\mu_i^{\kappa,2\omega}$. Thus, in each of the considered cases it follows that $\mu_i^{\kappa,2\omega}$ is pushed.

It remains to show that $\mu_i^{\kappa,2\omega+1}$ flips after the flip of $\mu_i^{\kappa,2\omega}$ in the case that γ_ω has no influence on a NOR-node and $\omega < p$. But this follows simply from the fact that after the flip of $\mu_i^{\kappa,2\omega}$, node $\mu_i^{\kappa,2\omega+1}$ is unhappy and node $\mu_i^{\kappa,2\omega}$ has influence only on $\mu_i^{\kappa,2\omega+1}$ in G^h and is not a variable of the formula of any node in $D(\varphi)$.

- Case $\gamma_\omega \in N_2$: We distinguish between the two possible cases for the color of γ_ω. At first, we consider the case that γ_ω is white. In this case, the formula $\varphi(\gamma_\omega)$ is according to (3.5.27) and (3.5.26) satisfied if and only if the formula $\varphi(\gamma_\omega)$ for the case $\gamma_\omega \in N_4$ is satisfied. Thus, the flips of the nodes γ_ω and $\mu_i^{\kappa,2\omega}$ follow as in the case for $\gamma_\omega \in N_4$ where γ_ω is an input to a NOR-node. Node $\mu_i^{\kappa,2\omega}$ has influence only on $\mu_i^{\kappa,2\omega+1}$ and $\mu_i^{\kappa,2\omega+1}$ becomes unhappy by the flip of $\mu_i^{\kappa,2\omega}$. The node γ_ω has only influence on $\gamma_{\omega+2}$, which itself has influence only on $\gamma_{\omega+3}$. However, the formula $\varphi(\gamma_{\omega+3})$ is not satisfied according to (3.5.26) since node $\mu_i^{\kappa,2\omega+5}$ has its unnatural color and therefore $\gamma_{\omega+3}$ is, depending on whether the NOR-node $\gamma_{\omega+2}$ flipped after the flip of γ_ω, either closed or blocked. In both cases, there is no flip of $\gamma_{\omega+3}$ prior to the flip of $\mu_i^{\kappa,2\omega+1}$ to its natural color. Thus, node $\mu_i^{\kappa,2\omega+1}$ flips back to its natural color.

  Then, according to the case for $\gamma_\omega \in N_4$, node $\mu_i^{\kappa,2\omega+2}$ flips and node $\gamma_{\omega+1}$ flips if and only if it is R'-incorrect. A flip of $\gamma_{\omega+1}$ may make the NOR-node

$\gamma_{\omega+2}$ unhappy, but this can only be the case if the flip of γ_ω, should γ_ω have been R'-incorrect, did not make $\gamma_{\omega+2}$ unhappy since the happiness of $\gamma_{\omega+2}$ is independent of the color of $\gamma_{\omega+1}$ if γ_ω is black. Thus, node $\gamma_{\omega+2}$ does not become unhappy for the second time after the flip of γ_ω, should it have been R'-incorrect. Hence, γ_ω does flip twice and does therefore not make r_1^k irregular. Consequently, node $\gamma_{\omega+3}$ is, depending on whether $\gamma_{\omega+2}$ flipped, either closed or blocked—recall that the formula $\varphi(\gamma_{\omega+3})$ is still unsatisfied since $\mu_i^{\kappa,2\omega+5}$ has its unnatural value. However, the formula $\varphi(\mu_i^{\kappa,2\omega+3})$ is satisfied according to (3.5.30) since γ_ω has the opposite color as its input. Therefore, $\mu_i^{\kappa,2\omega+3}$ is pushed.

Now we consider the case that γ_ω is black. Then the formula $\varphi(\gamma_\omega)$ is not satisfied according to (3.5.27). Thus, γ_ω is blocked. On the other hand, the formula $\varphi(\mu_i^{\kappa,2\omega})$ is satisfied according to (3.5.29) and therefore node $\mu_i^{\kappa,2\omega}$ is pushed. Node $\mu_i^{\kappa,2\omega}$ only has influence on $\mu_i^{\kappa,2\omega+1}$ in G^h which becomes unhappy by the flip of $\mu_i^{\kappa,2\omega}$. Since node $\mu_i^{\kappa,2\omega}$ is not a variable of the formula $\varphi(\gamma_\omega)$, γ_ω is still blocked and therefore there is a flip of $\mu_i^{\kappa,2\omega+1}$ after the flip of $\mu_i^{\kappa,2\omega}$. Then, according to the case for $\gamma_\omega \in N_4$ node $\gamma_{\omega+1}$ flips if and only if it is R'-incorrect and the node $\mu_i^{\kappa,2\omega+2}$ also flips. The flip of $\gamma_{\omega+1}$ cannot make the NOR-node $\gamma_{\omega+2}$ unhappy since γ_ω is black and $\gamma_{\omega+2}$ is therefore white according to (R4). After the flip of $\mu_i^{\kappa,2\omega+2}$ to its natural color node $\mu_i^{\kappa,2\omega+3}$ is open.

Now we distinguish between the two possible cases for the R'-correctness of γ_ω in R_j. If γ_ω was R'-correct then it is now closed. Then $\varphi(\mu_i^{\kappa,2\omega+3})$ is satisfied according to (3.5.30) and therefore $\mu_i^{\kappa,2\omega+3}$ is pushed. Now consider the case that γ_ω was R'-incorrect. Then γ_ω is now open and therefore $\varphi(\mu_i^{\kappa,2\omega+3})$ is not satisfied according to (3.5.30). Consequently, $\mu_i^{\kappa,2\omega+3}$ is blocked. On the other hand, the formula $\varphi(\gamma_\omega)$ is satisfied after the flip of $\mu_i^{\kappa,2\omega+2}$ to its natural color. Thus, node γ_ω is pushed. Its flip may make the NOR-node $\gamma_{\omega+2}$ unhappy, which in turn may also flip, but then node $\gamma_{\omega+3}$ is blocked according to (3.5.26) since $\mu_i^{\kappa,2\omega+3}$ has its unnatural color. After the flip of γ_ω the formula $\varphi(\mu_i^{\kappa,2\omega+3})$ is satisfied according to (3.5.30), which implies that $\mu_i^{\kappa,2\omega+3}$ is pushed.

- Case $\gamma_\omega \in V_{nor}^{C^h}$: According to the induction hypothesis of this case—see (IH2)—the node $\mu_i^{\kappa,2\omega-5}$ flipped back to its natural color and the inputs of the inputs of γ_ω have their R'-correct colors. According to the cases for $\gamma_\omega \in N_2$ and $\gamma_\omega \in N_4$ the nodes $\mu_i^{\kappa,2\omega-4},\dots,\mu_i^{\kappa,2\omega-1}$ flip after the flip of $\mu_i^{\kappa,2\omega-5}$ to their natural color. In the following we distinguish between the two possible cases for the R'-correctness of γ_ω in R_j.

  Assume first that γ_ω is R'-correct in R_j. For this case, we first show that γ_ω does not become unhappy by the flips of its input nodes. For the flips of the input nodes, we have three possibilities: Either none of them flips, one

of them flips or both of them flip. If none of them flips then γ_ω obviously remains happy. If one of them flips then its flip does not make γ_ω unhappy since otherwise γ_ω was R'-incorrect in R_j, which is a contradiction. Now consider the case that both inputs of γ_ω flip. Then the inputs of γ_ω have different colors in R_j since otherwise γ_ω would, again, have been R'-incorrect in R_j. In case of $c_{R_j}(\gamma_{\omega-2}, \gamma_{\omega-1}) = (0,1)$ the flip of $\gamma_{\omega-2}$—which is according to the cases for $\gamma_\omega \in N_2$ and $\gamma_\omega \in N_4$ the first of the two input nodes of γ_ω that flips—to the black color does not make γ_ω unhappy since $c_{R_j}(\gamma_\omega) = 0$ in this case. Then $\gamma_{\omega-1}$ flips to white whereafter γ_ω is still happy. Now consider the case $c_{R_j}(\gamma_{\omega-2}, \gamma_{\omega-1}) = (1,0)$. Then, according to, again, the cases for $\gamma_\omega \in N_2$ and $\gamma_\omega \in N_4$, node $\gamma_{\omega-1}$ first flips to black and then $\gamma_{\omega-2}$ flips to white. Analogously to the case $c(\gamma_{\omega-2}, \gamma_{\omega-1}) = (0,1)$ it follows that neither of the two flips makes γ_ω unhappy. Thus, in neither case there is a flip of the input nodes of γ_ω that makes γ_ω unhappy and therefore γ_ω does not flip.

It remains to show that the nodes $\mu_i^{\kappa,2\omega}, \ldots, \mu_i^{\kappa,2\omega+3}$ flip back to their natural colors. The flip of node $\mu_i^{\kappa,2\omega-1}$ back to its natural color makes the node $\mu_i^{\kappa,2\omega}$—which is the only node on which $\mu_i^{\kappa,2\omega-1}$ has influence—unhappy. Since none of the flips of the input nodes of γ_ω, should they have flipped at all, make γ_ω unhappy, it follows that there is a flip of $\mu_i^{\kappa,2\omega}$. Node $\mu_i^{\kappa,2\omega}$ has influence only on $\mu_i^{\kappa,2\omega+1}$ which becomes unhappy by the flip of $\mu_i^{\kappa,2\omega}$. Analogously to the flip of $\mu_i^{\kappa,2\omega}$ it follows that $\mu_i^{\kappa,2\omega+1}$ also flips. Node $\mu_i^{\kappa,2\omega+1}$ itself has only influence on $\mu_i^{\kappa,2\omega+2}$ and the formula $\varphi(\mu_i^{\kappa,2\omega+2})$ is satisfied according to (3.5.31). Thus, $\mu_i^{\kappa,2\omega+2}$ is pushed. Node $\mu_i^{\kappa,2\omega+2}$ has only influence on $\mu_i^{\kappa,2\omega+3}$ whose flip follows analogously to the flip of $\mu_i^{\kappa,2\omega+1}$.

Now we consider the case that γ_ω is R'-incorrect in R_j. Then at least one of the inputs of γ_ω flips according to the induction hypothesis. If one of them flips then its flip makes γ_ω unhappy. If both of them flip then exactly one of the two flips makes γ_ω unhappy since if γ_ω becomes unhappy after the flip of the first input node then the flip of the second input node cannot make γ_ω unhappy again—otherwise γ_ω would have been R'-correct in R_j, which is a contradiction. Thus, in both cases node γ_ω can flip at most once. Node γ_ω has influence only on $\gamma_{\omega+1}$ but the formula $\varphi(\gamma_\omega)$ is according to (3.5.26) not satisfied as long as $\mu_i^{\kappa,2\omega+1}$ has its unnatural color.

Hence, it follows as in the case for the R'-correctness of γ_ω that there are flips of the nodes $\mu_i^{\kappa,2\omega}$ and $\mu_i^{\kappa,2\omega+1}$. After the flip of $\mu_i^{\kappa,2\omega+1}$ node $\mu_i^{\kappa,2\omega+2}$ is open. However, the formula $\varphi(\mu_i^{\kappa,2\omega+2})$—see (3.5.31)—is not satisfied as long as $\gamma_{\omega+1}$ did not flip and therefore $\mu_i^{\kappa,2\omega+2}$ is blocked. On the other hand, after the flip of $\mu_i^{\kappa,2\omega+1}$ the formula $\varphi(\gamma_{\omega+1})$ is satisfied according to (3.5.26) which implies that $\gamma_{\omega+1}$ is pushed. The formulas of the nodes on which $\gamma_{\omega+1}$ has influence in V^{C^h} are not satisfied according to (3.5.26) and (3.5.27)

since the nodes $\mu_i^{\kappa,\alpha}$ for $2\omega+5 \leq \alpha \leq 2m$ have their unnatural colors and therefore are, depending on whether or not their corresponding input node in G^h flipped, either closed or blocked. But the formula $\varphi(\mu_i^{\kappa,2\omega+2})$—see (3.5.31)—is satisfied after the flip of $\gamma_{\omega+1}$, which implies that $\mu_i^{\kappa,2\omega+2}$ flips. Analogously to the flip of $\mu_i^{\kappa,2\omega+1}$ it follows that there is a flip of $\mu_i^{\kappa,2\omega+3}$ after the flip of $\mu_i^{\kappa,2\omega+2}$. This finishes the consideration of the flips of the R'-incorrect γ_α for $1 \leq \alpha \leq m$ and of the nodes $\mu_i^{\kappa,\alpha}$ for $2i \leq \alpha \leq 2p$.

It remains to show that there is a flip of the NOR-node $h(R')$ in case of $h(R') \in V_{nor}^{C^h}$—see third column of Table 3.4. For this, we assume that $h(R') \in V_{nor}^{C^h}$. From Lemma 3.5.39 we know that there was no flip of $\pi_{\sigma(R')}$ prior to the first flip of $\tau_{\sigma(R')}$ in r_{j+1}^k. The flip of $\tau_{\sigma(R')}$ took the edge $\{\tau_{\sigma(R')}, \pi_{\sigma(R')}\}$ out of the cut. Node $\mu_i^{\kappa,2p}$ has influence only on $\mu_i^{\kappa,2p+1}$ and the sequence of unique influences continues up to $\mu_i^{\kappa,2m}$. Neither one of the nodes that flipped after the flip of g_i in P1 nor anyone that is equal to a node $\mu_i^{\kappa,\alpha}$ for $p+1 \leq \alpha \leq 2m$ has influence on $h(R')$. Thus, there is a flip of $h(R')$.

P11 The flip of $\mu_i^{\kappa,2p}$ in P10 made $\mu_i^{\kappa,2p+1}$ unhappy. Node $\mu_i^{\kappa,2p+1}$ has influence in G^h only on $\mu_i^{\kappa,2p+2}$ and the sequence of unique influences continues up to node $\mu_i^{\kappa,2m}$. Neither of the nodes $\mu_i^{\kappa,\omega}$ for $2p \leq \omega \leq 2m$ has influence in G^h on any node outside of V_i^κ and the only one of them that occurs as a variable of a formula of a node of $D(\varphi) \setminus V_i^\kappa$ is $\mu_i^{\kappa,2m}$. In particular, $\mu_i^{\kappa,2m}$ occurs in the formula $\varphi(\rho_{\sigma(R')})$—see (3.5.33). In P13 we will show that the nodes $\mu_i^{\kappa,\omega}$ for $2p \leq \omega \leq 2m$ flip in ascending order in ω, but since their flips may begin as early as in P11, we already refer to their flips here.

If $h(R') \neq nil$ then the flip of γ_p in P10 makes $\pi_{\sigma(R')}$ unhappy since the input node of $g_{\sigma(R')}$ in C has the same color as $g_{\sigma(R')}$—the generalized pivot rule only chooses gates for which this is the case— according to (R2i), $\pi_{\sigma(R')}$ also has the color of $g_{\sigma(R')}$. Thus, node $\pi_{\sigma(R')}$ flips. Node $\pi_{\sigma(R')}$ only has influence on $\theta_{\sigma(R')}$ and the sequence of unique influences in G^h continues with $\xi_{\sigma(R')}$ and $\rho_{\sigma(R')}$. The formula $\varphi(\rho_{\sigma(R')})$ is unsatisfied according to (3.5.33) as long as $\mu_i^{\kappa,2m}$ has its unnatural color. Thus, node $\rho_{\sigma(R')}$ is blocked until $\mu_i^{\kappa,2m}$ has its natural color in the case that $\xi_{\sigma(R')}$ did not yet flip and it is closed otherwise. Altogether, the flip of $\pi_{\sigma(R')}$ and the flips of the nodes that can be initiated by the flip of $\pi_{\sigma(R')}$, namely the nodes $\theta_{\sigma(R')}$ and $\xi_{\sigma(R')}$, do not affect the happiness of any node of V_i^κ or the satisfaction of a formula of a node of V_i^κ.

P12 The flip of $\pi_{\sigma(R')}$ in P11 made only $\theta_{\sigma(R')}$ unhappy. Since no one of the nodes $\mu_i^{\kappa,\omega}$ for $2p \leq \omega \leq 2m$ has influence on $\theta_{\sigma(R')}$ it follows that there is a flip of $\theta_{\sigma(R')}$.

P13 Analogously to the flip of $\theta_{\sigma(R')}$ in P12 it follows that there is a flip of $\xi_{\sigma(R')}$. After the flip of $\xi_{\sigma(R')}$ node $\rho_{\sigma(R')}$ is open. However, the formula $\varphi(\rho_{\sigma(R')})$ is unsatisfied

according to (3.5.33) as long as $\mu_i^{\kappa,2m}$ has its unnatural color. Thus, node $\rho_{\sigma(R')}$ is blocked until $\mu_i^{\kappa,2m}$ has its natural color.

The formulas $\varphi(\mu_i^{\kappa,\omega})$ for $2p \leq \omega \leq 2m$ for which $\mu_i^{\kappa,\omega} \in D(\varphi)$ are satisfied according to (3.5.32) since all nodes τ_ω for $\sigma(R') < \omega < n$ are R'-correct—the nodes themselves and their inputs were R'-correct in R_j. Therefore, as in P9 for the unnatural colors it follows that the nodes $\mu_i^{\kappa,\omega}$ for $2p \leq \omega \leq 2m$ that did not yet flip back to their natural color do it in ascending order in ω before $\rho_{\sigma(R')}$ flips.

P14 After the flip of $\xi_{\sigma(R')}$ in P13, node $\rho_{\sigma(R')}$ is open. The formula $\varphi(\rho_{\sigma(R')})$ is satisfied after the flip of $\mu_i^{\kappa,2m}$ in P11–P13. Thus, $\rho_{\sigma(R')}$ is pushed. This finishes the consideration of the flips.

Let $1 \leq k \leq q$ be the smallest index such that r_1^k contains the flips of the phases P1–P14. In the following, we show that the conditions (i)–(iv) of Lemma 3.5.40 are satisfied for R_k and r_1^k, respectively.

For condition (i) we have to show that R_k is recurring and begin with property (R1). Since R_j is recurring, property (R1) is satisfied in R_j. The only nodes that have influence on NOR-nodes of V^C are, by assumption, NOT-nodes of V^C. The only NOT-node of V^C that flips in r_{j+1}^k is g_i—see Table 3.4. If the flip of g_i makes a NOR-node unhappy then, also by assumption, $h(R')$ is the unhappy NOR-node and flips in P2–P10—see Table 3.4. After its flip property (R1) is satisfied again. For (R2i) note that g_i flips in P1 and if $h(R') \neq nil$ then the nodes $\pi_{\sigma(R')}$, $\theta_{\sigma(R')}$, $\xi_{\sigma(R')}$ and $\rho_{\sigma(R')}$ flip in P11–P14. Property (R2ii) is satisfied in R_k since, besides g_i in P1, the nodes $\lambda_i^{0,1}$, $\lambda_i^{0,2}$, $\lambda_i^{0,3}$, $\lambda_i^{1,1}$, $\lambda_i^{1,2}$ flip exactly once according to P2–P8. Property (R2iii) is satisfied since the nodes of V_i^κ flip exactly twice. Finally, the properties (R3) and (R4) follow from the flips in P10.

Now we show condition (ii). Since R_j is recurring, each $v \in V_I^h \setminus \{g_i\}$ has the opposite color as $u := H_{G^h}(v)$ in R_j. Since r_1^j is alternating, it follows that each node of $D(\varphi)$ is closed in R_j. By means of Table 3.4 one can verify that for each flip of u there is a flip of v after the flip of u in r_{j+1}^k and node u does not flip a second time before node v flips. Thus, r_1^k is alternating.

Since node g_i flips exactly once in r_{j+1}^k condition (iii) is satisfied and the condition (iv) can easily be verified by means of Table 3.4. This finishes the proof of Lemma 3.5.40.□

Lemma 3.5.40 implies the claim of the Enforcing Theorem for the remaining case $h(P^C) \neq nil$. □

3.6 All-Exp Property

Theorem 3.6.1 (All-Exp Theorem). LocalMax-Cut *has the all-exp property for graphs with maximum degree four.*

Proof. We adopt several parts of the proof of the Is-Exp Theorem (i.e., Theorem 3.4.1). In particular, we let C^n be the is-exp circuit (see Definition 3.4.3), $P_0^n := P^n$ be the initial

is-exp partition of the nodes of the graph $G^n = (V^n, E^n)$ that constitutes C^n, and $s(n)$ be the is-exp sequence of dimension n built up on the basis of the shifted is-exp sequence $s(n-1)^+$ of dimension $n-1$ by appropriately adding the first and the second is-exp modules t_1 and t_2, respectively. For $n \in \mathbb{N}$ we let $s(n) = (w_1^n, \ldots, w_{q_n}^n)$ for $q_n \in \mathbb{N}$ and $w_i^n \in V^n$ for all $1 \leq i \leq q_n$. We show the theorem by means of the Enforcing Theorem (i.e., Theorem 3.5.22) and develop for this purpose a polynomial-time computable pivot rule h_n for any $n \in \mathbb{N}_0$ that induces $s(n)$ in G^n. The pivot rule makes use of the following notation.

Definition 3.6.2. *For $n \in \mathbb{N}_0$ and $P \in \mathscr{P}(V^n)$ a node $v_i \in V^n$ for $2 \leq i \leq 4(n-1)+2$ is called **pausing** in P if the following conditions are satisfied:*

- $i \equiv 2 \mod 4$
- $c_P(v_{i-1}) = c_P(v_i) = c_P(v_{i+1}) = 0$ *and* $c_P(v_{i+2}) = 1$
- *There is a $j > i$ for which v_j is unhappy in P.*

The pseudo-code of the pivot rule h_n is presented in Algorithm 3.6.

Input: Partition $P \in \mathscr{P}(V^n)$ for graph $G^n = (V^n, E^n)$
Output: An element of the set $V^n \cup \{nil\}$

1: **if** P is a local optimum for G^n **then**
2: **return** *nil*
3: **else**
4: **return** node v_i with smallest i that is unhappy and not pausing in P

Algorithm 3.6: The pivot rule h_n.

In the following, we prove by induction on n that h_n induces $s(n)$ in (G^n, P_0^n) for all $n \in \mathbb{N}_0$, i.e., $h_n(P_{i-1}^n) = w_i^n$ for all $1 \leq i \leq q_n$. Before showing the induction basis and the induction step, we first show a property of (G^n, P_0^n) for all $n \in \mathbb{N}_0$ that we use to simplify the argumentation of both the induction basis and the induction step. Each node of the set $S^n := \{v_{4n+5}, \ldots, v_{12n+13}\}$ is happy in P_0^n. Neither of them is influenced by any node of $V^n \setminus S^n$. Thus, each of them is happy in P_k^n for all $0 \leq k \leq q_n$. Consequently, when showing that h_n induces $s(n)$ in (G^n, P_0^n) we can ignore the nodes of S^n as possible candidates for an output of h_n in any of the partitions of T^n and consider their colors to be constant throughout the sequence T^n.

As induction basis, we consider the case $n = 0$. In Figure 3.19 the nodes of $M^0 := \{v_1, \ldots, v_4\} \subset V^0$ and their corresponding colors in the partitions of T^0 are drawn. In the drawing for $(M^0, P_i^0|_{M^0})$ for $1 \leq i \leq 3$ the node w_i^0, i.e., the node that flips next in P_i^0 according to $s(0)$, is marked by a black star next to it. Note that v_1 is never pausing. Then, one can verify by means of Figure 3.19 that the sequence induced by h_0 in (G^0, P_0^0) is $(v_1, v_2, v_1) = s(0)$.

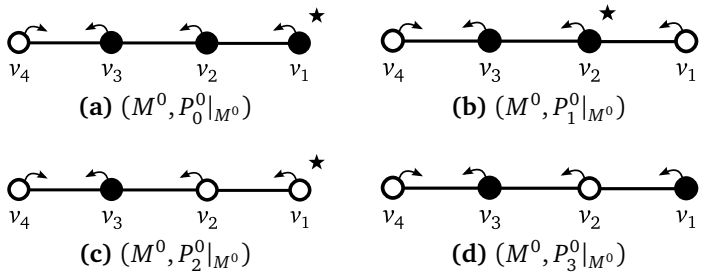


Figure 3.19: The partitions of M^0 according to $s(0)$ started at (G^0, P_0^0).

As induction hypothesis **(IH1)** we assume that h_n induces $s(n)$ in (G^n, P_0^n) for an arbitrary $n \in \mathbb{N}_0$. Before showing the induction step, we consider some properties of $s(n+1)$ and (G^{n+1}, P_0^{n+1}) in comparison to $s(n)$ and (G^n, P_0^n). In G^{n+1} no node of the set $M^{n+1} := \{v_1, \dots, v_4\} \subset V^{n+1}$ has influence on any node v_i for $i > 4$. Recall that the sequence $s(n+1)$ arises from $s(n)$ by increasing the index of all nodes of $s(n)$ by four and including the sequence t_1 after the flips of v_5 to white and the sequence t_2 after the flips of v_5 to black. Thus, there is a unique function $\sigma : \{1, \dots, q_{n+1}\} \to \{1, \dots, q_n\}$ such that $s(n+1)_1^k|_{V^{n+1} \setminus M^{n+1}} = s(n)_1^{\sigma(k)}$ for all $1 \leq k \leq q_{n+1}$. For σ we have the property that each node v_j for $j > 4$ has the same color in (G^{n+1}, P_k^{n+1}) for any $0 \leq k \leq q^{n+1}$ as v_{j-4} in $(G^n, P_{\sigma(k)}^n)$. Let $P \in \mathscr{P}(V^{n+1})$ such that all nodes of the set M^{n+1} are happy or pausing in P and $Q \in \mathscr{P}(V^n)$ such that $c_Q(v_j) = c_P(v_{j+4})$ for all j. Then $h_{n+1}(P) = v_{j+4}$ if $h_n(Q) = v_j$ for $1 \leq j \leq 12n + 13$ and $h_{n+1}(P) = nil$ if $h_n(Q) = nil$. Consequently, the induction hypothesis (IH1) implies for each partition P_i^{n+1} for $0 \leq i \leq q_{n+1}$ in which all nodes of M^{n+1} are happy or pausing that $h_{n+1}(P_i^{n+1}) = w_{i+1}^{n+1}$ if $i < q_{n+1}$ and that $h_{n+1}(P_i^{n+1}) = nil$ if $i = q_{n+1}$.

Now we show the induction step, i.e., we show that h_{n+1} induces $s(n+1)$ in (G^{n+1}, P_0^{n+1}). In particular, we show by induction on j that $h_{n+1}(P_{j-1}^{n+1}) = w_j^{n+1}$ for all $1 \leq j \leq q^{n+1}$. For the induction basis, note that in P_0^{n+1} all nodes of the set M^{n+1} are happy and therefore $h_{n+1}(P_0^{n+1}) = w_1^{n+1}$. As induction hypothesis **(IH2)** we assume for an arbitrary $0 \leq j < q^{n+1}$ that $h_{n+1}(P_{k-1}^{n+1}) = w_k^{n+1}$ for all $1 \leq k \leq j$.

At first, we show that for the induction step, we can focus on the nodes of M^{n+1} and the nodes that have influence on them. For this, we consider the case that $w_j^{n+1} \in V^{n+1} \setminus M^{n+1}$. Then each node of M^{n+1} is either happy or pausing in P_{j-1}^{n+1} since otherwise $h(P_{j-1}^{n+1}) \in M^{n+1}$. Assume that w_j^{n+1} has no influence on a node of M^{n+1}. Then each node of M^{n+1} that is happy in P_{j-1}^{n+1} is also happy in P_j^{n+1}. If v_2 is pausing in P_{j-1}^{n+1} then it is not pausing in P_j^{n+1} only if no node v_k for $k > 2$ is unhappy in P_j^{n+1}. Due to (IH1) the function $h_n(\cdot)$ only returns *nil* at the end of the sequence $s(n)$. The last flip of $s(n)$ is the flip of $v_1 \in V^n$ to the black color. Recall that the node $v_1 \in V^n$ corresponds to the node $v_5 \in V^{n+1}$. After the flip of v_5 to the black color, however, node v_4 is unhappy. Thus, if v_2 is pausing in P_{j-1}^{n+1} then it is also pausing in P_j^{n+1} and therefore it remains to

show $h_{n+1}(P_j^{n+1}) = w_{j+1}^{n+1}$ for the cases in which w_j^{n+1} has influence on a node of M^{n+1} or $w_j^{n+1} \in M^{n+1}$.

The only nodes that are not in M^{n+1} but have influence on a node of M^{n+1} are $v_{4(n+1)+6}$ and v_5. We already know that $v_{4(n+1)+6}$ does not flip in $s(n+1)$ and therefore $w_j^{n+1} \neq v_{4(n+1)+6}$. Recall that the flips of the nodes of M^{n+1} are induced by the sequences t_1 and t_2 and that after the execution of $t := t_1 \circ t_2$ the same partition as in P_0^{n+1} is reached for the nodes of M^{n+1} due to Observation 4.

Since after each of the ten flips of t there is at least one node that flipped an odd number of times in t, there are ten different partitions of the nodes of M^{n+1} in $s(n+1)$. Together with the flips of v_5 to the white color that precede the corresponding flips of t_1 in M^{n+1} and the flips of v_5 to the black color that precede the corresponding flips of t_2 we get twelve different partitions $Q_0, \ldots, Q_{11} \in \mathscr{P}(M^{n+1} \cup \{v_5\})$ in T^{n+1}, i.e., for each $0 \leq i \leq q_{n+1}$ there is a $0 \leq j \leq 11$ such that $P_i^{n+1}|_{M^{n+1} \cup \{v_5\}} = Q_j$.

The twelve partitions $Q_0, \ldots, Q_{11}$ are depicted in Figure 3.20 where Q_0 is the partition of $M^+ := M^{n+1} \cup \{v_5\}$ in P_0^{n+1}, i.e., $P_0^{n+1}|_{M^+} = Q_0$, and partition Q_i for $i \in \{1,5\}$ arises from Q_{i-1} by flipping v_5 and partition Q_i for $i \in \{0,\ldots,11\} \setminus \{1,5\}$ arises from $Q_{i-1 \bmod 12}$ by flipping the corresponding node of t. As in Figure 3.19 we mark in (M^+, Q_i) for $0 \leq i \leq 11$ the node that flips between the partitions Q_i and $Q_{i+1 \bmod 12}$. In Figure 3.20a and Figure 3.20e the little star is gray—in contrast to the remaining figures where it is black—to indicate that the (possibly) following flip of v_5 is not necessarily the node that flips next in $s(n+1)$ but only the next flip of $s(n+1)|_{M^+}$—in the remaining partitions the node that flips next is the same in $s(n+1)$ as in $s(n+1)|_{M^+}$.

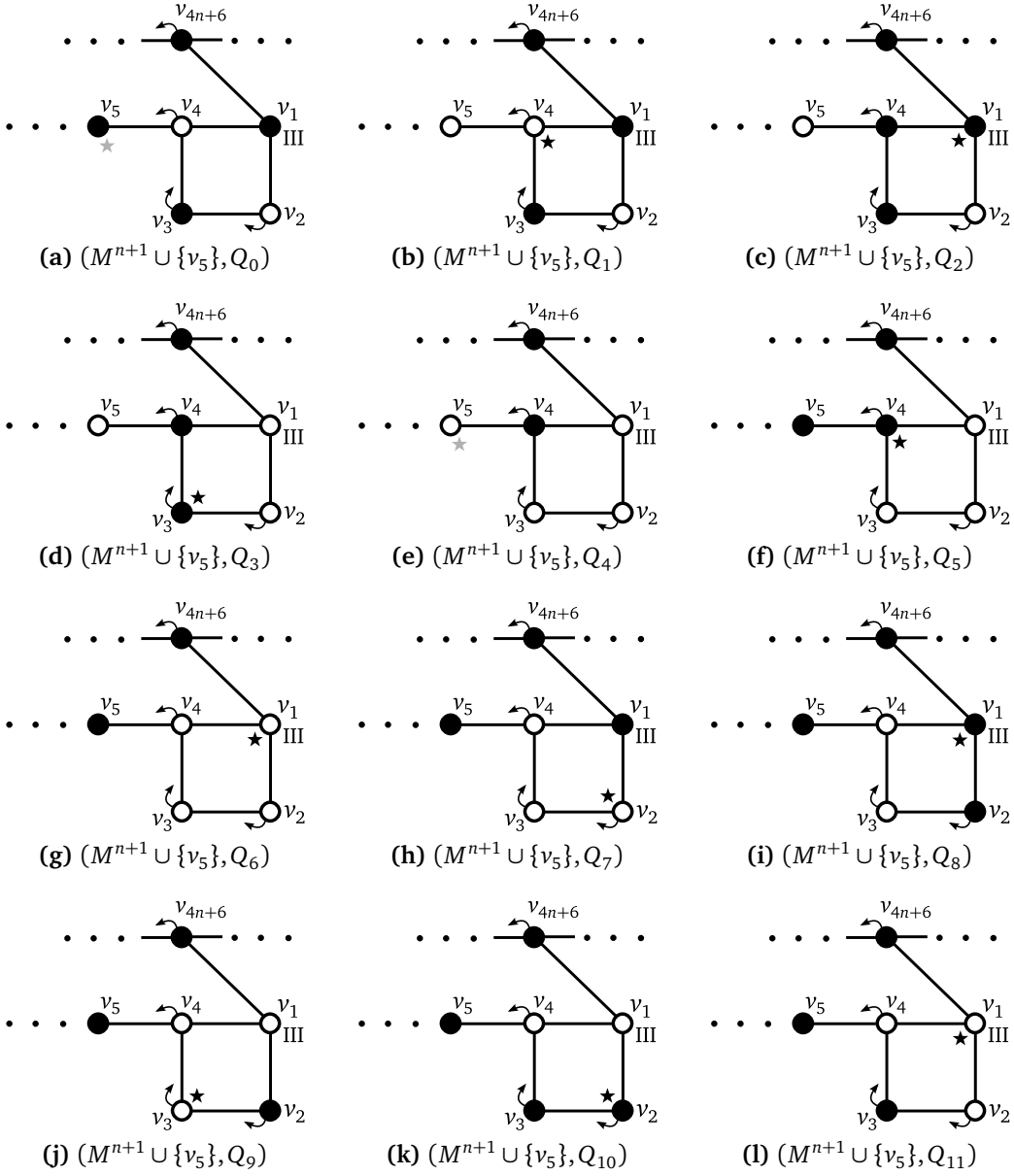
By means of Figure 3.20 one can verify that $h_{n+1}(P_j^{n+1}) = w_{j+1}^{n+1}$ as follows: For the partitions Q_k for $k \in \{1,\ldots,11\} \setminus \{4\}$ one can verify that $h_{n+1}(P_j^{n+1})$ coincides with the node that flips next in t and therefore in $s(n+1)$, i.e., the node marked by the star. In the partition Q_0 all nodes of M^{n+1} are happy and therefore (IH1) implies that $h_{n+1}(P_j^{n+1}) = w_{j+1}^{n+1}$ if $j < q_{n+1}$ and $h_{n+1}(P_j^{n+1}) = \mathit{nil}$ otherwise. Finally, in P_4 the nodes v_1, v_3 and v_4 are happy and v_2 is pausing and therefore, again, (IH1) implies that $h_{n+1}(P_j^{n+1}) = w_{j+1}^{n+1}$.

Thus, h_n induces the sequence $s(n)$ starting at (G^n, P_0^n). The pivot rule h_n is obviously polynomial-time computable. Consequently, the Enforcing Theorem (i.e., Theorem 3.5.22) implies that LocalMax-Cut has the all-exp property for graphs with maximum degree four. □

3.7 PSPACE-completeness of the Standard Algorithm Problem

Theorem 3.7.1 (SAPPSC Theorem). *The* StandardAlgorithmProblem *for* LocalMax-Cut *is* PSPACE*-complete for graphs with maximum degree four.*

Proof. Clearly, the problem is computable in polynomial space. We reduce from the PSPACE-complete problem of deciding whether a linear bounded automaton M halts for a given input [24]. A configuration of M for inputs of length n consists of the state of M, the position of the head and a string of length n. Thus, the number of configurations of



(a) $(M^{n+1} \cup \{v_5\}, Q_0)$ **(b)** $(M^{n+1} \cup \{v_5\}, Q_1)$ **(c)** $(M^{n+1} \cup \{v_5\}, Q_2)$

(d) $(M^{n+1} \cup \{v_5\}, Q_3)$ **(e)** $(M^{n+1} \cup \{v_5\}, Q_4)$ **(f)** $(M^{n+1} \cup \{v_5\}, Q_5)$

(g) $(M^{n+1} \cup \{v_5\}, Q_6)$ **(h)** $(M^{n+1} \cup \{v_5\}, Q_7)$ **(i)** $(M^{n+1} \cup \{v_5\}, Q_8)$

(j) $(M^{n+1} \cup \{v_5\}, Q_9)$ **(k)** $(M^{n+1} \cup \{v_5\}, Q_{10})$ **(l)** $(M^{n+1} \cup \{v_5\}, Q_{11})$

Figure 3.20: The partitions Q_i for $0 \leq i \leq 11$ of $M^{n+1} \cup \{v_5\}$ occurring in T^{n+1}.

M for inputs of length n is bounded by k^n for some constant k. We choose $m \in \mathbb{N}$ such that $2^{m-2} \leq k^n < 2^{m-1}$ and let $\boldsymbol{b(c)}$ be a bit vector that encodes a configuration c of M. Moreover, we let c_i for $i \in \mathbb{N}_0$ be the configuration of M after i steps of M where $c_i = c_j$ for all $j > i$ if c_i is a halting configuration.

We let C^m be the is-exp circuit (see Definition 3.4.3) of length m, add NOT-gates as depicted in Figure 3.21 to C^m and call the resulting circuit C^+. Let $G^+ = (V^+, E^+)$ be

the graph that constitutes C^+ and V^m be the set of nodes that represent the gates of C^m. For convenience, we use the same name for a node in V^+ and the gate of C^+ that it represents—recall that each gate of C^+ is represented by exactly one node in V^+. We let y^κ for $1 \leq \kappa \leq 3$ be the vector $(y_1^\kappa, \ldots, y_m^\kappa)$.

Proof in a nutshell The reduction uses the Enforcing Theorem (i.e., Theorem 3.5.22) to simulate the steps of M as a sequence of partitions induced in (G^+, P_0)—see Figure 3.21—for a partition $P_0 \in \mathscr{P}(V^+)$ that is specified later. The decision whether M halts is made by means of the colors of the nodes of G in the local optimum reached at the end of the sequence of partitions.

More concretely, for the initial partition P_0 of G^+ we choose the colors of the nodes of V^m such that they correspond to the initial is-exp partition (see Definition 3.4.3), the colors of the nodes of x such that each of them is happy, the colors of y^3 such that they encode c_0, and the colors of the nodes of y^1 and y^2 such that the lightest edge incident to each of its nodes is in the cut. The nodes of y^3 are supposed to periodically contain by means of their colors encoding of the current configuration of M. For this purpose, we introduce a generalized pivot rule f that performs the simulation of M by means of the colors of the vector y^3 as follows.

The rule f first chooses flips of the nodes of V^+ according to the pivot rule h_m—see Algorithm 3.6—until g_1 flips for the first time, i.e., it flips to the black color. Then it selects the nodes x_i for $1 \leq i \leq 2m-1$ to flip consecutively in ascending order in i—note that after these flips the nodes x_j are white for all odd j. Let c' be the configuration of M one step after the configuration c that is encoded by the colors of the nodes of y^3. Then f selects consecutively in ascending order in i those nodes y_i^1 for $1 \leq i \leq m$ that would be black if the colors of y^1 encoded c'. The partitions arising after this step will, among others, be called recurring. Then f selects nodes of V^m according to h_m until g_1 flips back to white. Then it again chooses the nodes of x until all of them are happy—after that, the nodes x_j are black for all odd j. Then it consecutively selects in ascending order in i the nodes y_i^1 for $1 \leq i \leq m$ that would be white if the colors of y^1 encoded c'. After that, the vector of colors of y^1 in fact encodes c'. Finally, it chooses the unhappy nodes of y^2 and then those of y^3 to flip consecutively. Then the vector y^3 also encodes c'—the partitions arising after this step will, beside the initial partitions, be called *strictly* recurring. This procedure is repeated until there are no unhappy nodes in V^m. Then we can show for the local optimum that is finally reached that the vector of colors of y^3 equals the vector of colors of y^1 if and only if M halts if started with c_0.

Now we continue with the proof. Let $P \in \mathscr{P}(V^n)$ be a partition in which the colors of the nodes of y^3 encode a configuration c of M. We let $\boldsymbol{d(P)}$ be the vector of the colors of y^3 in P and $\boldsymbol{d^+(P)}$ be the bit vector encoding the configuration of M one step after the configuration that is encoded by $d(P)$. Moreover, for a bit vector $z \in \{0,1\}^n$ for $n \in \mathbb{N}$ we let z_i for $1 \leq i \leq n$ be the i-th component of z.

Definition 3.7.2. *A partition $P \in \mathscr{P}(V^+)$ is called **recurring** if for all $1 \leq i \leq 2m-1$*

- *x_i is happy in P*

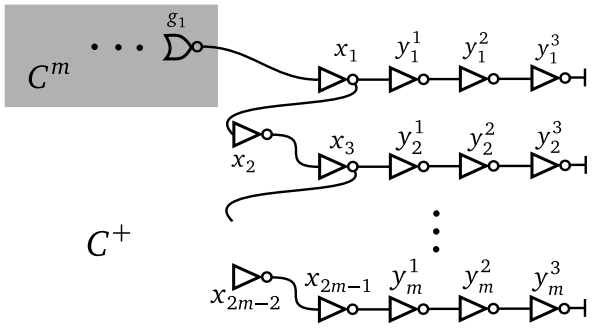


Figure 3.21: Gates of the Boolean circuit C^+.

and for all $1 \leq i \leq m$

- $c_P(y_i^2) \neq c_P(y_i^3)$,
- *if* $c_P(g_1) = 1$ *then* $c_P(y_i^1) \neq c_P(y_i^2)$,
- *if* $c_P(g_1) = d_i^+(P) = 0$ *then* $c_P(y_i^1) = 0$.

A recurring partition $P \in \mathscr{P}(V^+)$ *is called* ***strictly recurring*** *if* $c_P(g_1) = 1$.

Definition 3.7.3. *For a partition* $P \in \mathscr{P}(V^+)$*, we call* y_j^κ *for any* $1 \leq j \leq m$*,* $1 \leq \kappa \leq 3$ ***up-to-date*** *in* P *if the following conditions are satisfied:*

- *Case* $\kappa = 1$*: If* $d_j^+(P) = c_P(g_1) = 0$ *then* $c_P(y_j^1) = 0$.
- *Case* $\kappa = 2$*: If* $c_P(g_j) = 1$ *then* y_j^2 *is happy in* P.
- *Case* $\kappa = 3$*:* y_j^3 *is happy in* P.

Otherwise we call y_i^κ ***outdated***.

We choose the partition $P_0 \in \mathscr{P}(V^+)$ such that it satisfies the following four conditions:

- $c_{P_0}(v) = c_{P_0^m}(v)$ for all $v \in V^m$ where P_0^m is the initial is-exp partition of V^m.
- $d(P_0) = b(c_0)$.
- x_i is happy for all $1 \leq i \leq 2m-1$.
- $c_{P_0}(y_i^1) \neq c_{P_0}(y_i^2) \neq c_{P_0}(y_i^3)$ for all $1 \leq i \leq 2m-1$.

Now we introduce a generalized pivot rule f for the graph G^+ whose purpose is for each step of M to successively change the colors of the nodes of V^+ such that the colors of the nodes of the vector y^3 encode the configuration of M after the corresponding step. The generalized pivot rule f is presented in Algorithm 3.7. It makes use of the pivot rule h_m as introduced in Algorithm 3.6.

In the rest of the proof we make use of the invariant in Lemma 3.7.4.

Input: Partition $Q \in \mathscr{P}(V^+)$ for graph G^+
Output: An element of the set $V^+ \cup \{nil\}$

```
1: if Q is locally optimal for G^+ then
2:     return nil
3: else
4:     if Q is recurring then
5:         return h_m(Q|_{V^m})
6:     else
7:         if ∃x_i for 1 ≤ i ≤ 2m − 1 that is unhappy in Q then
8:             return x_i with smallest i that is unhappy
9:         else
10:            if ∃y_i^κ for 1 ≤ i ≤ m, 1 ≤ κ ≤ 3 that is outdated in Q then
11:                return y_i^κ with smallest m · κ + i that is outdated
12:            else
13:                return nil
```

Algorithm 3.7: The generalized pivot rule f.

Lemma 3.7.4. *Let $s := (w_1, \ldots, w_q)$ for $q \in \mathbb{N}$, $w_j \in V^+$ for all $1 \leq j \leq q$ be the sequence of improving flips induced by the generalized pivot rule f starting at (G^+, P_0) and $0 \leq i \leq q$ be such that the following conditions are satisfied:*

- *P_i is strictly recurring.*
- *$d(P_i) = b(c_r)$ for some $r \in \mathbb{N}_0$.*
- *g_1 flips k times in s_1^i for $0 \leq k \leq 2^m - 2$.*

Then there is an index $i < j \leq q$ such that the following conditions are satisfied:

- *P_j is strictly recurring.*
- *$d(P_j) = b(c_{r+1})$.*
- *g_1 flips $k+2$ times in s_1^j.*

Proof. The argumentation is divided into nine steps, namely Step1–Step8.

Step1 Since g_1 flips k times in s_1^i for $0 \leq k \leq 2^m - 2$, not all nodes of V_m are happy in P_i—recall that we showed in the proof for Theorem 3.6.1 that h_n induces 2^m flips in V_m. Then, since P_i is strictly recurring, the generalized pivot rule f chooses nodes of V^m as long as g_1 did not yet flip to white in s_i^q according to line 5 of Algorithm 3.7.

Step2 After the first flip of g_1 in s the resulting partition is not recurring anymore since x_1 is unhappy then. Then the function f chooses the nodes x_i for $1 \leq i \leq 2m-1$

to flip in ascending order in i according to line 8 of Algorithm 3.7 until all of them are happy.

Step3 When x_i is happy for all i then f chooses those nodes of y_i^1 for $1 \leq i \leq m$ to flip in ascending order in i for which $d_i(P) \neq c_P(y_i^3) = 0$ according to line 11 where P is the corresponding partition—note that the flips of the nodes y_i^1 that satisfy this condition are in fact improving since the nodes that have influence on them, i.e., the nodes x_j for odd $1 \leq j \leq 2m-1$ are white. After the flips of the nodes y_i^1 for the corresponding i, the resulting partition is recurring again.

Step4 As in Step1, the generalized pivot rule f chooses flips of the nodes of V^m according to h_m until g_1 flips back to black whereafter the resulting partition is not recurring anymore.

Step5 As in Step2, the nodes x_i for all $1 \leq i \leq 2m-1$ flip exactly once in ascending order in i.

Step6 Similarly to Step3, f chooses that nodes of y_i^1 for $1 \leq i \leq m$ in ascending order for the next flips for which $d_i(P) \neq c_P(y_i^3) = 1$ for the corresponding partition P. After these flips, the colors of the nodes of the vector y^1 encode the next configuration c_{r+1} of M with respect to the configuration encoded by the colors of the nodes of y^3, i.e., c_r.

Step7 f chooses the unhappy nodes y_i^2 for all $1 \leq i \leq m$ in ascending order in i for the next flips according to line 11 of Algorithm 3.7. Then the colors of the nodes y_i^2 for all i correspond to the bitwise complement of the encoding of c_{r+1}.

Step8 f lets the unhappy nodes y_i^3 for all $1 \leq i \leq m$ flip according to, again, line 11 whereafter the colors of the nodes of y^3 correspond to the encoding of the configuration c_{r+1} and the resulting partition is strictly recurring again. □

By definition, P_0 is strictly recurring and in s_1^0, i.e., the empty sequence, node g_1 does not flip. Then Lemma 3.7.4 implies for the sequence s induced by f starting at (G^+, P_0) the following two conditions:

- Node g_1 flips 2^m times in s.
- In the partition P after the $2k$-th flip of g_1 in s for any $0 \leq k \leq 2^{m-1}$ we have $d(P) = c_k$.

Since node g_1 flips 2^m times in s, all nodes of V^m are happy in P which implies $f(P) = nil$ according to line 5 of Algorithm 3.7.

Recall that if c_i for any $i \in \mathbb{N}_0$ is a halting configuration then $c_i = c_j$ for all $j > i$. Since f is polynomial-time computable, the Enforcing Theorem (i.e., Theorem 3.5.22) implies that one can compute in polynomial time a graph $G = (V, E)$ with $V^+ \subseteq V$ and a partition $P \in \mathscr{P}(V)$ for which $P|_{V^+} = P^+$ such that for any sequence t of improving flips starting at (G, P) we have $t|_{V^+} = s$. Thus, for the local optimum $Q \in \mathscr{P}(V)$ reached at

the end of the sequence t starting at (G, P), we have $Q|_{V^+} = P_q$. Consequently, node y_i^3 for any $1 \leq i \leq m$ has the same color in Q as in P_q and therefore the vector of the colors of y^3 in Q encodes the configuration $c_{2^{m-1}}$. Therefore, if $c_{2^{m-1}}$ is a halting configuration then M halts if started with c_0 and if $c_{2^{m-1}}$ is not a halting configuration then M does not halt since at least one configuration occurs at least twice in the sequence of partitions induced by s in (G^+, P^+) due to $2^{m-1} > c^n$. Thus, the colors of the nodes y_i^3 for all i in Q encode a halting configuration of M if and only if M halts if started with c_0. □

Chapter 4

Complexity of Local Max-Cut: Maximum Degree Five

4.1 Overview of Contribution

At first, we introduce a technique by which we substitute graphs whose nodes of degree greater than five have a certain type—we will call these nodes *comparing*—by graphs of maximum degree five. For the graphs arising by this substitution, we show that in local optima that have a certain property the nodes of the subgraphs that substitute the comparing nodes have unique colors, i.e. the Substituting Lemma (Lemma 4.3.3). In particular, those nodes of the subgraph substituting a comparing node v that are adjacent to nodes of the original graph all have the same color. Namely, they have the color that v had in the corresponding partition of the original graph if it was happy. Thus, from the viewpoint of the nodes that are adjacent to v in the original graph, the nodes of the subgraph substituting v behave in certain local optima of the extended graph as the single node v in the original graph.

Then we prove the PLS-completeness of computing a local optimum of Max-Cut on graphs with maximum degree five by reducing from the PLS-complete problem CircuitFlip. In a nutshell, we map instances of CircuitFlip to graphs of degree greater than five. Some parts of the graphs are adjustments of subgraphs of the PLS-completeness proof of Schäffer and Yannakakis [54]. Then, using the Substituting Lemma, we show that local optima for these graphs induce local optima in the corresponding instances of CircuitFlip.

4.2 Usage of the P-hardness Reduction

In our technique, as well as in the PLS-completeness proof, we make use of the Constituting Theorem (in particular of Theorem 3.3.1(ii)). The graph G_f, as introduced in the said proof, can be constructed in logarithmic space and thus polynomial time for any polynomial-time computable function f. In the rest of the chapter we use the graph G_f for several functions f and we will scale the weights of its edges. Then the edges of G_f give incentives of appropriate weight to certain nodes of the graphs to which we add G_f. The incentives *bias* the nodes to take the colors induced by f. We already point out that for any node v we will introduce at most one subgraph that biases v. However, we

sometimes introduce more than one subgraph that looks at a node v. For an overview see Figure 4.1.

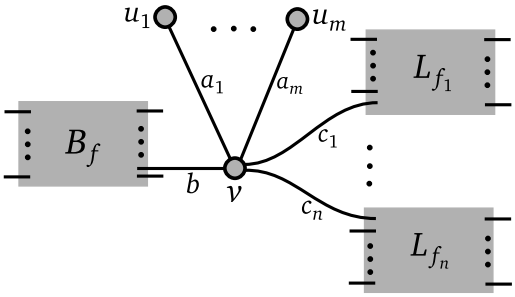


Figure 4.1: Subgraph B_f biases v and subgraphs $L_{f_1}, \ldots, L_{f_n}$ look at v.

In Figure 4.1, there is a subgraph B_f that looks at a subset of the nodes of the given graph and biases v, possibly among other nodes, according to the function f. Moreover, there are subgraphs $L_{f_1}, \ldots, L_{f_n}$ that look at v, possibly among other nodes as well, and bias some further nodes according to the functions $f_1, \ldots, f_n$. Finally, there are nodes $u_1, \ldots, u_m$ adjacent to v that are not contained in any subgraph that biases v or looks at v. The relations of the weights of the edges incident to v will be such that $b < a_i$ for all $1 \leq i \leq m$ and $b > \sum_{i=1}^{n} c_i$. If there is no subgraph that biases v then we will have $a_j > \sum_{i=1}^{n} c_i$ for all $1 \leq j \leq m$.

4.3 Substituting Certain Nodes of Unbounded Degree

The following definition introduces a notation for a family of (sub-)graphs and a notation for a factor that is related to the weights of the edges of the corresponding (sub-)graph. The (sub-)graphs and their factors are needed for the subsequent definition of *comparing* nodes.

Definition 4.3.1. *Let $G = (V, E)$ be a graph. For $n \in \mathbb{N}$ we let $\boldsymbol{B(n)}$ be an arbitrary but fixed subgraph of G that looks at a node $v \in V$ and biases nodes $u_1, \ldots, u_{4n-1} \in V$ in the following way. The nodes $u_1, \ldots, u_{2n}$ are biased to the opposite color of v and the nodes $u_{2n+1}, \ldots, u_{4n-1}$ are biased to the same color as v (a possible implementation of $B(n)$ is a binary tree with appropriate edge weights, where v is the root and the nodes u_i for $1 \leq i \leq 4n-1$ are leaves of appropriate height). Let w_{max} be the maximum weight of all edges of $B(n)$ and w_{min} be the corresponding minimum. Then we call $\boldsymbol{b(n)} := w_{max}/w_{min}$.*

Definition 4.3.2. *Let $G = (V, E)$ be a graph. A node $v \in V$—see Figure 4.2—is called **comparing** if there is an $m \in \mathbb{N}$ with $m \geq 3$ such that the following conditions are satisfied:*

i) v is adjacent to nodes $u_i^j, bn(v) \in V \setminus \{v\}$ for $1 \leq j \leq m$, $1 \leq j \leq 2$ with edge weights $w(\{u_i^j, v\}) = a_i$, $w(\{bn(v), v\}) = \delta$ for $a_i, \delta \in \mathbb{Q}_{>0}$ and optionally to a node $ln(v)$ with edge weight $w(\{ln(v), v\}) = \epsilon$ for $\epsilon \in \mathbb{Q}_{>0}$.

ii) $a_i \geq 2a_{i+1}$ *for all* $1 \leq i < m$ *and* $a_m \geq 2\delta$.

iii) If v *is adjacent to* $ln(v)$ *then* $\delta > b(m) \cdot \epsilon$.

For u_i^j *with* $1 \leq i \leq m$, $1 \leq j \leq 2$ *we call the node* u_i^k *with* $1 \leq k \leq 2$ *and* $k \neq j$ *adjacent to* v *via the unique edge with the same weight as* $\{u_i^j, v\}$ *the* ***counterpart*** *of* u_i^j *with respect to* v. *The nodes* u_i^j *for all* i, j *and the node* $ln(v)$*—should it be adjacent to* v*—are called* ***receiving nodes*** *of* v.

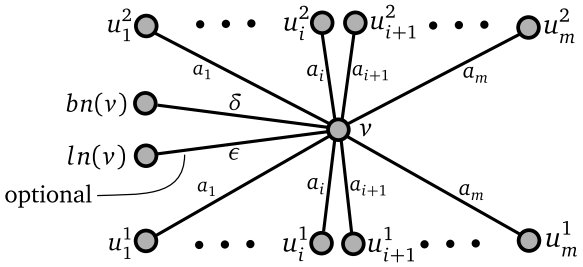


Figure 4.2: Node v is a comparing node.

Comment The name *comparing node* stems from its behavior in local optima. If we treat the colors of the neighbors $u_1^1, \ldots, u_m^1$ of v as a binary number a, where u_1^1 is the most significant bit, and the colors of $u_1^2, \ldots, u_m^2$ as the bitwise complement of a binary number b, then, in a local optimum, the comparing node v is white if $a > b$, it is black if $a < b$, and if $a = b$ then v has the opposite color of $bn(v)$. In this way, the color of v "compares" a and b in local optima.

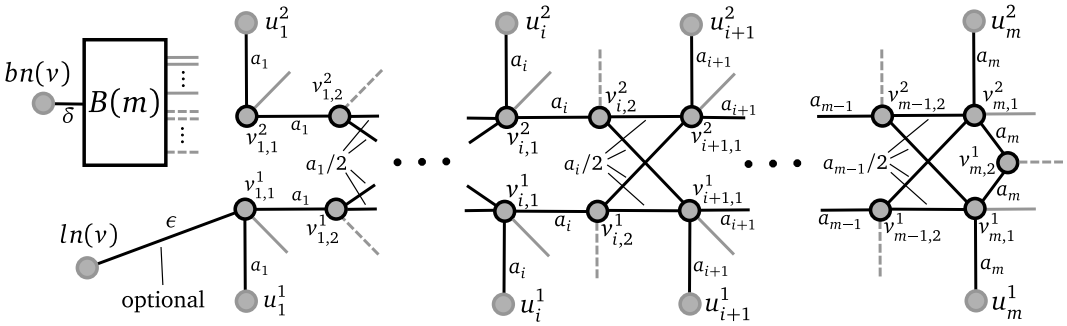


Figure 4.3: The gadget that substitutes a comparing node v.

In the following, we let $G = (V, E)$ be a graph and $v \in V$ be a comparing node with adjacent nodes and incident edges as in Figure 4.2. We say that we **degrade** v if we remove v and its incident edges and add the following nodes and edges. We introduce nodes $v_{i,j}^k(v)$ for $1 \leq i < m, 1 \leq j \leq 2, 1 \leq k \leq 2$, nodes $v_{m,1}^k(v)$ for $1 \leq k \leq 2$ and $v_{m,2}^1(v)$. For the purpose of succinctness, we may omit the attached expression "(v)" if

it is clear from the context which comparing node is substituted. The edges and their corresponding weights are as depicted in Figure 4.3—the nodes u_i^j in Figure 4.3 have gray circumcircles to indicate that they, in contrast to the other nodes, also occur in G. Furthermore, we add the subgraph $B(m)$ that looks at u and biases all nodes $v_{i,1}^k$ to the opposite of the color of u (this is illustrated by short gray edges in Figure 4.3) and the nodes $v_{i,2}^k$ to the color of u (short gray dashed edges). The weights of the edges of $B(m)$ are scaled such that the unique edge of $B(m)$ incident to u has the weight δ—this edge is depicted in Figure 4.2. We let $\boldsymbol{D(G)} = (D(V), D(E))$ be the graph arising from G by iteratively degrading all comparing nodes of G.

Now we introduce several notations for partitions $P \in \mathscr{P}(D(V))$ that encapsulate properties of the nodes that substitute v in $D(V)$. We say that v is **weakly indifferent** in P if $c_P(u_i^1) \neq c_P(u_i^2)$ for all $1 \leq i \leq m$. If v is not weakly indifferent in P then we call the two nodes u_i^1, u_i^2 adjacent to v via the edges with highest weight for which $c_P(u_i^1) = c_P(u_i^2)$ the **decisive** neighbors of v in P. We let $V_{com} \subseteq V$ be the set of comparing nodes of V and $\boldsymbol{col_P} : V_{com} \to \{0,1\}$ be the partial function defined by

$$col_P(v) = \begin{cases} 0, & \text{if } c_P(v_{i,1}^j) = 0 \text{ for all } i,j. \\ 1, & \text{if } c_P(v_{i,1}^j) = 1 \text{ for all } i,j. \end{cases}$$

We say that v has the color $\kappa \in \{0,1\}$ if $col_P(v) = \kappa$. Moreover, we say that v is **guided** in P if v is weakly indifferent in P or v is not weakly indifferent and $bn(v)$ has the same color as the decisive neighbors of v in P.

Lemma 4.3.3 (Substituting Lemma). *Let $G = (V,E)$ be a graph and $P \in \mathscr{P}(D(G))$ be a local optimum in which each comparing node is guided. Then for each comparing node $v \in V$ we have*

$$col_P(v) \neq c_P(bn(v)).$$

Comment Note the restriction that in the local optimum P, node v is guided. In the proof of the Completeness Theorem (i.e., Theorem 4.4.2) every comparing node v is designed to be guided in every local optimum. Then we can use the Substituting Lemma to argue about $col_P(v)$ in $D(G)$. Moreover, let $S(v)$ be the set containing the nodes $v_{j,1}^i$ for all i,j. The property $col_P(v) \neq c_P(bn(v))$ implies that all nodes of $S(v)$ have the same color in P, namely $col_P(v)$. Thus, from the viewpoint of the receiving nodes of v, the property means that the nodes of $S(v)$ behave in P as a single node.

Proof (of Lemma 4.3.3). Let $v \in V$ be an arbitrary comparing node with adjacent nodes and incident edges as depicted in Figure 4.2. In the following, we show for a local optimum $P \in \mathscr{P}D(V)$ that $col_P(v) \neq c_P(bn(v))$—see Figure 4.3. Then we get $col_P(w) \neq c_P(bn(w))$ for all comparing nodes $w \in V$ since v is chosen arbitrarily.

Let $\kappa := c_P(bn(v))$. For all i,j we call the color of $v_{j,1}^i$ **correct** if $c_P(v_{j,1}^i) = \overline{\kappa}$ and we call the color of $v_{j,2}^i$ correct if $c_P(v_{j,2}^i) = \kappa$. Moreover, we call $v_{j,k}^i$ correct for any i,j,k if it has its correct color.

Each node $v_{i,j}^k$ is biased by an edge with weight lower than δ to its correct color. Moreover, the edge $\{ln(v), v_{1,1}^1\}$, if existing, weighs less than all other edges incident to

$v^1_{1,1}$ due to $\delta > b(m) \cdot \epsilon$. Therefore, to show that a node $v^k_{i,j}$ for any i, j, k is correct in the local optimum P, it suffices to show that it gains at least half of the sum of weights of the incident edges with weight greater than δ if it is correct. We prove the Theorem by means of the following Lemmas which are each proven via straightforward inductive arguments.

Lemma 4.3.4. *Let $q \leq m$ and $c_P(u^1_i) = \kappa$ for all $i \leq q$. Then $v^1_{i,1}$ and $v^1_{i,2}$ are correct for all $i \leq q$.*

Proof. We prove the claim by induction on i. Due to $c_P(u^1_1) = \kappa$ we get the correctness of $v^1_{1,1}$—recall that the edge $\{ln(v), v^1_{1,1}\}$, if existing, weighs less than all other edges incident to $v^1_{1,1}$. For each $i \leq q$ the correctness of $v^1_{i,1}$ implies the correctness of $v^1_{i,2}$. Moreover, for each $i < q$, the correctness of $v^1_{i,2}$ together with $c_P(u^1_{i+1}) = \kappa$ implies the correctness of $v^1_{i+1,1}$. □

Lemma 4.3.5. *Let $q \leq m$, node $v^1_{i,1}$ and $v^1_{i,2}$ be correct for all $i \leq q$ and $v^2_{q,1}$ be correct. Then $v^2_{i,1}$ and $v^2_{i,2}$ are correct for all $i < q$.*

Proof. We prove the claim by induction on i. Node $v^2_{q,1}$ and $v^1_{q,1}$ are correct by assumption. For each $i < q$, node $v^2_{i,2}$ is correct if $v^2_{i+1,1}$ is correct since $v^1_{i+1,1}$ is correct by assumption. Moreover, for each $1 < i < q$, node $v^2_{i,1}$ is correct if $v^2_{i,2}$ is correct since $v^1_{i-1,2}$ is correct by assumption. Finally, node $v^2_{1,1}$ is correct if $v^2_{1,2}$ is correct. □

Lemma 4.3.6. *Let $q \leq m$. If $v^1_{q,1}$ and $v^2_{q,1}$ are correct then $v^k_{i,j}$ is correct for any j, k and $q \leq i \leq m$.*

Proof. If $q = m$ then the correctness of $v^1_{m,1}$ implies the correctness of $v^1_{m,2}$. The case $q < m$ is done by induction on i. Node $v^1_{q,1}$ and $v^2_{q,1}$ are correct by assumption. Assume that $v^1_{i,1}$ and $v^2_{i,1}$ are correct for an arbitrary $q \leq i < m$. Then the nodes $v^1_{i,2}$ and $v^2_{i,2}$ are correct whereafter the correctness of $v^1_{i+1,1}$ and $v^2_{i+1,1}$ follows. Finally, the correctness of $v^1_{m,1}$ implies the correctness of $v^1_{m,2}$. □

We first consider the case that v is weakly indifferent. Then, for each i, at least one of the nodes u^1_i and u^2_i has the color $\overline{\kappa}$. Due to the symmetry between the nodes $v^1_{i,j}$ and $v^2_{i,j}$ we may assume without loss of generality that $c_P(u^1_i) = \kappa$ for all i. Then Lemma 4.3.4 implies that $v^1_{i,1}$ and $v^1_{i,2}$ are correct for all i. Then the correctness of $v^1_{m,2}$ and $v^1_{m-1,2}$ together imply the correctness of $v^2_{m,1}$. Then Lemma 4.3.5 implies the correctness of $v^2_{i,1}$ and $v^2_{i,2}$ for all $i < m$.

Now assume that v is not weakly indifferent and let u^1_q and u^2_q be the decisive neighbors of v. As in the previous case we assume without loss of generality that $c_P(u^1_i) = \kappa$ for all $i \leq q$. Then, due to Lemma 4.3.4, node $v^1_{i,1}$ and $v^1_{i,2}$ are correct for all $i \leq q$. If $q = 1$ then $c(u^2_1) = \kappa$ implies the correctness of $v^2_{1,1}$. On the other hand, if $q > 1$ then the correctness

of $v^1_{q-1,2}$ and $c(u^2_q) = \kappa$ together imply the correctness of $v^2_{q,1}$. Then Lemma 4.3.5 implies the correctness of $v^2_{i,1}$ and $v^2_{i,2}$ for all $i < q$. Finally, Lemma 4.3.6 implies the correctness of $v^k_{i,j}$ for all j,k and $q \leq i \leq m$. □

4.4 PLS-completeness

Our reduction is based on the following PLS-complete problem CircuitFlip (in [30] it is called Flip, which we avoid in this thesis since the neighborhood of LocalMax-Cut has the same name).

Definition 4.4.1 ([30]). *An instance of* **CircuitFlip** *is a Boolean circuit C with n input bits and m output bits. A feasible solution of* CircuitFlip *is a vector $v \in \{0,1\}^n$ of input bits for C and the value of a solution is the output of C treated as a binary number. Two solutions are neighbors if they differ in exactly one bit. The objective is to maximize the output of C.*

Theorem 4.4.2 (Completeness Theorem). LocalMax-Cut *is* PLS*-complete for graphs with maximum degree five.*

Proof. We reduce from the PLS-complete problem CircuitFlip. Let C be an instance of CircuitFlip with input links $X_1,\dots,X_n$, outputs links $C_1,\dots,C_m$ and gates $G_N,\dots,G_1$. For the sake of simplicity, we let G_i also denote as the output of gate G_i. The two inputs of a gate G_i are denoted by $\boldsymbol{I_1(G_i)}$ and $\boldsymbol{I_2(G_i)}$. Without loss of generality, we make the following five assumptions. First, all gates are NOR-gates with a fan-in of two (this assumption can be made due to Proposition 2.4.3) and are topologically sorted such that $i > j$ if G_i is an input of G_j. Second, the gates $G_1,\dots,G_m$ compute the output of C where G_m is the most significant bit and $G_{m+1},\dots,G_{2m}$ compute the corresponding negations of the output bits. Third, the gates $G_{2m+1},\dots,G_{2m+n}$ and $G_{2m+n+1},\dots,G_{2m+2n}$ return the same better neighbor solution if there is one and return the input of $X_1,\dots,X_n$ otherwise. The following two assumptions are made to simplify technical matters. Fourth, $I_1(G_i) = I_2(G_i) = X_i$ for all $N-n+1 \leq i \leq N$. Fifth and finally, $I_j(G_i) \neq X_k$ for all $1 \leq j \leq 2$, $1 \leq k \leq n$ and $1 \leq i \leq N-n$.

In the following, we describe a graph $G_C = (V_C, E_C)$ that contains only nodes of maximum degree five. In our description for $G_C = (V_C, E_C)$, we introduce several comparing nodes of degree greater than five. However, we assume that the graph $G_C = (V_C, E_C)$ is obtained by iteratively degrading the comparing nodes whereby we get a maximum degree of five for G_C. The proof will show two properties for local optima $P \in \mathscr{P}(V_C)$. First, every comparing node v is guided in P. Then the Substituting Lemma (i.e., Lemma 4.3.3) implies $col_P(v) \neq c_P(bn(v))$. Second, the colors of the nodes of G_C induce a local optimum for C.

The graph G_C consists of two isomorphic subgraphs G^0_C, G^1_C representing copies of C—the overall structure of the proof is inspired by Krentel [39]. For each gate G_i in C there is a subgraph S^κ_i for $\kappa \in \{0,1\}$ in G_C. The subgraphs S^κ_i are taken from Schäffer and Yannakakis [54] and adjusted such that they have maximum degree five without

changing local optima. In particular, each S_i^κ contains a comparing node g_i^κ whose color corresponds to the output of G_i.

We introduce nodes x_i^κ for $1 \leq i \leq n$ and call them the **input nodes** of G_C^κ. For $1 \leq i \leq N-n$, $i < j \leq N$, $1 \leq k \leq 2$ we let $\boldsymbol{I_k(g_i^\kappa)} := g_j^\kappa$ if $G_j = I_k(G_i)$ in C and for $N-n+1 \leq i \leq N$, $1 \leq j \leq 2$ we let $I_j(g_i^\kappa) := x_{i-N+n}^\kappa$. We let $w_{i,1}^\kappa := g_{2m+i}^\kappa, w_{i,2}^\kappa := g_{2m+n+i}^\kappa$ for $1 \leq i \leq n$ and $\hat{g}_i^\kappa := g_{m+i}^\kappa$ for $1 \leq i \leq m$. Moreover, we let w_j^κ for $1 \leq j \leq 2$ be the vector of nodes induced by $w_{i,j}^\kappa$ for $1 \leq i \leq m$. Each subgraph G_C^κ contains nodes y_i^κ, z_i^κ for $0 \leq i \leq 2N+1$ which induce vectors y^κ and z^κ.

For a partition $P \in \mathscr{P}(V_C)$ and $\kappa \in \{0,1\}$ we let $\boldsymbol{C_P(x^\kappa)}$ be the output of C on input $c_P(x^\kappa)$ and $\boldsymbol{w_P(x^\kappa)}$ be the better neighbor computed by C on input $c_P(x^\kappa)$. If the partition is clear from the context then we omit the subscript indicating the partition. We call the subgraph G_C^0 the **winner** and the subgraph G_C^1 the **loser** if $C(x^0) \geq C(x^1)$, otherwise we call G_C^0 the loser and G_C^1 the winner.

The proof in a nutshell: We show that the colors of the nodes of the subgraphs S_i^κ, in local optima, either correspond to the correct outputs of NOR-gates or have a reset state, i.e., a state in which each input node of S_i^κ is indifferent with respect to its neighbors in S_i^κ. For each $\kappa \in \{0,1\}$ we have a subgraph T^κ that looks at nodes that have, in local optima, the same colors as the nodes $w_{i,1}^\kappa$ and $w_{i,2}^\kappa$ for $1 \leq i \leq n$, i.e., the nodes that correspond to the gates that return the improving neighbor with respect to the given input, and biases each input node of $G_C^{\overline{\kappa}}$ to the color of its corresponding $w_{i,1}^\kappa$ and $w_{i,2}^\kappa$. Finally, we have a subgraph that looks at the input nodes of G_C^0, G_C^1, decides whose input results in a greater output with respect to C, and biases the subgraphs S_i^κ of the winner to behave like NOR-gates and the subgraphs of the loser to take the reset state. Then we show that the colors of the subgraphs S_i^κ of the winner G_C^κ for $\kappa \in \{0,1\}$ in fact reflect the correct outputs with respect to the colors of their inputs and that the input nodes of the loser in fact are indifferent with respect to their neighbors in the subgraphs $S_i^{\overline{\kappa}}$. Then, due to the bias of T^κ, the input nodes of the loser take the colors of the improving neighbor computed by the winner whereafter the loser becomes the new winner. Hence, the improving solutions switch back and forth between the two copies until the colors of the vectors of nodes of x^0 correspond to a local optimum for C. This finishes the "in a nutshell" description of the proof.

We introduce the nodes and edges of G_C via what we call components. A **component** of G_C is a tupel (V_C', E_C') with $V_C' \subseteq V_C$ and $E_C' \subseteq E_C$. The components of G_C have fifteen types: Type 1 up to Type 15, where we say that the nodes, edges and weights of the edges of the components have the same types as their corresponding components. We explicitly state weights for the edges of Type 2 up to Type 7. However, the weights of these components are stated only to indicate the relations between edge weights of the same type. The only edge weights that interleave between two different types are those of Type 3 and Type 4. The edges of Type 3 and Type 4 are scaled by the same number. For all other types we assume that their weights are scaled such that the weight of an edge of a given type is greater than eight times the sum of the weights of the edges of

higher types combined. Note that for these types, a lower type implies a higher edge weight. Moreover, we assume that the weights of the edges of Type 9 and higher are scaled such that each edge of one of these types is by the factor $8 \cdot b(4)$ smaller than any edge of Type 1 up to Type 8—for the definition of the function $b(\cdot)$ see Definition 4.3.1. To distinguish between the meaning of the explicitly stated edge weights and the final edge weights, i.e., the weights resulting by the scale, we call the explicitly stated weights **relative edge weights**.

The components of Type 9 up to Type 15 are subgraphs that look at certain nodes and bias other nodes. To some nodes more than one subgraph looks at. We assume that the component of the lowest type which looks at such a node v not only biases the nodes of which we state that it biases them but also biases extra nodes $v'_1, \dots, v'_k$, for $k \in \mathbb{N}$ great enough to the same color as v, and the components of higher types look at $v'_1, \dots, v'_k$ instead of the original nodes. In this way, it is ensured that to any node v to which more than one subgraph looks at, only one edge is incident that is an edge of the subgraphs that look at v.

The components of some types are introduced via drawings. In the drawings, the thick black edges and the nodes with black circumcircles are nodes counted among the components of the introduced type. Gray edges and nodes with gray circumcircles are of a different type than the component introduced in the corresponding drawing and are only (re-)drawn to simplify the verification of the proofs for the reader—in particular, the condition that each node is of maximum degree five. For comparing nodes, we only redraw such edges in the component in which the corresponding comparing node is introduced—these nodes have a degree that is greater than five anyway. If for a gray edge there is no explicit relative weight given, then the edge is among the types $8-14$. If a gray edge is dotted then it is of higher type than the non-dotted gray edges of the same drawing. If a node has a black or a white filling then it is of Type 1. These nodes are also (re-)drawn in components for Types higher than 1.

Type 1 contains nodes s, t which are connected by an edge whose weight is greater than the sum of the weights of all other edges in E_C. Assume without loss of generality $c(s) = 0$ and let S and T be the sets of nodes representing the constants 0 and 1. The component looks at s and biases the nodes of S to the color of s and the nodes of T to the opposite. In the following, we assume for each constant introduced in components of higher types, there is a separate node in the sets S, T. In the drawings that introduce the following components, nodes with a white filling are in the set S and nodes with a black filling are in T.

Comment Type 1 is to provide the constants 0 and 1 for the components of higher type.

Type 2 contains the nodes d^0, d^1, u^0, u^1—we will see later that d^0 and d^1 are comparing nodes—with edges and relative weights as depicted in Figure 4.4.

Comment The purpose of the edges of Type 2 is—together with the edges of Type 10 and Type 11—to guarantee that d^0 and d^1 are not both black in local

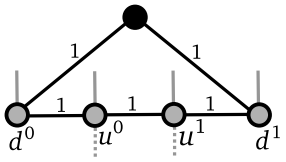


Figure 4.4: The component of Type 2,

optima. We will see in Lemma 4.4.4 that the nodes d^0 and d^1 are comparing nodes.

Type 3 consists of subgraphs S_i^κ which are to represent the gates G_i of C—see Figure 4.5. We call the edges $\{g_i^\kappa, u_{i,j}^\kappa\}$ for all $1 \leq i \leq N$, $j \in \{2,3,6,7,10,11\}$ **corresponding to** g_i^κ.

Comment The nodes d^0, d^1, g_i^κ (and $I_j(g_i^\kappa)$, respectively) and x_i^κ are the only nodes which have a degree greater than five—we will see later that they are comparing. The components of Type 3 to Type 7 are to represent the two subgraphs G_C^0 and G_C^1. The components are very similar to certain clauses of [54]. There are three differences between our components and their clauses. First, we omit some nodes and edges to obtain a maximum degree of five for all nodes unequal to g_i^κ, $I_1(g_i^\kappa)$ and $I_2(g_i^\kappa)$ for $1 \leq i \leq N$. Second, we use different edge weights. However, the weights are manipulated in a way such that the happiness of each node for given colors of the corresponding adjacent nodes is the same as in [54]. Third, we add nodes that we bias and which we look at. Their purpose is to derive the color that a comparing node g_i^κ would have in a local optimum. To this color node g_i^κ is biased.

Type 4 is depicted in Figure 4.6. As in [54] we say that the **natural value** of the nodes y_i^κ is 1 and the natural value of the nodes z_i^κ is 0.

Comment Type 4 checks whether the outputs of the gates represented by the components of Type 3 are correct and gives incentives to nodes of other components depending on the result. The nodes $y_{N+1}^\kappa, z_{N+1}^\kappa, \ldots, y_2^\kappa, z_2^\kappa$ check the correct computation of the corresponding gates and give incentives to their corresponding gates depending on whether the previous gates are correct. The nodes $y_1^\kappa, z_1^\kappa, y_0^\kappa, z_0^\kappa$ are to give incentives to d^0, d^1 depending on whether all gates are correct. Recall that the weights of the edges of Type 4 are the only weights that interleave with weights of edges of a higher type, namely with those of Type 3. For further comments, see comment of Type 3.

Type 5 contains the nodes and edges depicted in Figure 4.7 for $1 \leq i \leq m$ and the nodes and edges depicted in Figure 4.8.

Comment The aim of the component is twofold. On the one hand it is to incite that one of the nodes d^0 and d^1 to become black for which the output of the corresponding copy G_C^0 and G_C^1 is smaller and the other one to become white. On

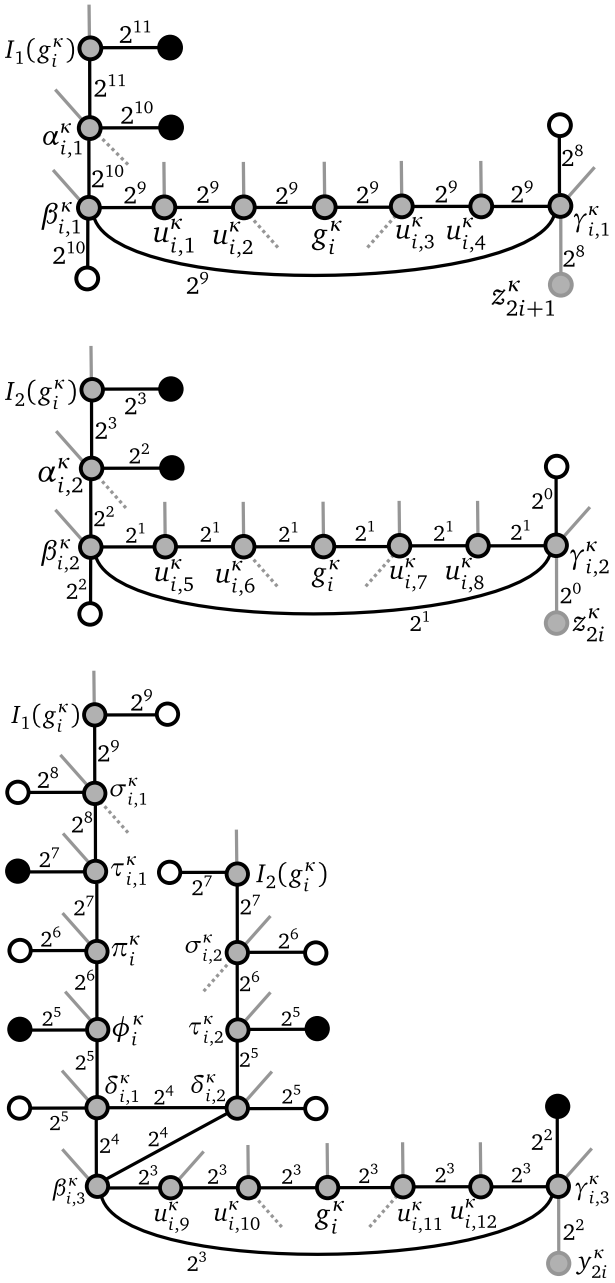


Figure 4.5: The components of Type 3; $1 \leq i \leq N$. Extra factor for relative edge weights: 2^{12i-2}.

the other hand, the edges $\{1, d^0\}$, $\{1, d^0\}$, $\{0, d^1\}$ and $\{0, d^1\}$ are to break the tie in favor of G_C^0 if the outputs of G_C^0 and G_C^1 are equal. For further comments, see comment of Type 3.

Type 6 contains nodes and edges as depicted in Figure 4.9.

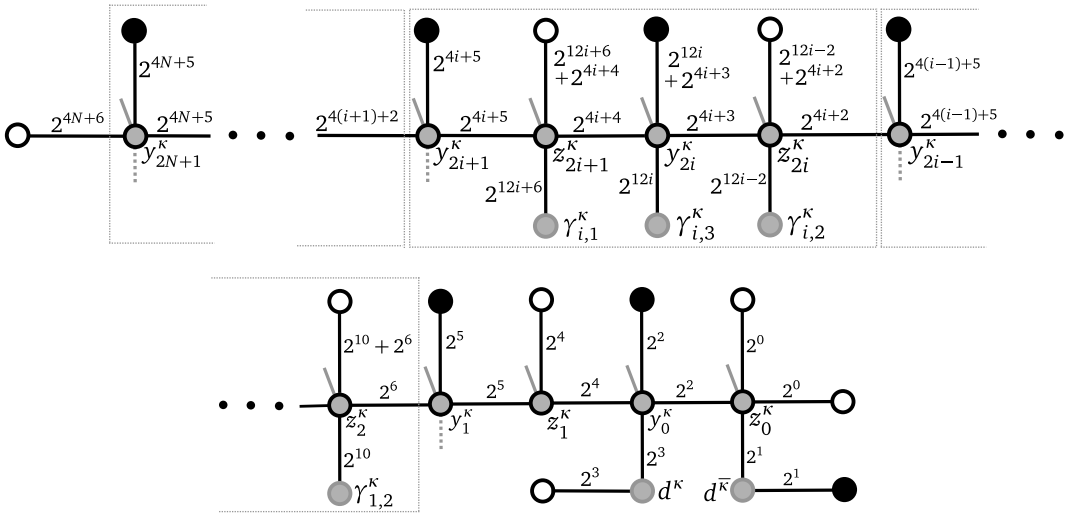


Figure 4.6: The components of Type 4.

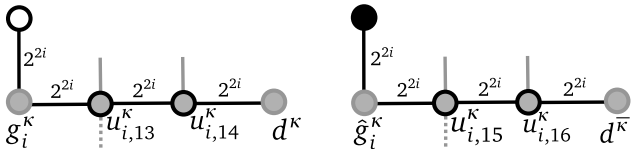


Figure 4.7: First part of the components of Type 5; $1 \leq i \leq m$.

Comment The edges $\{\hat{d}^\kappa_i, d^\kappa\}$ for $1 \leq i \leq n$ are to ensure that $col(d^\kappa) \neq c(\hat{d}^\kappa_i)$ for all i and the edges $\{1, d^\kappa\}$ for $1 \leq i \leq n$ are needed to make d^κ a comparing node. For further comments, see comment of Type 3.

Type 7 is depicted in Figure 4.10.

Comment Type 7 is to incite the nodes of the vector $\lambda^{\overline{\kappa}}$ to take the color corresponding to the better neighbor computed by G^κ_C if $col(d^\kappa) = 0$ and $col(w^\kappa_{i,1}) = col(w^\kappa_{i,2})$. On the other hand, if $col(d^\kappa) = 1$ then the component incites the nodes $\theta^\kappa_{i,1}$ and $\theta^\kappa_{i,2}$ to have the opposite color of η^κ_i whereafter a flip of a node $w^\kappa_{i,j}$ for $1 \leq j \leq 2$ does not decrease the cut by a weight of Type 7. For further comments, see comment of Type 3.

Type 8 looks for each $1 \leq i \leq n$ at the nodes λ^κ_i, $\alpha^\kappa_{N-n+i,1}$, $\alpha^\kappa_{N-n+i,2}$, $\sigma^\kappa_{N-n+i,1}$ and $\sigma^\kappa_{N-n+i,2}$ and biases x^κ_i as follows. The component computes whether x^κ_i is weakly

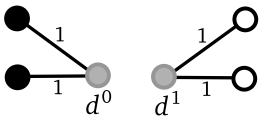


Figure 4.8: Second part of the components of Type 5.

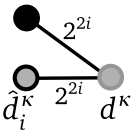


Figure 4.9: The component of Type 6;$1 \leq i \leq n$.

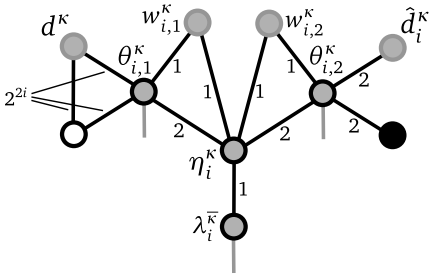


Figure 4.10: The components of Type 7; $1 \leq i \leq n$.

indifferent and it computes the color $\rho \in \{0,1\}$ node x_i^κ would have if it were not weakly indifferent. The component biases x_i^κ to ρ if it is not weakly indifferent and to the color of λ_i^κ otherwise.

Comment As we will see in Lemma 4.4.4, the nodes x_i^κ are comparing nodes. Thus, we have to ensure that in every local optimum P, node x_i^κ is guided. For this purpose, the component of Type 8 looks at all nodes that are adjacent to x_i^κ—except the constants—and biases it appropriately if it is not weakly indifferent. However, if it is weakly indifferent then it is biased to the color of λ_i^κ. As we will see later, x_i^κ is weakly indifferent when the subgraph G_C^κ is supposed to take the improving neighbor computed by $G_C^{\overline{\kappa}}$ as its input. In this case, the nodes of the vector λ^κ have the colors corresponding to the improving neighbor. Then the bias of Type 8 ensures that, due to the weak indifference of the nodes of x^κ, the nodes x^κ also take the colors corresponding to the improving neighbor.

Type 9 looks at the vectors x^0, x^1 of nodes representing the inputs of G_0^C and G_1^C and at the vectors λ^0, λ^1 and biases the vectors y^0, z^0, y^1 and z^1 in the following way. The nodes y_i^0, z_i^0 for all $0 \leq i \leq 2N+1$ are biased to their unnatural value, as defined in Type 4, if $C(x^0) < C(x^1)$, $w(x^1) = c(\lambda^0)$ and $c(\lambda^0) \neq col(x^0)$, and to their natural value otherwise. Similarly, y_i^1, z_i^1 are biased to their unnatural value if $C(x^0) \geq C(x^1)$, $w(x^0) = c(\lambda^1)$ and $c(\lambda^1) \neq col(x^1)$, and to their natural value otherwise.

Comment The comparison between $C(x^0)$ and $C(x^1)$ is used to decide which circuit is the winner and which one is the loser, and the consideration of the colors of the other nodes is to avoid certain troublemaking local optima. The nodes x_i^κ are the only comparing nodes to which a subgraph, namely Type 9, looks at.

Type 10 looks at y_1^0, y_1^1 and at the vectors x^0 and x^1 and biases u^0 and u^1 as follows.

If $C(x^0) \geq C(x^1)$ then it biases u^0 to the color of y_1^0 and u^1 to the opposite. Otherwise it biases u^1 to the color of y_1^1 and u^0 to the opposite.

Comment The idea behind the components of Type 10 and Type 11 is as follows. In any local optimum, we want for the nodes d^0 and d^1 at most one to be black. The natural idea to reach this is to use a simple edge between them in the component of Type 2 (see Figure 4.4) without the intermediate nodes u^0 and u^1. Recall that we have to ensure that the comparing node d^κ is biased to the color that it has in a local optimum. For this, we need to know the colors of the neighbors adjacent to d^κ via the edges of the highest weight, which includes the color of $d^{\overline{\kappa}}$. But biasing $d^{\overline{\kappa}}$ analogously needs the information about the color of d^κ. To solve this problem, we introduce the intermediate nodes u^0 and u^1, bias them appropriately and use their colors to bias d^0 and d^1.

Type 11 looks at u^0, u^1, y_1^0, y_1^1 and at the vectors x^0 and x^1 and biases d^0 and d^1 as follows. If $c(y_1^0) = c(y_1^1) = 0$ then d^0 is biased to the color of u^1 and d^1 to the color of u^0. If $c(y_1^0) \neq c(y_1^1)$ then d^0 is biased to the color of y_1^1 and d^1 to the opposite. If $c(y_1^0) = c(y_1^1) = 1$ then we distinguish two cases. If $C(x^0) \geq C(x^1)$ then d^0 is biased to 0 and d^1 to 1, otherwise d^0 to 1 and d^1 to 0.

Comment See comment of Type 10.

Type 12 looks at y_{2i+1}^κ for $1 \leq i \leq N$ and biases $\alpha_{i,1}^\kappa$, $\alpha_{i,2}^\kappa$, $\gamma_{i,1}^\kappa$, $\gamma_{i,2}^\kappa$, $\beta_{i,3}^\kappa$, $\tau_{i,1}^\kappa$, $\tau_{i,2}^\kappa$ and ϕ_i^κ to the color of y_{2i+1}^κ and $\beta_{i,1}^\kappa, \beta_{i,2}^\kappa, \gamma_{i,3}^\kappa, \sigma_{i,1}^\kappa, \sigma_{i,2}^\kappa$, π_i^κ, $\delta_{i,1}^\kappa$ and $\delta_{i,2}^\kappa$ to the opposite.

Comment Type 12 is to bias the nodes of Type 3 to certain preferred colors depending on whether y_{2i+1}^κ has its natural value. If it has its natural value then it biases the subgraph S_i^κ to colors which reflect the behavior of a NOR-gate for S_i^κ, otherwise it biases them such that the input nodes $I_1(g_i^\kappa)$ and $I_2(g_i^\kappa)$ are indifferent with respect to their neighbors in S_i^κ.

Type 13 looks at $y_{2i+1}^\kappa, y_{2i-1}^\kappa, \alpha_{i,1}^\kappa$ and $\alpha_{i,2}^\kappa$ and biases $u_{i,1}^\kappa$, $u_{i,3}^\kappa$, $u_{i,5}^\kappa$, $u_{i,7}^\kappa$, $u_{i,10}^\kappa$, $u_{i,12}^\kappa$ to white and $u_{i,2}^\kappa, u_{i,4}^\kappa, u_{i,6}^\kappa, u_{i,8}^\kappa, u_{i,9}^\kappa, u_{i,11}^\kappa$ to black if $c(y_{2i+1}^\kappa) = c(y_{2i-1}^\kappa)$. Otherwise, $u_{i,3}^\kappa$, $u_{i,4}^\kappa$, $u_{i,7}^\kappa$, $u_{i,8}^\kappa$, $u_{i,11}^\kappa$, $u_{i,12}^\kappa$ are biased to their corresponding opposite and the biases of the remaining nodes split into the following cases. Node $u_{i,1}^\kappa$ is biased to $c(\alpha_{i,1}^\kappa)$ and $u_{i,2}^\kappa$ to the opposite. Similarly, $u_{i,5}^\kappa$ is biased to $c(\alpha_{i,2}^\kappa)$ and $u_{i,6}^\kappa$ to the opposite. Finally, $u_{i,9}^\kappa$ is biased to $c(\alpha_{i,2}^\kappa) \wedge c(\alpha_{i,2}^\kappa)$ and $u_{i,10}^\kappa$ to the opposite.

Comment The aim of the components of Type 13 up to Type 15 is to bias every comparing node g_i^κ such that it is guided in every local optimum P. To reach this, we need to know the colors of the nodes adjacent to g_i^κ. Thus, we introduce—similarly as in the component of Type 2—extra nodes $u_{i,j}^\kappa$, bias them appropriately, introduce a component that looks at the nodes $u_{i,j}^\kappa$ and use their colors to bias the nodes g_i^κ such that they are guided.

Type 14 looks for all $1 \leq i \leq m$ at $y_{2i-1}^\kappa, \alpha_{i,1}^\kappa$ and $\alpha_{i,2}^\kappa$ and biases $u_{i,14}^\kappa$ to $c(y_{2i-1}^\kappa) \wedge c(\alpha_{i,1}^\kappa) \wedge c(\alpha_{i,2}^\kappa)$ and $u_{i,13}^\kappa$ to the opposite. Similarly, it looks for all $m+1 \leq i \leq 2m$

at $y^\kappa_{2i-1}, \alpha^\kappa_{i,1}$ and $\alpha^\kappa_{i,2}$ and biases $u^\kappa_{i-m,15}$ to $c(y^\kappa_{2i-1}) \wedge (\neg c(\alpha^\kappa_{i,1}) \vee \neg c(\alpha^\kappa_{i,2}))$ and $u^\kappa_{i-m,16}$ to the opposite—recall that $\hat{g}^\kappa_i = g^\kappa_{m+i}$ for all $1 \leq i \leq m$, $\kappa \in \{0,1\}$.

Comment See comment of Type 13.

Type 15 looks at all nodes of type lower than Type 15 that are adjacent to g^κ_i with the single exception of η^κ_i if $g^\kappa_i = w^\kappa_{j,k}$. Namely, it looks at $u^\kappa_{i,4j+2}, u^\kappa_{i,4j+3}$ for $0 \leq j \leq 2$, at $\alpha^\kappa_{j,k}$ and $\sigma^\kappa_{j,k}$ if $I_k(g_j)^\kappa = (g^\kappa_i)$ for $k \in \{1,2\}$, at $u^\kappa_{i,13}$ if $i \leq m$, at $u^\kappa_{i-m,15}$ if $m+1 \leq i \leq 2m$. Furthermore, it looks at λ^κ_i if $g^\kappa_i = w^\kappa_{j,k}$, at $\alpha^\kappa_{i,1}$ and $\alpha^\kappa_{i,2}$. The component treats the color of λ^κ_i as if it were the opposite color of η^κ_i if $g^\kappa_i = w^\kappa_{j,k}$—we will see in Lemma 4.4.3 that $c(\eta^\kappa_i) \neq c(\lambda^{\overline{\kappa}}_i)$ in any local optimum. Then the component computes whether g^κ_i is weakly indifferent and computes the color $\rho \in \{0,1\}$ of its decisive neighbors if it is not weakly indifferent. The component biases g^κ_i to $\overline{\rho}$ if g^κ_i is not weakly indifferent. If g^κ_i is weakly indifferent then it biases g^κ_i to $c(\alpha^\kappa_{i,1}) \wedge c(\alpha^\kappa_{i,2})$.

Comment See comment of Type 13.

This finishes the description of G_C. For an overview showing which nodes a given type of component biases see Table 4.1.

Type	Biases	Condition
8	x^κ_i	$\kappa \in \{0,1\}$, $0 \leq i \leq n$
9	y^κ_i, z^κ_i	$\kappa \in \{0,1\}$, $0 \leq i \leq 2N+1$
10	u^κ	$\kappa \in \{0,1\}$
11	d^κ	$\kappa \in \{0,1\}$
11	$\alpha^\kappa_{i,j}$, $\beta^\kappa_{i,k}$, $\gamma^\kappa_{i,k}$, $\sigma^\kappa_{i,k}$ $\tau^\kappa_{i,j}$, π^κ_i, ϕ^κ_i, $\delta^\kappa_{i,j}$	$\kappa \in \{0,1\}$, $1 \leq i \leq N$, $1 \leq j \leq 2$, $1 \leq k \leq 3$
13	$u^\kappa_{i,j}$	$\kappa \in \{0,1\}$, $1 \leq i \leq N$, $1 \leq j \leq 12$
14	$u^\kappa_{i,j}$	$\kappa \in \{0,1\}$, $1 \leq i \leq N$, $13 \leq j \leq 16$
15	g^κ_i	$\kappa \in \{0,1\}$, $1 \leq i \leq N$

Table 4.1: Relation between types of components and nodes that they bias.

Now we consider the colors of the nodes of G_C in an arbitrary local optimum. All the remaining Lemmas are assumed to have an inherent statement "for any local optimum P". We call a gate g^κ_i **correct** if $col(g^\kappa_i) = \neg(col(I_1(g^\kappa_i)) \vee col(I_2(g^\kappa_i))$. The following Lemmas characterize properties of some components.

Lemma 4.4.3. *$c(u^0) \neq c(u^1)$ and $c(\eta^\kappa_i) \neq c(\lambda^{\overline{\kappa}}_i)$ for all $1 \leq i \leq n$.*

Proof. Due to the weights of the edges incident to u^0 and u^1 and since they are biased to different colors by Type 10, in each local optimum at least one of them is unhappy if both have the same color. Thus, we get $c(u^0) \neq c(u^1)$.

The claim $c(\eta^\kappa_i) \neq c(\lambda^{\overline{\kappa}}_i)$ follows directly from the weights of the edges incident to $\lambda^{\overline{\kappa}}_i$—see Figure 4.10. □

Lemma 4.4.4. *For all $1 \leq i \leq N$, $1 \leq j \leq n$, $\kappa \in \{0,1\}$ the nodes d^0, d^1, g_i^κ and x_j^κ are comparing nodes.*

Proof. In Table 4.2 and Table 4.3 we name all nodes adjacent to d^0, d^1, g_i^κ and x_j^κ for all $1 \leq i \leq N$, $1 \leq j \leq n$, $\kappa \in \{0,1\}$ and the weights of the corresponding edges. By means of Table 4.2 it can be verified that the nodes d^0, d^1, g_i^κ are comparing.

Now consider the nodes x_j^κ for $1 \leq j \leq n$. Our assumption that the edges of Type 9 and higher are scaled such that each edge of one of these types is by the factor $8 \cdot b(4)$ smaller than any edge of lower type ensures that condition (iii) of Definition 4.3.2 is satisfied for x_j^κ. The remaining properties needed for x_j^κ to be a comparing node can be verified by means of Table 4.3. □

Lemma 4.4.5. *For all $1 \leq i \leq N$ we have either $col(g_i^\kappa) = 0$ or $col(g_i^\kappa) = 1$ and for all $1 \leq j \leq n$ we have either $col(x_j^\kappa) = 0$ or $col(x_j^\kappa) = 1$.*

Proof. We first consider the nodes g_i^κ. The nodes $\eta_i^{\overline{\kappa}}$ for $g_j^\kappa = w_{i,k}^\kappa$ and any $k \in \{1,2\}$ are the only nodes (apart from the constants) adjacent to node g_j^κ to which Type 15, i.e., the component that biases g_i^κ, does not look at. From Lemma 4.4.3, we know that $c(\eta_i^\kappa) \neq c(\lambda_i^{\overline{\kappa}})$. No node adjacent to g_i^κ is a comparing node—see Table 4.2. Moreover, no node to which the component of Type 15 looks at is a comparing node. Thus, the color of the decisive neighbors of g_i^κ—should g_i^κ not be weakly indifferent—and the colors of the nodes to which the component of Type 15 looks at are uniquely determined in P. Consequently, the component of Type 15 correctly decides whether g_i^κ is weakly indifferent as presented in the description of Type 15 and biases it to the opposite color of its decisive neighbors in this case. Therefore, $bn(g_i^\kappa)$ has the same color as the decisive neighbors of g_i^κ which implies that g_i^κ is guided. Hence, the Substituting Lemma (i.e., Lemma 4.3.3) implies that either $col(g_i^\kappa) = 0$ or $col(g_i^\kappa) = 1$.

Now we consider the nodes x_i^κ. No node adjacent to g_i^κ is a comparing node—see Table 4.2. Moreover, no node to which the component of Type 8 looks at, i.e., the component that biases x_i^κ, is a comparing node. Thus, the color of the decisive neighbors of x_i^κ—should x_i^κ not be weakly indifferent—and the colors of the nodes to which the component of Type 8 looks at are uniquely determined in P. Consequently, the component of Type 5 correctly decides whether x_i^κ is weakly indifferent as presented in the description of Type 8 and biases it to the opposite color of its decisive neighbors in this case. Therefore, $bn(x_i^\kappa)$ has the same color as the decisive neighbors of x_i^κ, which implies that x_i^κ is guided. Hence, the Substituting Lemma implies that either $col(x_i^\kappa) = 0$ or $col(x_i^\kappa) = 1$. □

Comment Later in the proof, we also show that either $col(d^\kappa) = 0$ or $col(d^\kappa) = 1$ for $\kappa \in \{0,1\}$.

Lemma 4.4.6 (similar to Claims 5.9.B and 5.10.B in [54]). *If $col(d^\kappa) = 1$ then neither flipping $w_{i,1}^\kappa$ nor $w_{i,2}^\kappa$ changes the cut by a weight of Type 7. If $col(d^\kappa) = 0$ and $col(w_{i,1}^\kappa) = col(w_{i,2}^\kappa)$ then $col(w_{i,1}^\kappa) \neq c(\eta_i^\kappa)$.*

Node	Neighbor	Type	R. Weight	Condition
d^κ	1 u^κ	2	1	
	0 y_0^κ	4	2^3	
	1 $z_0^{\overline{\kappa}}$		2^1	
	$u_{i,14}^\kappa$ $u_{i,16}^{\overline{\kappa}}$	5	2^{2i}	$1 \le i \le m$
	κ κ		1	
	1 $\hat{d}_i^\kappa$	6	2^{2i}	$1 \le i \le n$
	$\theta_{i,1}^\kappa$ 0	7	2^{2i}	$1 \le i \le m$
	$bn(d^\kappa)$	11	1	

Node	Condition	Neighbor	Type	R. Weight	Condition
g_i^κ	$1 \le i \le N$	$u_{i,2}^\kappa$ $u_{i,3}^\kappa$	3	2^{12i+7}	
		$u_{i,10}^\kappa$ $u_{i,11}^\kappa$		2^{12i+1}	
		$u_{i,6}^\kappa$ $u_{i,7}^\kappa$		2^{12i-1}	
	$2(n+m)+1$ $\le i \le N$	1 $\alpha_{j,1}^\kappa$		2^{12j+9}	$I_1(g_j) = g_i$
		0 $\sigma_{j,1}^\kappa$		2^{12j+7}	
		0 $\sigma_{j,2}^\kappa$		2^{12j+5}	$I_2(g_j) = g_i$
		1 $\alpha_{j,2}^\kappa$		2^{12j+1}	
	$1 \le i \le m$	0 $u_{i,13}^\kappa$	5	2^{2i}	
	$m+1 \le i \le 2m$	1 $u_{i-m,15}^\kappa$		$2^{2(i-m)}$	
	$2m+n+1 \le$ $i \le 2(m+n)$	$\theta_{j,2}^\kappa$ η_j^κ	7	2	$j := i-2m-n$
	$2m+1 \le$ $i \le 2m+n$	$\theta_{j,1}^\kappa$ η_j^κ			
		$bn(g_i^\kappa)$	15	1	

Table 4.2: Neighborhood of the nodes d^0, d^1 and g_i^κ for $1 \le i \le N$, $\kappa \in \{0,1\}$.

Proof. The proof uses the following claim.

Claim 4.4.7. *If $col(d^\kappa) = \rho$ for $\rho \in \{0,1\}$ then $c(\hat{d}_i^\kappa) = \overline{\rho}$ for all $1 \le i \le n$.*

Proof. There are three edges incident to each node $\hat{d}_i^\kappa$ as introduced in Type 6. Namely, one edge of Type 6 and two edges of Type 7. Since the weight of the edge of Type 6 is greater than the sum of the weights of all edges of higher type, in particular the two edges of Type 7, the claim follows. □

Assume $col(d^\kappa) = 1$. Then, by Claim 4.4.7, we have $c(\hat{d}_i^\kappa) = 0$ for all i. Since $col(d^\kappa) = 1$, the weights of the five edges incident to $\theta_{i,1}^\kappa$ depicted in Figure 4.10 imply $c(\theta_{i,1}^\kappa) \neq c(\eta_i^\kappa)$. Similarly, we can argue that $c(\theta_{i,2}^\kappa) \neq c(\eta_i^\kappa)$. But then neither a flip of $w_{i,1}^\kappa$ nor a flip of $w_{i,2}^\kappa$ can change the cut by a weight of Type 7.

Now assume $col(d^\kappa) = 0$ and $col(w_{i,1}^\kappa) = col(w_{i,2}^\kappa)$. Due to Claim 4.4.7 we have $c(\hat{d}_i^\kappa) = 1$ for all i. The weights of the edges incident to $\theta_{i,1}^\kappa$ and $\theta_{i,2}^\kappa$ imply $c(\theta_{i,1}^\kappa) = 1$ and $c(\theta_{i,2}^\kappa) = 0$. Since $col(w_{i,1}^\kappa) = col(w_{i,2}^\kappa)$ and $c(\theta_{i,1}^\kappa) \neq c(\theta_{i,2}^\kappa)$, node η_i^κ is happy if and only if its color is unequal to the color of $w_{i,1}^\kappa$ and $w_{i,2}^\kappa$. □

Lemma 4.4.8 (similar to Lemma 4.1H in [54]). *If $c(z_j^\kappa) = 1$ then $c(y_{j-1}^\kappa) = 0$. If $c(y_j^\kappa) = 0$ then $c(y_p^\kappa) = 0$ and $c(z_p^\kappa) = 1$ for all $p \le j$.*

Proof. The sum of the weights of the edges $\{z_j^\kappa, y_{j-1}^\kappa\}$ and $\{y_{j-1}^\kappa, 1\}$ is greater than the sum of all other edges incident to y_{j-1}^κ. Thus, if $c(z_j^\kappa) = 1$ then $c(y_{j-1}^\kappa) = 0$. Similarly, we can argue that $c(z_p^\kappa) = 1$ if $c(y_p^\kappa) = 0$ has its unnatural value. Therefore, the claim follows by induction. □

Lemma 4.4.9 (similar to Lemma 4.1 in [54]). *If g_i^κ is not correct then $c(z_{2i}^\kappa) = 1$.*

Proof. The proof uses the following claims.

Claim 4.4.10. *If $c(z_{2i}^\kappa) = 0$ then $c(y_{2i-1}^\kappa) = 1$.*

Node	Neighbor	Type	R. Weight	Condition
x_i^κ	1 $\alpha_{i,1}^\kappa$	3	2^{12j+9}	$j := N - n + i$
	0 $\sigma_{i,1}^\kappa$		2^{12j+7}	
	0 $\sigma_{i,2}^\kappa$		2^{12j+5}	
	1 $\alpha_{i,2}^\kappa$		2^{12j+1}	
	$bn(x_i^\kappa)$	8	1	
	$ln(x_i^\kappa)$	9	1	

Table 4.3: Neighborhood of the nodes x_i^κ for $1 \le i \le n$, $\kappa \in \{0,1\}$.

Proof. Assume $c(z^\kappa_{2i}) = c(y^\kappa_{2i-1}) = 0$. If y^κ_{2i-1} is biased to black by the component of Type 9 then $c(y^\kappa_{2i-1}) = 1$ since $c(z^\kappa_{2i}) = 0$, which is a contradiction. Thus, y^κ_{2i-1} is biased to 0. Since z^κ_{2i} and y^κ_{2i-1} are biased to opposite colors by Type 9, node z^κ_{2i} is biased to 1. Due to the weight of its incident edges it cannot be white then, but this is a contradiction. □

Claim 4.4.11. *If* $col(I_1(g^\kappa_i)) = 1$ *and* $col(g^\kappa_i) = 1$ *then* $c(z^\kappa_{2i}) = 1$. *If* $col(I_2(g^\kappa_i)) = 1$ *and* $col(g^\kappa_i) = 1$ *then* $c(z^\kappa_{2i}) = 1$.

Proof. Assume for the sake of contradiction $col(I_1(g^\kappa_i)) = 1$, $col(g^\kappa_i) = 1$ but $c(z^\kappa_{2i}) = 0$. Claim 4.4.10 implies $c(y^\kappa_{2i-1}) = 1$ since $c(z^\kappa_{2i}) = 0$. Moreover, Lemma 4.4.8 implies $c(y^\kappa_{2i+1}) = 1$ since $c(z^\kappa_{2i}) = 0$. Thus, $c(y^\kappa_{2i+1}) = c(y^\kappa_{2i-1})$ and therefore the nodes $u^\kappa_{i,3}$ and $u^\kappa_{i,4}$ are biased to 0 and 1, respectively, by the component of Type 13. Then $c(u^\kappa_{i,3}) = 0$ and therefore $c(u^\kappa_{i,4}) = 1$. Due to $col(I_1(g^\kappa_i)) = 1$ we have $c(\alpha^\kappa_{i,1}) = 0$ and therefore $c(\beta^\kappa_{i,1}) = 1$ which implies $c(\gamma^\kappa_{i,1}) = 0$. Consequently, $c(z^\kappa_{2i+1}) = 1$ according to the weights of the edges incident to $c(z^\kappa_{2i+1}) = 1$ and then $c(z^\kappa_{2i}) = 1$ due to Lemma 4.4.8, which is a contradiction. The proof for $col(I_2(g^\kappa_i)) = 1$ is analogous. □

Claim 4.4.12. *If* $col(I_1(g^\kappa_i)) = 0$ *then* $c(\delta^\kappa_{i,1}) = 1$. *If* $col(I_2(g^\kappa_i)) = 0$ *then* $c(\delta^\kappa_{i,2}) = 1$.

Proof. If $col(I_1(g^\kappa_i)) = 0$ then $c(\sigma^\kappa_{i,1}) = 1$ since the edges $\{I_1(g^\kappa_i), \sigma^\kappa_{i,1}\}$ and $\{\sigma^\kappa_{i,1}, 0\}$ combined weigh more than the sum of all other edges incident to $\sigma^\kappa_{i,1}$. Due to the weight of the incident edges, it follows that $c(\tau^\kappa_{i,1}) = 0$, then $c(\pi^\kappa_i) = 1$, then $c(\phi^\kappa_i) = 0$ and then $c(\delta^\kappa_{i,1}) = 1$. Similarly, $col(I_2(g^\kappa_i)) = 0$ implies $c(\delta^\kappa_{i,2}) = 1$. □

Claim 4.4.13. *If* $col(I_1(g^\kappa_i)) = col(I_2(g^\kappa_i)) = 0$ *then* $c(\beta^\kappa_{i,3}) = 0$.

Proof. Due to Claim 4.4.12, $c(\delta^\kappa_{i,1}) = c(\delta^\kappa_{i,2}) = 1$. Since the sum of the weights of the edges $\{\beta^\kappa_{i,3}, \delta^\kappa_{i,1}\}$ and $\{\beta^\kappa_{i,3}, \delta^\kappa_{i,2}\}$ is greater than the sum of all other edges incident to $\beta^\kappa_{i,3}$, the claim follows. □

Claim 4.4.14. *Suppose* $col(I_1(g^\kappa_i)) = col(I_2(g^\kappa_i)) = col(g^\kappa_i) = 0$. *Then* $c(z^\kappa_{2i}) = 1$.

Proof. Assume for the sake of contradiction $c(z^\kappa_{2i}) = 0$. Then Lemma 4.4.8 implies $c(y^\kappa_{2i+1}) = 1$ since $c(z^\kappa_{2i}) = 0$. Moreover, Claim 4.4.10 implies $c(y^\kappa_{2i-1}) = 1$. Thus, $c(y^\kappa_{2i+1}) = c(y^\kappa_{2i-1})$ and therefore the nodes $u^\kappa_{i,11}$ and $u^\kappa_{i,12}$ are biased to 1 and 0, respectively, by the component of Type 13. Then $c(u^\kappa_{i,11}) = 1$ and therefore $c(u^\kappa_{i,12}) = 0$. But then Claim 4.4.13 implies $c(\beta^\kappa_{i,3}) = 0$ whereafter we get $c(\gamma^\kappa_{i,3}) = 1$. Consequently, $c(y^\kappa_{2i}) = 0$ according to the weights of the edges incident to y^κ_{2i} and then $c(z^\kappa_{2i}) = 1$ due to Lemma 4.4.8, which is a contradiction. Thus, the claim follows. □

If $col(I_1(g^\kappa_i)) = 1$ or $col(I_2(g^\kappa_i)) = 1$ then $c(z^\kappa_{2i}) = 1$ follows from Claim 4.4.11. If $col(I_1(g^\kappa_i)) = col(I_2(g^\kappa_i)) = 0$ then $c(z^\kappa_{2i}) = 1$ follows from Claim 4.4.14. □

Lemma 4.4.15 (partially similar to Lemma 4.2 in [54]). *If* $c(y^\kappa_{2i+1}) = 0$ *then* $c(\alpha^\kappa_{i,1}) = c(\alpha^\kappa_{i,2}) = 0$ *and* $c(\sigma^\kappa_{i,1}) = c(\sigma^\kappa_{i,2}) = 1$.

Proof. Assume $c(y^\kappa_{2i+1}) = 0$. From Lemma 4.4.8 we know that $c(z^\kappa_{2i+1}) = c(z^\kappa_{2i}) = 1$ and $c(y^\kappa_{2i}) = c(y^\kappa_{2i-1}) = 0$. From the component of Type 12 node $\beta^\kappa_{i,1}$ is biased to 1 and $\gamma^\kappa_{i,1}$ is biased to 0. From Lemma 4.4.8 we know that $c(y^\kappa_{2i+1}) = c(y^\kappa_{2i-1}) = 0$ and $c(z^\kappa_{2i+1}) = c(z^\kappa_{2i}) = 1$. Thus, the nodes $u^\kappa_{i,1}$ and $u^\kappa_{i,3}$ are biased to white and $u^\kappa_{i,2}$ and $u^\kappa_{i,4}$ to black by the component of Type 13.

Now we show that $c(\beta^\kappa_{i,1}) = 1$ and $c(\gamma^\kappa_{i,1}) = 0$. Assume first that $col(g^\kappa_i) = 0$. Then $c(u^\kappa_{i,2}) = 1$ and $c(u^\kappa_{i,1}) = 0$. Thus, $c(\beta^\kappa_{i,1}) = 1$ and therefore $c(\gamma^\kappa_{i,1}) = 0$. Now assume that $col(g^\kappa_i) = 1$. Then $c(u^\kappa_{i,3}) = 0$ and $c(u^\kappa_{i,4}) = 1$. Thus, $c(\gamma^\kappa_{i,1}) = 0$ and therefore $c(\beta^\kappa_{i,1}) = 1$.

Since $c(\beta^\kappa_{i,1}) = 1$, node $\alpha^\kappa_{i,1}$ must be white since it is biased to white by Type 12. The proof for $\alpha^\kappa_{i,1}, \beta^\kappa_{i,2}$ and $\gamma^\kappa_{i,2}$ is analogous.

Type 12 biases $\gamma^\kappa_{i,3}$, $\delta^\kappa_{i,1}$, $\delta^\kappa_{i,2}$, $\sigma^\kappa_{i,1}$, $\sigma^\kappa_{i,2}$ and π^κ_i to black and $\beta^\kappa_{i,3}$, $\tau^\kappa_{i,1}$, $\tau^\kappa_{i,2}$ and ϕ^κ_i to white. Due to $c(y^\kappa_{2i+1}) = c(y^\kappa_{2i-1}) = 0$ the nodes $u^\kappa_{i,9}$ and $u^\kappa_{i,11}$ are biased to black and $u^\kappa_{i,10}$ and $u^\kappa_{i,12}$ to white by the component of Type 13.

Now we show that $c(\beta^\kappa_{i,3}) = 0$ and $c(\gamma^\kappa_{i,3}) = 1$. Assume first that $c(\delta^\kappa_{i,1}) = c(\delta^\kappa_{i,2}) = 0$. Then both nodes $\delta^\kappa_{i,1}$ and $\delta^\kappa_{i,2}$ are unhappy. Therefore, we may assume that at least one of them is black. If $c(\beta^\kappa_{i,3}) = c(\gamma^\kappa_{i,3}) = 1$ then $\beta^\kappa_{i,3}$ is unhappy. Now assume $c(\beta^\kappa_{i,3}) = c(\gamma^\kappa_{i,3}) = 0$. Then node $\gamma^\kappa_{i,3}$ is unhappy since $c(y_{2i}) = 0$ has its unnatural value due to Lemma 4.4.8. Now assume $c(\beta^\kappa_{i,3}) = 1$ and $c(\gamma^\kappa_{i,3}) = 0$. If $col(g^\kappa_i) = 0$ then $c(u^\kappa_{i,11}) = 1$ and $c(u^\kappa_{i,12}) = 0$, which is a contradiction since $\gamma^\kappa_{i,3}$ is unhappy in this case. But if $col(g^\kappa_i) = 1$ then $c(u^\kappa_{i,10}) = 0$ and $c(u^\kappa_{i,9}) = 1$, which is also a contradiction since $\beta^\kappa_{i,3}$ is unhappy in this case. Thus, $c(\beta^\kappa_{i,3}) = 0$ and $c(\gamma^\kappa_{i,3}) = 1$.

Since $c(\beta^\kappa_{i,3}) = 0$, we get $c(\delta^\kappa_{i,1}) = c(\delta^\kappa_{i,2}) = 1$. Then a sequence of implications leads to $c(\phi^\kappa_i) = c(\tau^\kappa_{i,2}) = 0$, $c(\pi^\kappa_i) = c(\sigma^\kappa_{i,2}) = 1$, $c(\tau^\kappa_{i,1}) = 0$ and then $c(\sigma^\kappa_{i,1}) = 1$. □

Lemma 4.4.16 (partially similar to Lemma 4.3 in [54]). *Assume that $c(y^\kappa_{2i+1}) = 1$ and $c(y^\kappa_{2i-1}) = 0$. If g^κ_i is correct then z^κ_{2i}, z^κ_{2i+1} and y^κ_{2i} have the colors to which they are biased by Type 9. If g^κ_i is not correct then flipping g^κ_i does not decrease the cut by a weight of an edge of Type 3 corresponding to g^κ_i and increases it by a weight of an edge of Type 15.*

Proof. The proof uses the following three claims.

Claim 4.4.17. *Assume that $c(y^\kappa_{2i+1}) = 1$. Then $c(\alpha^\kappa_{i,1}) = \neg col(I_1(g^\kappa_i))$ and $c(\alpha^\kappa_{i,2}) = \neg col(I_2(g^\kappa_i))$. If, in addition, $c(y^\kappa_{2i-1}) = 0$ then $c(\beta^\kappa_{i,1}) = col(I_1(g^\kappa_i))$ and $c(\beta^\kappa_{i,2}) = col(I_2(g^\kappa_i))$.*

Proof. If $col(I_1(g^\kappa_i)) = 1$ then $c(\alpha^\kappa_{i,1}) = 0$. If, on the other hand, $col(I_1(g^\kappa_i)) = 0$ then $c(\alpha^\kappa_{i,1}) = 1$ since $\alpha^\kappa_{i,1}$ is biased to 1 by Type 12.

Now assume additionally $c(y^\kappa_{2i-1}) = 0$. If $col(I_1(g^\kappa_i)) = 1$ then $c(\beta^\kappa_{i,1}) = 1$ since $c(\alpha^\kappa_{i,1}) = 0$. Now assume $col(I_1(g^\kappa_i)) = 0$. Due to $c(\alpha^\kappa_{i,1}) = 1$ and since $\beta^\kappa_{i,1}$ is biased to 0 by Type 12, it can only be black if $\gamma^\kappa_{i,1}$ and $u^\kappa_{i,1}$ are both white. But if $\gamma^\kappa_{i,1}$ is white then $u^\kappa_{i,4}$ must be black since $\gamma^\kappa_{i,1}$ is biased to black by Type 12. If $col(g^\kappa_i) = 1$ then $c(u^\kappa_{i,2}) = 0$ and $c(u^\kappa_{i,1}) = 1$ due to the bias of Type 13, which is a contradiction. On the other hand, if $col(g^\kappa_i) = 0$ then $c(u^\kappa_{i,3}) = 1$ and $c(u^\kappa_{i,4}) = 0$ due to the bias of Type 13, which is also a contradiction. Thus, $c(\beta^\kappa_{i,1}) = 0$.

The argumentation for $\alpha^\kappa_{i,2}$ and $\beta^\kappa_{i,2}$ is analogous. □

Claim 4.4.18. *Assume* $c(y^\kappa_{2i+1}) = 1$ *and* $c(y^\kappa_{2i-1}) = 0$. *Then* $c(\beta^\kappa_{i,3}) = col(I_1(g^\kappa_i)) \vee col(I_2(g^\kappa_i))$.

Proof. If an input is white then the corresponding $\delta^\kappa_{i,j}$ is black due to Claim 4.4.12. Thus, if both inputs are white then $\beta^\kappa_{i,3}$ is white.

Now assume that at least one input is black. Assume first $I_1(g^\kappa_i) = 1$. Since $\sigma^\kappa_{i,1}$ is biased to white, we have $c(\sigma^\kappa_{i,1}) = 0$. Analogously, we get $c(\tau^\kappa_{i,1}) = 1$, $c(\pi^\kappa_i) = 0$ and $c(\phi^\kappa_i) = 1$. Node $\delta^\kappa_{i,1}$ is biased to white by Type 12. If both nodes $\delta^\kappa_{i,1}$ and $\delta^\kappa_{i,2}$ are black then $\delta^\kappa_{i,1}$ is unhappy. Thus, we may assume that at least one of them is white. Since $\beta^\kappa_{i,3}$ is biased to 1 by Type 12, it can only be white if $\gamma^\kappa_{i,3}$ and $u^\kappa_{i,9}$ are both black. But if $\gamma^\kappa_{i,3}$ is black then $u^\kappa_{i,12}$ must be white since $\gamma^\kappa_{i,3}$ is biased to white by Type 12. Then, similarly as in the proof of Claim 4.4.17, the bias of Type 13 implies that if g^κ_i is white then $u^\kappa_{i,10}$ is black and $u^\kappa_{i,9}$ is white, and if g^κ_i is black then $u^\kappa_{i,11}$ is white and $u^\kappa_{i,12}$ is black, each resulting in a contradiction. Thus, $c(\beta^\kappa_{i,3}) = 1$.

The case for $I_2(g^\kappa_i) = 1$ is, apart from the consideration of the colors of the nodes of π^κ_i and ϕ^κ_i which are obsolete in this case, analogous. □

Claim 4.4.19. *Assume* $c(y^\kappa_{2i+1}) = 1$ *and* $c(y^\kappa_{2i-1}) = 0$. *If* g^κ_i *is correct then* $c(\gamma^\kappa_{i,1}) = c(\gamma^\kappa_{i,2}) = 1$ *and* $c(\gamma^\kappa_{i,3}) = 0$. *If* g^κ_i *is not correct then at least one of the nodes* $u^\kappa_{i,2}$, $u^\kappa_{i,3}$ *has the same color as* g^κ_i, *at least one of the nodes* $u^\kappa_{i,6}$, $u^\kappa_{i,7}$ *has the same color as* g^κ_i *and at least one of the nodes* $u^\kappa_{i,10}$, $u^\kappa_{i,11}$ *has the same color as* g^κ_i.

Proof. Assume first that g^κ_i is correct. From Claim 4.4.17 we know that $c(\beta^\kappa_{i,1}) = col(I_1(g^\kappa_i))$. Since g^κ_i is correct, at least one of the two nodes $\beta^\kappa_{i,1}$ and g^κ_i is white.

In the following, we show that at least one of the nodes $\beta^\kappa_{i,1}$ and $u^\kappa_{i,4}$ is white. Then the bias of Type 12 implies that $c(\gamma^\kappa_{i,1}) = 1$. The case $c(\beta^\kappa_{i,1}) = 0$ is clear. Now consider the case that $c(\beta^\kappa_{i,1}) = 1$ and therefore $col(g^\kappa_i) = 0$. Then, due to Claim 4.4.17, we have $c(\alpha^\kappa_{i,1}) = 0$. Since g^κ_i is white and $u^\kappa_{i,3}$ and $u^\kappa_{i,4}$ are biased to 1 and 0, respectively, by Type 13, we have $c(u^\kappa_{i,3}) = 1$ and $c(u^\kappa_{i,4}) = 0$. Thus, $c(\gamma^\kappa_{i,1}) = 1$. Analogously, we can argue that $\gamma^\kappa_{i,2}$ is also black.

Now we show that at least one of the nodes $\beta^\kappa_{i,3}$ and $u^\kappa_{i,12}$ is black. Then the bias of Type 12 implies that $c(\gamma^\kappa_{i,3}) = 0$. By Claim 4.4.18 we know that $c(\beta^\kappa_{i,3}) = col(I_1(g^\kappa_i)) \vee col(I_2(g^\kappa_i))$. Since g^κ_i is correct, it has the opposite color of $\beta^\kappa_{i,3}$. The case $c(\beta^\kappa_{i,3}) = 1$ is clear. Now assume $c(\beta^\kappa_{i,3}) = 0$ and therefore $col(g^\kappa_i) = 1$. Then $c(u^\kappa_{i,11}) = 0$ and $c(u^\kappa_{i,12}) = 1$ since they are biased to 0 and 1, respectively, by Type 13. Thus, $c(\gamma^\kappa_{i,3}) = 0$.

Now assume that g^κ_i is not correct. If $col(I_1(g^\kappa_i)) = 1$ then $c(\alpha^\kappa_{i,1}) = 0$ and $c(\beta^\kappa_{i,1}) = 1$ due to Claim 4.4.17. Moreover, since g^κ_i is not correct, we have $col(g^\kappa_i) = 1$. Then $c(\alpha^\kappa_{i,1}) = 0$ and the biases of Type 13 imply $c(u^\kappa_{i,1}) = 0$ and $c(u^\kappa_{i,2}) = 1$. If $col(I_1(g^\kappa_i)) = 0$ then $c(\alpha^\kappa_{i,1}) = 1$ and $c(\beta^\kappa_{i,1}) = 0$ due to Claim 4.4.17. Since $\gamma^\kappa_{i,1}$ is biased to 1 by Type 12, we get $c(\gamma^\kappa_{i,1}) = 1$. Moreover, since $c(\alpha^\kappa_{i,1}) = 1$ the biases of Type 13 imply $c(u^\kappa_{i,1}) = 1$, $c(u^\kappa_{i,2}) = 0$, $c(u^\kappa_{i,4}) = 0$ and $c(u^\kappa_{i,3}) = 1$. The proof for $c(u^\kappa_{i,6})$ and $c(u^\kappa_{i,7})$ is analogous.

By Claim 4.4.18 we know that $c(\beta^\kappa_{i,3}) = col(I_1(g^\kappa_i)) \vee col(I_2(g^\kappa_i))$. Since g^κ_i is not correct, we have $col(g^\kappa_i) = c(\beta^\kappa_{i,3})$. We consider the possible cases for the color of g^κ_i and $\beta^\kappa_{i,3}$. Assume first $col(g^\kappa_i) = c(\beta^\kappa_{i,3}) = 0$. Due to Claim 4.4.17, we have $c(\alpha^\kappa_{i,1}) = c(\alpha^\kappa_{i,2}) =$

1. Then the biases of the component of Type 13 imply $c(u^\kappa_{i,9}) = 1$ and $c(u^\kappa_{i,10}) = 0$. Thus, $u^\kappa_{i,10}$ has the same color as g^κ_i. Now consider the case $col(g^\kappa_i) = c(\beta^\kappa_{i,3}) = 1$. Then $c(\gamma^\kappa_{i,3}) = 0$ since it is biased to white by Type 12. Moreover, $c(\alpha^\kappa_{i,1}) = 0$ or $c(\alpha^\kappa_{i,2}) = 0$ due to Claim 4.4.17. Then the biases of the component of Type 13 imply $c(u^\kappa_{i,9}) = 0$ and $c(u^\kappa_{i,10}) = 1$ as well as $c(u^\kappa_{i,12}) = 1$ and $c(u^\kappa_{i,11}) = 0$. Then we have $c(u^\kappa_{i,10}) \neq c(u^\kappa_{i,11})$, which proves the claim. □

Assume $c(y^\kappa_{2i+1}) = 1$ and $c(y^\kappa_{2i-1}) = 0$. Assume furthermore that g^κ_i is correct. Then, due to Claim 4.4.19, we have $c(\gamma^\kappa_{i,1}) = c(\gamma^\kappa_{i,2}) = 1$ and $c(\gamma^\kappa_{i,3}) = 0$. Thus, if the nodes y^κ_j, z^κ_j for all j are biased to their natural values by Type 9 due to $c(y^\kappa_{2i+1}) = 1$ we get $c(z^\kappa_{2i+1}) = 0$, $c(y^\kappa_{2i}) = 1$ and $c(z^\kappa_{2i}) = 0$. If, on the other hand, the nodes y^κ_j, z^κ_j for all j are biased to their unnatural values by Type 9 then due to $c(y^\kappa_{2i-1}) = 0$ we get $c(z^\kappa_{2i}) = 1$, $c(y^\kappa_{2i}) = 0$ and $c(z^\kappa_{2i+1}) = 1$.

Now assume that g^κ_i is not correct. Then Claim 4.4.19 implies that flipping g^κ_i does not decrease the cut by a weight of Type 3 corresponding to g^κ_i since at least one of the nodes $u^\kappa_{i,2}$ and $u^\kappa_{i,3}$, at least one of the nodes $u^\kappa_{i,6}$ and $u^\kappa_{i,7}$ and at least one of the nodes $u^\kappa_{i,10}$ and $u^\kappa_{i,11}$ has the opposite color of g^κ_i. Finally, Claim 4.4.17 implies $c(\alpha^\kappa_{i,j}) = \neg col(I_j(g^\kappa_i))$ for $1 \leq j \leq 2$. Thus, flipping g^κ_i to its correct color gains a weight of Type 15. □

Lemma 4.4.20. *Assume $col(d^\kappa) = 1$, $col(d^{\bar{\kappa}}) = 0$ and that all nodes y^κ_i, z^κ_i for $0 \leq i \leq 2N+1$ are biased to their natural values by Type 9. Then $c(y^\kappa_1) = 1$.*

Proof. We show that all gates of G^κ_C are correct. For the sake of contradiction, we assume that G^κ_C contains an incorrect gate and let g^κ_i be the incorrect gate with the highest index.

We first show by induction that the nodes y^κ_j, z^κ_j for $j > 2i+1$ and y^κ_{2i+1} have their natural values. Since y^κ_{2N+1} is biased to its natural value by Type 9, we have $c(y^\kappa_{2N+1}) = 1$. Assume $c(y^\kappa_{2j+1}) = 1$ for any $j > i$. If any one of the nodes $z^\kappa_{2j+1}, y^\kappa_{2j}, z^\kappa_{2j}$ has its unnatural value then Lemma 4.4.8 implies $c(y^\kappa_{2j-1}) = 0$. Then Lemma 4.4.16 implies that all nodes $z^\kappa_{2j+1}, y^\kappa_{2j}, z^\kappa_{2j}$ have their natural values whereafter Claim 4.4.10 implies $c(y^\kappa_{2i-1}) = 1$, which is a contradiction. Thus, $c(y^\kappa_{2j+1}) = 1$ implies $c(y^\kappa_{2j-1}) = 1$ for any $j > i$ and therefore it follows by induction that all nodes y^κ_j, z^κ_j for $j > 2i+1$ and y^κ_{2i+1} have their natural values.

Since g^κ_i is incorrect, all nodes y^κ_j, z^κ_j for $j \leq 2i-1$ have their unnatural values due to Lemma 4.4.9 and Lemma 4.4.8. According to Lemma 4.4.15 flipping g^κ_i does not decrease the cut by a weight of Type 3 corresponding to a node g^κ_j for which $I_k(g^\kappa_j) = g^\kappa_i$ for $1 \leq k \leq 2$. Due to Lemma 4.4.16, correcting g^κ_i does not decrease the cut by a weight of Type 3 corresponding to g^κ_i and gains a weight of Type 15. In the following, we distinguish between three cases for the index i and show that g^κ_i is unhappy in each of the cases. First, if $i > 2n + 2m$ then there are no edges of Type 5 or Type 7 incident to g^κ_i. Thus, g^κ_i is unhappy in this case. Second, if $2m + 1 \leq i \leq 2n + 2m$ then there are no edges of Type 5 incident to g^κ_i. Due to Lemma 4.4.6, correcting g^κ_i does not decrease the cut by a weight of Type 7. Third, if $i \leq 2m$ then there are no edges of Type 7 incident to g^κ_i. Correcting g^κ_i does not decrease the cut by a weight of Type 5 since due

to the biases of Type 14 we have $c(u^\kappa_{i,14}) = 0$, $c(u^\kappa_{i,13}) = 1$ for $i \leq m$ and $c(u^\kappa_{i-m,16}) = 1$, $c(u^\kappa_{i-m,15}) = 0$ for $m < i \leq 2m$. Altogether, g^κ_i is unhappy in each of the three cases, which is a contradiction. Thus, g^κ_i is correct for all i.

Therefore, all nodes y^κ_i, z^κ_i for $1 \leq i \leq 2N+1$ have their natural values.

Lemma 4.4.21. *Suppose* $c(y^\kappa_1) = c(u^\kappa) = 0$ *and* $c(u^{\overline{\kappa}}) = 1$. *Then* $col(d^\kappa) = 1$, $col(d^{\overline{\kappa}}) = 0$.

Proof. Independently of the color of $y^{\overline{\kappa}}_1$, node d^κ is biased to 1 and $d^{\overline{\kappa}}$ to 0 by Type 11. Lemma 4.4.8 implies $c(y^\kappa_0) = 0$. Since $c(u^\kappa) = 0$ and $c(y^\kappa_0) = 0$, node y^κ_0 and its counterpart, namely the constant 0, are decisive for d^κ. Therefore, $bn(d^\kappa)$ has the same color as the decisive neighbors of d^κ which implies that implies that d^κ is guided. Hence, the Substituting Lemma implies $col(d^\kappa) = 1$.

Since $c(u^{\overline{\kappa}}) = 1$, node $u^{\overline{\kappa}}$ and its counterpart, namely the constant 1, are decisive for $d^{\overline{\kappa}}$. Again, $bn(d^\kappa)$ has the same color as the decisive neighbors of d^κ which implies that $d^{\overline{\kappa}}$ is guided whereafter the Substituting Lemma implies $col(d^{\overline{\kappa}}) = 0$. □

Lemma 4.4.22. *Assume* $c(y^\kappa_1) = c(u^{\overline{\kappa}}) = 0$ *and* $c(y^{\overline{\kappa}}_1) = c(u^\kappa) = c(y^{\overline{\kappa}}_0) = 1$. *Then* $col(d^{\overline{\kappa}}) = 0$.

Proof. Lemma 4.4.8 implies $c(z^\kappa_0) = 1$ since $c(y^\kappa_1) = 0$. Node $d^{\overline{\kappa}}$ is biased to 0 by Type 11. Since $c(u^{\overline{\kappa}}) = 0$, $c(y^{\overline{\kappa}}_0) = 1$ and $c(z^\kappa_0) = 1$, node z^κ_0 and its counterpart, i.e., the constant 1, are decisive for $d^{\overline{\kappa}}$. Therefore, $bn(d^\kappa)$ has the same color as the decisive neighbors of d^κ which implies that $d^{\overline{\kappa}}$ is guided. Hence, the Substituting Lemma implies $col(d^{\overline{\kappa}}) = 0$. □

Lemma 4.4.23. *If* $c(y^\kappa_1) = 1$ *and all* y^κ_i, z^κ_i *are biased to their natural values by Type 9 then* $c(z^\kappa_1) = c(z^\kappa_0) = 0$ *and* $c(y^\kappa_0) = 1$.

Proof. Due to $c(y^\kappa_1) = 1$ and since z^κ_1 is biased to its natural value, i.e., white, we get $c(z^\kappa_1) = 0$. Analogously, we get $c(y^\kappa_0) = 1$ and $c(z^\kappa_0) = 0$. □

Lemma 4.4.24. *Assume* $col(d^\kappa) = 1$, $col(d^{\overline{\kappa}}) = 0$. *Then all nodes* y^κ_i, z^κ_i *are biased to their natural values by Type 9.*

Proof. Assume for the sake of contradiction that all nodes y^κ_i, z^κ_i are biased to their natural values by Type 9. At first, we show that all y^κ_i, z^κ_i in fact have their unnatural values. Since $col(d^{\overline{\kappa}}) = 0$, the bias by Type 9 to the unnatural value implies $c(z^\kappa_0) = 1$. Then $col(d^{\overline{\kappa}}) = 1$ together with the bias to the unnatural value imply $c(y^\kappa_0) = 0$. Then $c(z^\kappa_1) = 1$ and therefore $c(y^\kappa_1) = 0$. If $c(y^\kappa_{j-1}) = 0$ for any $2 \leq j \leq 2N+1$ then the bias to the unnatural value implies $c(z^\kappa_j) = 1$. Analogously, if $c(z^\kappa_j) = 1$ for any $2 \leq j \leq 2N+1$ then $c(y^\kappa_j) = 0$. Thus, it follows by induction that all y^κ_i, z^κ_i have their unnatural values.

Now we show that $c(\lambda^\kappa) = col(x^\kappa)$. Lemma 4.4.15 implies $c(\alpha^\kappa_{N-n+i,1}) = c(\alpha^\kappa_{N-n+i,2}) = 0$ and $c(\sigma^\kappa_{N-n+i,1}) = c(\sigma^\kappa_{N-n+i,2}) = 1$. Therefore, x^κ is weakly indifferent. Then, due to the bias of Type 8, node x^κ_i has the color of λ^κ_i for all $1 \leq i \leq n$. Thus, $c(\lambda^\kappa) = col(x^\kappa)$.

But this is a contradiction to the assumption that all nodes y_i^κ, z_i^κ are biased to their unnatural values by Type 9. Thus, all nodes y_i^κ, z_i^κ are biased to their natural values by Type 9. □

Lemma 4.4.25. *Assume $c(y_{2i-1}^\kappa) = 1$ and $c(\alpha_{i,j}^\kappa) \neq col(I_j(g_i^\kappa))$ for all $1 \leq j \leq 2$, $1 \leq i \leq 2m$. Then $c(u_{i,14}^\kappa) = col(g_i^\kappa)$ and $c(u_{i,16}^\kappa) = col(\hat{g}_i^\kappa)$ for all $1 \leq i \leq m$.*

Proof. Let $1 \leq i \leq m$ be arbitrarily chosen. The nodes g_i^κ and $\hat{g}_i^\kappa$ are correct due to Lemma 4.4.9 and Lemma 4.4.8.

Since $c(y_{2i-1}^\kappa) = 1$, node $u_{i,14}^\kappa$ is biased to $c(\alpha_{i,1}^\kappa) \wedge c(\alpha_{i,2}^\kappa)$ by Type 14. Thus, it is biased to black if and only if $\alpha_{i,1}^\kappa$ and $\alpha_{i,2}^\kappa$ are both black. Since $c(\alpha_{i,j}^\kappa) \neq col(I_j(g_i^\kappa))$ for all $1 \leq j \leq 2$ and g_i^κ is black if and only if both $I_1(g_i^\kappa)$ and $I_2(g_i^\kappa)$ are white, node g_i^κ is black if and only if $\alpha_{i,1}^\kappa$ and $\alpha_{i,2}^\kappa$ are both black. Thus, $u_{i,14}^\kappa$ is biased to the color of g_i^κ and $u_{i,13}^\kappa$ to the opposite. Then $col(g_i^\kappa) \neq c(u_{i,13}^\kappa) \neq c(u_{i,14}^\kappa)$—see Figure 4.7—and therefore $col(g_i^\kappa) = c(u_{i,14}^\kappa)$.

Now let $m+1 \leq i \leq 2m$ be arbitrarily chosen. Since $c(y_{2i-1}^\kappa) = 1$, node $u_{i-m,15}^\kappa$ is biased to $\neg c(\alpha_{i,1}^\kappa) \vee \neg c(\alpha_{i,2}^\kappa)$ by Type 14. Thus, it is biased to white if and only if $\alpha_{i,1}^\kappa$ and $\alpha_{i,2}^\kappa$ are both black. Since $c(\alpha_{i,j}^\kappa) \neq col(I_j(g_i^\kappa))$ for all $1 \leq j \leq 2$ and g_i^κ is black if and only if both $I_1(g_i^\kappa)$ and $I_2(g_i^\kappa)$ are white, node g_i^κ is black if and only if $\alpha_{i,1}^\kappa$ and $\alpha_{i,2}^\kappa$ are both black. Thus, it follows that $u_{i-m,15}^\kappa$ is biased to the opposite color of g_i^κ. Since $u_{i-m,16}^\kappa$ is biased to the opposite color of $u_{i-m,15}^\kappa$, we have $col(g_i^\kappa) \neq c(u_{i-m,15}^\kappa) \neq c(u_{i-m,16}^\kappa)$—see Figure 4.7—and therefore $col(g_i^\kappa) = c(u_{i-m,16}^\kappa)$. Since, by definition, $\hat{g}_i^\kappa = g_{i+m}^\kappa$ for all $1 \leq i \leq m$ we get $c(u_{i,16}^\kappa) = col(\hat{g}_i^\kappa)$ for all $1 \leq i \leq m$. □

Now we continue to prove the Completeness Theorem. Let P be a local optimum in G_C. From Lemma 4.4.3 we know that $c(u^0) \neq c(u^1)$. In the following, we consider the possible cases for the vector $c(y_1^0, y_1^1)$ and distinguish within them, if necessary, between the two cases for $c(u^0, u^1)$. For the cases in which at least one of the nodes y_1^0 and y_1^1 is white, we show that they cannot occur in local optima, and for the case that both nodes are black we show that the bitwise complement of the colors of the nodes g_i^0 for $N-n+1 \leq i \leq N$ induce a local optimum of C.

$c(y_1^0, y_1^1) = (0,0)$: Due to Lemma 4.4.8 we have $c(y_0^0) = c(y_0^1) = 0$ and $c(z_0^0) = c(z_0^1) = 1$. Let $c(u^\kappa, u^{\overline{\kappa}}) = (0,1)$ for $\kappa \in \{0,1\}$. Then Lemma 4.4.21 implies $col(d^\kappa) = 1$ and $col(d^{\overline{\kappa}}) = 0$. Consequently, Lemma 4.4.24 implies that the nodes y_i^κ, z_i^κ are biased to their natural values by Type 9. But then Lemma 4.4.20 implies that $c(y_1^\kappa) = 1$, which is a contradiction.

$c(y_1^\kappa, y_1^{\overline{\kappa}}) = (0,1)$ for $\kappa \in \{0,1\}$: According to Lemma 4.4.8 we have $c(y_0^\kappa) = 0$ and $c(z_0^\kappa) = 1$.

Assume first that $c(u^\kappa, u^{\overline{\kappa}}) = (0,1)$. Then Lemma 4.4.21 implies $col(d^\kappa) = 1$ and $col(d^{\overline{\kappa}}) = 0$. Then Lemma 4.4.24 implies that all nodes y_i^κ, z_i^κ are biased to their natural values by Type 9. But then Lemma 4.4.20 implies $c(y_1^\kappa) = 1$, which is a contradiction.

Now assume $c(u^\kappa, u^{\overline{\kappa}}) = (1,0)$. We first show that $c(y_0^{\overline{\kappa}}) = 1$ in this case. Assume for the sake of contradiction that $c(y_0^{\overline{\kappa}}) = 0$. If the nodes $y_i^{\overline{\kappa}}, z_i^{\overline{\kappa}}$ are biased to their natural values by Type 9 then Lemma 4.4.23 implies $c(y_0^{\overline{\kappa}}) = 1$, which is a contradiction. If they are biased to their unnatural values then $c(z_1^{\overline{\kappa}}) = 1$ since $c(y_0^{\overline{\kappa}}) = 0$—see Figure 4.6—whereafter we get $c(y_1^{\overline{\kappa}}) = 0$, which is also a contradiction. Thus, $c(y_0^{\overline{\kappa}}) = 1$. Then Lemma 4.4.22 implies $col(d^{\overline{\kappa}}) = 0$. Now we distinguish between the two possible cases for κ. If $\kappa = 0$ then Type 10 biases u^1, independently of whether $C(x^0) \geq C(x^1)$ or $C(x^0) < C(x^1)$, to 1 since $c(y_1^0, y_1^1) = (0,1)$. But then $c(u^1) = 1$ due to $col(d^1) = 0$, which is a contradiction. On the other hand, if $\kappa = 1$ then Type 10 biases u^0, also independently of whether $C(x^0) \geq C(x^1)$ or $C(x^0) < C(x^1)$, to 1 since $c(y_1^0, y_1^1) = (1,0)$. But then $c(u^0) = 1$ due to $col(d^0) = 0$, which is also a contradiction.

$c(y_1^0, y_1^1) = (1,1)$: Then Lemma 4.4.9 and Lemma 4.4.8 together imply that y_i^κ, z_i^κ have their natural values for all $\kappa \in \{0,1\}, 2 \leq i \leq 2N+1$ and all gates in G_C^0 and G_C^1 are correct. Therefore, we have $col(g_i^\kappa) \neq col(\hat{g}_i^\kappa)$ for all $1 \leq i \leq N$, $\kappa \in \{0,1\}$. Claim 4.4.17 implies $c(\alpha_{i,j}^\kappa) \neq col(I_j(g_i^\kappa))$ for all $1 \leq i \leq N$, $1 \leq j \leq 2$ and $\kappa \in \{0,1\}$.

In the following, we consider the two cases $C(x^\kappa) > C(x^{\overline{\kappa}})$ for some $\kappa \in \{0,1\}$ and $C(x^0) = C(x^1)$. We show that the first case cannot occur and that in the second case the colors of the nodes of x^0 induce a local optimum for C. Within the two cases, we argue about the color of the nodes d^κ for $\kappa \in \{0,1\}$. No node adjacent to d^κ is a comparing node—see Table 4.2—and therefore the colors of its adjacent nodes are uniquely determined. Due to the two constants of Type 5 adjacent to d^κ, node d^κ is not weakly indifferent. Thus, to show that $col(d^\kappa) = \rho$ for $\rho \in \{0,1\}$, it suffices to show that the color of the decisive neighbors of d^κ is $\overline{\rho}$ and that the component of Type 11 biases d^κ to ρ. Then $bn(d^\kappa)$ has the same color as the decisive neighbors of d^κ, which implies that d^κ is guided whereafter the Substituting Lemma (i.e., Lemma 4.3.3) implies $col(d^\kappa) = \rho$.

- Case $C(x^\kappa) > C(x^{\overline{\kappa}})$ for some $\kappa \in \{0,1\}$:
  Since all gates of G_C^κ and $G_C^{\overline{\kappa}}$ are correct, there is an index $1 \leq i \leq m$ such that $col(g_j^\kappa) = col(g_j^{\overline{\kappa}})$ for all $i < j \leq m$ and $col(g_i^\kappa) = 1$, $col(g_i^{\overline{\kappa}}) = 0$. We let $1 \leq i \leq m$ be this index. Since $col(g_j^\kappa) \neq col(\hat{g}_j^\kappa)$ and $col(g_j^{\overline{\kappa}}) \neq col(\hat{g}_j^{\overline{\kappa}})$ for all $1 \leq j \leq m$, we have $col(g_j^\kappa) \neq col(\hat{g}_j^{\overline{\kappa}})$, $col(\hat{g}_j^\kappa) \neq col(g_j^{\overline{\kappa}})$ for all $i < j \leq m$ and $col(g_i^\kappa) = col(\hat{g}_i^{\overline{\kappa}}) = 1$, $col(\hat{g}_i^\kappa) = col(g_i^{\overline{\kappa}}) = 0$. Then, since $c(y_{2j-1}^\kappa) = c(y_{2j-1}^{\overline{\kappa}}) = 1$ for all $1 \leq j \leq 2m$, Lemma 4.4.25 implies $c(u_{j,14}^\kappa) \neq c(u_{j,16}^{\overline{\kappa}})$, $c(u_{j,16}^\kappa) \neq c(u_{j,14}^{\overline{\kappa}})$ for all $i < j \leq m$ and $c(u_{i,14}^\kappa) = c(u_{i,16}^{\overline{\kappa}}) = 1$, $c(u_{i,16}^\kappa) = c(u_{i,14}^{\overline{\kappa}}) = 0$. Type 9 biases the nodes y_j^κ, z_j^κ for all j to their natural values. Thus, Lemma 4.4.23 implies $c(y_0^\kappa) = 1$ and $c(z_0^\kappa) = 0$.

  In the following, we first show $col(d^\kappa) = 0$ by naming the decisive neighbors of d^κ and showing that their color is black—for an overview of the nodes adjacent to d^κ see Table 4.2. We distinguish three cases. First, if $c(u^\kappa) = 1$ then u^κ and its counterpart, i.e., the constant 1, are decisive for d^κ. Second, if $c(u^\kappa) = 0$ and

$c(z_0^{\overline{\kappa}}) = 1$ then neither u^κ and its counterpart nor y_0^κ—which is black—and its counterpart are decisive. Instead, $z_0^{\overline{\kappa}}$ and its counterpart, i.e., the constant 1, are decisive. Third, if $c(u^\kappa) = 0$ and $c(z_0^{\overline{\kappa}}) = 0$ then the node $z_0^{\overline{\kappa}}$ and its counterpart are also not decisive—besides the nodes u^κ and y_0^κ and their corresponding counterparts. Moreover, due to $c(u_{j,14}^\kappa) \neq c(u_{j,16}^{\overline{\kappa}})$ for all $i < j \leq m$, the nodes $u_{j,14}^\kappa$ and their counterparts $u_{j,16}^{\overline{\kappa}}$ for all $i < j \leq m$ are also not decisive. But the two nodes $u_{i,14}^\kappa$ and $u_{i,16}^{\overline{\kappa}}$—which are both black—are decisive for d^κ. In all three cases the decisive neighbors of d^κ are black. Thus, $col(d^\kappa) = 0$. By Type 11 node d^κ is biased to 0. Then the bias of Type 10 implies due to $col(d^\kappa) = 0$ that $c(u^\kappa, u^{\overline{\kappa}}) = (1, 0)$.

Now we show that $col(d^{\overline{\kappa}}) = 1$. Due to $c(u^{\overline{\kappa}}) = 0$ node $c(u^{\overline{\kappa}})$ and its counterpart, i.e., the constant 1, are not decisive for $d^{\overline{\kappa}}$. We distinguish two cases. First, if $c(y_0^{\overline{\kappa}}) = 0$ then $y_0^{\overline{\kappa}}$ and its counterpart, i.e., the constant 0, are decisive. Second, if $c(y_0^{\overline{\kappa}}) = 1$ then due to $c(z_0^\kappa) = 0$ the nodes $u^{\overline{\kappa}}$, $y_0^{\overline{\kappa}}$ and z_0^κ and their corresponding counterparts are not decisive. Furthermore, due to $c(u_{j,16}^\kappa) \neq c(u_{j,14}^{\overline{\kappa}})$ for all $i < j \leq m$, the nodes $u_{j,16}^\kappa$ and their counterparts $u_{j,14}^{\overline{\kappa}}$ for all $i < j \leq m$ are also not decisive. Thus, the two nodes $u_{i,16}^\kappa$ and $u_{i,14}^{\overline{\kappa}}$—which are both white—are decisive for $d^{\overline{\kappa}}$. In both cases the decisive neighbors of $d^{\overline{\kappa}}$ are white. Thus, $col(d^{\overline{\kappa}}) = 1$. By Type 11 node $d^{\overline{\kappa}}$ is biased to 1.

Since all gates of G_C^κ are correct, we have $w(x^\kappa) = col(w_1^\kappa) = col(w_2^\kappa)$. Lemma 4.4.3 and Lemma 4.4.6 together imply that $col(w_1^\kappa) = col(w_2^\kappa) = c(\lambda^{\overline{\kappa}})$ and therefore $w(x^\kappa) = c(\lambda^{\overline{\kappa}})$. If $c(\lambda^{\overline{\kappa}}) \neq col(x^{\overline{\kappa}})$ then the nodes $y_i^{\overline{\kappa}}, z_i^{\overline{\kappa}}$ are biased to their unnatural values by Type 9. But Lemma 4.4.24 implies that they are biased to their natural values, which is a contradiction. Thus, we in fact have $c(\lambda^{\overline{\kappa}}) = col(x^{\overline{\kappa}})$ and therefore $w(x^\kappa) = col(x^{\overline{\kappa}})$, but this is a contradiction to the assumption that $C(x^\kappa) > C(x^{\overline{\kappa}})$.

- $C(x^0) = C(x^1)$:
  Since all gates of G_C^κ and $G_C^{\overline{\kappa}}$ are correct, we have $col(g_i^0) = col(g_i^1)$ for all $1 \leq i \leq m$. Since $col(g_i^0) \neq col(\hat{g}_i^0)$ and $col(g_i^1) \neq col(\hat{g}_i^1)$ for all $1 \leq i \leq m$, we have $col(g_i^0) \neq col(\hat{g}_i^1)$, $col(\hat{g}_i^0) \neq col(g_i^1)$ for all $1 \leq i \leq m$. Then, since $c(y_{2j-1}^0) = c(y_{2j-1}^1) = 1$ for all $1 \leq j \leq 2m$, Lemma 4.4.25 implies $c(u_{i,14}^0) \neq c(u_{i,16}^1)$, $c(u_{i,16}^0) \neq c(u_{i,14}^1)$ for all $1 \leq i \leq m$. Type 9 biases all nodes y_i^0, z_i^0 to their natural values. Thus, Lemma 4.4.23 implies $c(y_0^0) = 1$ and $c(z_0^0) = 0$.

  In the following, we first show $col(d^0) = 0$ by naming the decisive neighbors of d^0 and showing that their color is black—for an overview of the nodes adjacent to d^0 see Table 4.2. We distinguish three cases. First, if $c(u^0) = 1$ then u^0 and its counterpart, i.e., the constant 1, are decisive for d^0. Second, if $c(u^0) = 0$ and $c(z_0^1) = 1$ then due to $c(y_0^0) = 1$ the nodes u^0 and y_0^0 and their corresponding counterparts are not decisive. Thus, node z_0^1 and its counterpart, i.e., the constant 1, are decisive. Third, if $c(u^0) = 0$, $c(z_0^1) = 0$ then z_0^1 and its counterpart are also

not decisive—besides the nodes u^0 and y_0^0 and their corresponding counterparts. Moreover, due to $c(u_{i,14}^0) \neq c(u_{i,16}^1)$ for all $1 \leq i \leq m$, the nodes $u_{i,14}^0$ and their counterparts $u_{i,16}^1$ for all $1 \leq i \leq m$ are also not decisive. Then the neighbors of Type 5 representing the constant 1 adjacent to d^0 via edges of relative weight 1 are decisive for d^0. In each of the above cases the decisive neighbors of d^0 are black. Thus, $col(d^0) = 0$. By Type 11 node d^0 is biased to 0. Then, due to $col(d^0) = 0$, the bias of Type 10 implies $c(u^0, u^1) = (1, 0)$.

Now we show that $col(d^1) = 1$. Since $c(u^1) = 0$, node u^1 and its counterpart, i.e., the constant 1, are not decisive. We distinguish two cases. First, if $c(y_0^1) = 0$ then y_0^1 and its counterpart, i.e., the constant 0, are decisive. Second, if $c(y_0^1) = 1$ then due to $c(z_0^0) = 0$ the node z_0^0 and its counterpart, i.e., the constant 1, are also not decisive—besides u^1 and y_0^1 and their corresponding counterparts. Furthermore, due to $c(u_{i,16}^0) \neq c(u_{i,14}^1)$ for all $1 \leq j \leq m$, the nodes $u_{i,16}^0$ and their counterparts $u_{i,14}^1$ for all $1 \leq j \leq m$ are also not decisive. Thus, the neighbors of Type 5 representing the constant 0 adjacent to d^1 via edges of relative weight 1 are decisive for d^1. In both cases the decisive neighbors of d^1 are white. Thus, $col(d^1) = 1$. By Type 11 node d^1 is biased to 1.

Since all gates of G_C^0 are correct, we have $w(x^0) = col(w_1^0) = col(w_2^0)$. Then, Lemma 4.4.3 and Lemma 4.4.6 together imply that $col(w_1^0) = col(w_2^0) = c(\lambda^1)$ and therefore $w(x^0) = c(\lambda^1)$. If $c(\lambda^1) \neq col(x^1)$ then the nodes y_i^1, z_i^1 are biased to their unnatural values by Type 9. But Lemma 4.4.24 implies that they are biased to their natural values, which is a contradiction. Thus, we in fact have $c(\lambda^1) = col(x^1)$ and therefore $w(x^0) = col(w_1^0) = col(w_2^0) = col(x^1)$. Due to our assumption that C returns its input as better neighbor if and only if the input is a local optimum, the colors of x^0 induce a local optimum of C. □

Chapter 5

Impact of the Results on Other Problems

In this section, we discuss three problems on which our results have direct impact due to PLS-reductions from the literature that are based on LocalMax-Cut. For this, we make use of the following result essentially equivalent to Lemma 3.3 of Schäffer and Yannakakis [54]:

Theorem 5.0.1 ([54]). *Let* Π *and* Π' *be problems in* PLS *and let* Φ, Ψ *define a tight* PLS*-reduction from* Π *to* Π'. *Then the following properties are satisfied:*

i) If P has the all-exp property then Π' *has the all-exp property.*

ii) If the StandardAlgorithmProblem *is* PSPACE*-complete for* Π *then it is also* PSPACE*-complete for* Π'.

The direct impact of our results is always due to the following two properties of the PLS-reductions:

- They are tight.
- They preserve the degree of the LocalMax-Cut instance in some sense.

5.1 Max-2SAT with FLIP-neighborhood

An instance of LocalMax-2SAT is a Boolean SAT-formula in conjunctive normal form with weighted clauses containing at most two literals. A solution is an assignment of truth values to the variables and the objective is to maximize the sum of the weights of the satisfied clauses. The neighborhood of a solution contains all solutions in which the value of exactly one variable is switched.

Schäffer and Yannakakis [54] show that LocalMax-2SAT is PLS-complete by reducing from LocalMax-Cut. They introduce for each node $u \in V$ of a given instance $G = (V, E)$ of LocalMax-Cut a variable $\tilde{u}$ and for each edge $\{u, v\} \in E$ for $v \in V$ two clauses $(\tilde{u} \vee \tilde{v})$ and $(\overline{\tilde{u}} \vee \overline{\tilde{v}})$, where the two clauses have the same weight as the edge $\{u, v\}$. Then they show that a local optimum in the resulting Boolean formula corresponds to a local optimum in the LocalMax-Cut instance. Since their reduction is tight and each variable occurs twice as often in the resulting formula as the corresponding node in the LocalMax-Cut instance has incident edges, we obtain the following results from the All-Exp Theorem (i.e., Theorem 3.6.1), the SAPPSC Theorem (i.e., Theorem 3.7.1) and the Completeness Theorem (i.e., Theorem 4.4.2).

Theorem 5.1.1. *For the* LocalMax-2SAT(i) *problem, arising from* LocalMax-2SAT *by restricting the inputs such that each variable occurs in at most $i \in \mathbb{N}$ clauses, the following complexity results hold:* LocalMax-2SAT(8) *has the all-exp property, its corresponding* SAP *is* PSPACE*-complete, and* LocalMax-2SAT(10) *is* PLS*-complete.*

5.2 Congestion Games

A congestion game [52] is a tuple $(N, E, (S_i)_{i\in N}, (d_e)_{e\in E})$, where $N = \{1, \ldots, n\}$ is the set of players, $E = \{1, \ldots, m\}$ is the set of resources, $S_i \subseteq 2^E$ is the set of strategies of player i and $d_e : \mathbb{N} \to \mathbb{Z}$ is the delay function of resource e. Let $s := (s_1, \ldots, s_n)$ with $s_i \in S_i$ be a state and let $f_s(e) := |\{i : e \in s_i\}|$ be the congestion of resource e in s. The private cost of a player i in state s is defined by $c_i(s) := \sum_{e\in s_i} d_e(f_s(e))$. The problem is to find a pure Nash equilibrium, i.e., a state in which the private cost of each player does not decrease if the player unilaterally deviates from its strategy.

Fabrikant et al. [20] showed PLS-complexity for the problem of finding a Nash equilibrium in congestion games via reduction from a problem called PosNAE3SAT [54], which is essentially very similar to LocalMax-Cut—in fact, Schäffer and Yannakakis [54] showed that they can easily be reduced to each other. Ackermann et al. [3] introduced a subclass of congestion games called *threshold games* as a vehicle to prove PLS-completeness for the computation of Nash equilibria in congestion games. In threshold games, each player i has exactly two strategies. One strategy contains a single resource e_i which is not an element of any other strategy (including the strategies of the other players). The other strategy is a subset of the set of resources E not containing e_i for any player i. Moreover, in threshold games no resource is an element of more than two strategies. The authors show that the computation of Nash equilibria in threshold games is PLS-complete via reduction from LocalMax-Cut. In their reduction they construct a threshold game Γ in which every node v of the LocalMax-Cut instance G corresponds to a player i in Γ such that no strategy of S_i consists of more resources than there are edges incident to v in G. Due to the tightness of their reduction the All-Exp Theorem, the SAPPSC Theorem and the Completeness Theorem cause the following result.

Theorem 5.2.1. *For the problem* CongNash(i) *of computing a Nash equilibrium in congestion games in which every strategy contains at most $i \in \mathbb{N}$ resources, the following complexity results hold:* CongNash(4) *has the all-exp property, its corresponding* SAP *is* PSPACE*-complete, and* CongNash(5) *is* PLS*-complete.*

5.3 Partitioning with SWAP-neighborhood

An instance for the problem Partitioning [30] is a graph $G = (V, E)$ with weighted edges and maximum degree $i \in \mathbb{N}$ and an even number of vertices. A feasible solution is a partition of V into two sets V_1, V_2 of equal size. In the neighborhood of a solution s are all solutions that can be obtained from s by exchanging one node in V_1 by one node in

V_2. The objective is the weight of the cut and the goal is to minimize or to maximize the objective (Johnson et al. [30] note that the two alternatives are equivalent).

Schäffer and Yannakakis [54] prove the PLS-completeness of PARTITIONING by means of a reduction from LOCALMAX-CUT. From an instance G of LOCALMAX-CUT they construct a graph G' for which $deg(G') = deg(G) + 1$. Since their reduction is tight, due to the All-Exp Theorem, the SAPPSC Theorem and the Completeness Theorem we get:

Theorem 5.3.1. *For the problem* PARTITIONING(i) *arising from* PARTITIONING *by restricting the input graphs to maximum degree* $i \in \mathbb{N}$*, we have the following properties:* PARTITIONING(5) *has the all-exp property, its corresponding* SAP *is* PSPACE*-complete and* PARTITIONING(6) *is* PLS*-complete.*

Chapter 6

Conclusion and Open Problems

It was known that LocalMax-Cut is hard in general. It was also known that it becomes easy if the input is restricted to cubic graphs. However, the border lines of its complexity were unknown. In this thesis, we have shown that LocalMax-Cut already becomes hard for graphs with very small degree.

For graphs with nodes of maximum degree four with what we call Types I and III (for a formal definition, see Definition 3.2.1), we have shown that LocalMax-Cut is already P-hard. It would be interesting to know whether a local optimum can be computed in polynomial time for such graphs. Then the problem would become P-complete.

In contrast to cubic graphs, where the local search approach always leads to a local optimum in a quadratic number of steps, we could show that LocalMax-Cut has the is-exp property for graphs with nodes of Type I and III. However, our instances and initial solutions allow very short sequences of improving steps that lead to a local optimum. It remains open whether LocalMax-Cut has the all-exp property for graphs with nodes of Type I and III.

The enforcing technique that we have developed in this thesis extends graphs and initial solutions by further nodes and edges according to some given generalized pivot rule. For the resulting graph and initial partition, we get for *every* sequence s of improving steps starting at the initial partition the following property. If one deletes from s the steps of the nodes that are added by the extension, one obtains the sequence induced by the pivot rule in the original graph. Our technique turned out to be powerful enough to easily deduce the all-exp property and the PSPACE-completeness of the StandardAlgorithmProblem (SAP) for LocalMax-Cut on graphs with maximum degree four: Having the enforcing technique available, we could achieve these two results, in essence, by merely designing a generalized pivot rule that is polynomial-time computable and induces the desired behavior. In this respect, the enforcing technique has proven to be a very helpful tool for showing complexity results, in particular, as in our case, to construct worst case instances or reductions. Since it was designed by means of nodes of degree four and since there are only quadratically many improving steps possible for cubic graphs, our technique and the complexity results derived from it may be helpful to shed light on border lines of hardness results in other problems.

For graphs with maximum degree five, we have shown PLS-completeness for LocalMax-Cut. This result restricts the possibility for the minimum degree for which LocalMax-Cut is PLS-complete to either four or five (unless PLS $\subseteq$ P). Thus, the naturally remaining questions concern the complexity of LocalMax-Cut on graphs with maximum degree

four. Is it in P? Is it PLS-complete? Is it neither of the two?

Via existing tight PLS-reductions in the literature we have directly transferred our results to other problems and strengthened the previously known borders of hardness in these problems. One of the most important local search problems is the TRAVELLINGSALESMANPROBLEM (TSP) with k-opt neighborhood. Via a PLS-reduction from LOCALMAX-CUT to the TSP that transfers the degree of the instance of LOCALMAX-CUT to the size of the neighborhood of the TSP, one might get closer to borders of complexity properties. In particular, one could get closer to the minimum $d \in \mathbb{N}$ for which TSP with d-opt neighborhood is PLS-complete, the minimum for which it has the all-exp property, and the minimum for which its corresponding SAP is PSPACE-complete.

Bibliography

Note: Each entry is followed by a list of the pages from which there was a reference to that entry.

[1] E. Aarts and J. Lenstra. *Local Search in Combinatorial Optimization*. Princeton University Press, 2003. → 2

[2] E. H. L. Aarts, J. Korst, and W. Michiels. *Theoretical Aspects of Local Search*. Springer, 2007. → 2, 9

[3] H. Ackermann, H. Röglin, and B. Vöcking. On the impact of combinatorial structure on congestion games. *J. ACM*, 55(6):1–22, 2008. DOI: 10.1145/1455248.1455249. → 5, 11, 120

[4] I. Adler, R. M. Karp, and R. Shamir. A simplex variant solving an $m \times d$ linear program in $O(min(m^2, d^2))$ expected number of pivot steps. *Journal of Complexity*, 3(4):372–387, 1987. DOI: 10.1016/0885-064X(87)90007-0. → 9

[5] E. Angel. A survey of approximation results for local search algorithms. In *Efficient Approximation and Online Algorithms*, volume 3484 of *LNCS*, pages 30–73. Springer, 2006. DOI: 10.1007/11671541_2. → 2

[6] S. Arora and B. Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009. → 16

[7] D. Arthur, B. Manthey, and H. Röglin. Smoothed analysis of the k-means method. *Journal of the ACM*, 58(5), 2011. DOI: 10.1145/2027216.2027217. → 10

[8] F. Barahona, M. Grötschel, M. Jünger, and G. Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36(3):493–513, 1988. URL: http://www.jstor.org/stable/170992. → 5

[9] R. E. Bixby. Solving real-world linear programs: A decade and more of progress. *Operations Research*, 50(1):3–15, 2002. URL: http://www.jstor.org/stable/3088443. → 2

[10] K. H. Borgwardt. *Mathematical Developments Arising from Linear Programming*, volume 114 of *Contemporary Mathematics*, chapter Probabilistic Analysis of Simplex Algorithms, pages 21–34. American Mathematical Society, 1990. → 9

[11] B. Chandra, H. J. Karloff, and C. A. Tovey. New results on the old k-opt algorithm for the traveling salesman problem. *SIAM Journal on Computing Computing*, 28(6): 1998–2029, 1999. DOI: 10.1137/S0097539793251244. → 3, 9

[12] P. Crescenzi, R. Silvestri, and L. Trevisan. On weighted vs unweighted versions of combinatorial optimization problems. *Information and Computation*, 167(1): 10–26, 2001. DOI: 10.1006/inco.2000.3011. → 9

[13] G. Dantzig. Programming in linear structure. Technical report, U.S. Air Force, Washington, D.C., 1948. → 2

[14] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, Inc., 2000. → 2, 4

[15] D. Dumrauf. *On the Hardness of Computing Local Optima*. PhD thesis, University of Paderborn, 2011. → 10

[16] D. Dumrauf and B. Monien. On the pls-complexity of maximum constraint assignment. submitted, 2008. URL: http://homepages.uni-paderborn.de/dumrauf/MCA.pdf. → 10

[17] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264, 1972. DOI: 10.1145/321694.321699. → 9

[18] R. Elsässer and T. Tscheuschner. Settling the complexity of local max-cut (almost) completely. In *Proceedings of the 38th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 6755 of *LNCS*, pages 171–182. Springer, 2011. DOI: 10.1007/978-3-642-22006-7_15. → 8, 10

[19] M. Englert, H. Röglin, and B. Vöcking. Worst case and probabilistic analysis of the 2-opt algorithm for the TSP. *Electronic Colloquium on Computational Complexity (ECCC)*, 13(092), 2006. URL: http://doi.acm.org/10.1145/1283383.1283522. → 3, 10

[20] A. Fabrikant, C. H. Papadimitriou, and K. Talwar. The complexity of pure nash equilibria. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing (STOC)*, pages 604–612, 2004. DOI: 10.1145/1007352.1007445. → 11, 120

[21] P. Floreen and P. Orponen. Complexity issues in discrete hopfield networks. Technical Report NC-TR-94-009, Neuro-COLT, October 1994. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.53.2598. → 5

[22] O. Friedmann, T. Hansen, and U. Zwick. Subexponential lower bounds for randomized pivoting rules for the simplex algorithm. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, pages 283–292. ACM, 2011. DOI: 10.1145/1993636.1993675. → 4

[23] M. Garey and D. Johnson. Some simplifiednp-completegraphproblems. *Theoretical Computer Science*, 1(3):237–267, 1976. DOI: 10.1016/0304-3975(76)90059-1. → 9

[24] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. Mathematical Sciences Series. W. H. Freeman & Co., New York, 1990. → 85

[25] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995. DOI: 10.1145/227683.227684. → 9

[26] A. Haken. Connectionist networks that need exponential time to stabilize. Technical report, University of Toronto, Canada, 1989. → 5

[27] A. Haken and M. Luby. Steepest descent can take exponential time for symmetric connection networks. *Complex Systems*, 2(2):191–196, 1988. URL: https://www.complex-systems.com/pdf/02-2-3.pdf. → 5, 8, 25

[28] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys (CSUR)*, 31(3):264–323, 1999. DOI: 10.1145/331499.331504. → 2

[29] D. S. Johnson and L. A. McGeoch. The Traveling Salesman Problem: A Case Study. In E. H. L. Aarts and J. K. Lenstra, editors, *Local Search in Combinatorial Optimization*, pages 215–310. Wiley and Sons, New York, 1997. → 3

[30] D. S. Johnson, C. H. Papadimtriou, and M. Yannakakis. How easy is local search? *Journal of Computer and System Science*, 37(1):79–100, 1988. DOI: 10.1016/0022-0000(88)90046-3. → 4, 9, 10, 13, 14, 98, 120, 121

[31] G. Kalai. A subexponential randomized simplex algorithm. In *Proceedings of the Twenty Fourth Annual ACM Symposium on Theory of Computing (STOC)*, pages 475–482. ACM, 1992. DOI: 10.1145/129712.129759. → 3, 4

[32] G. Kalai and D. Kleitman. A quasi-polynomial bound for the diameter of graphs of polyhedra. *American Mathematical Society*, 26(2): 315–316, 1992. URL: http://www.ams.org/journals/bull/1992-26-02/S0273-0979-1992-00285-9/S0273-0979-1992-00285-9.pdf. → 3

[33] N. Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing (STOC)*, volume 4, pages 302–311. ACM, 1984. DOI: 10.1145/800057.808695. → 3

[34] R. M. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972. DOI: 10.1007/978-3-540-68279-0_8. → 5

[35] L. Khachiyan. A polynomial algorithm in linear programming. *Doklady Akademia Nauk SSSR*, pages 1093–1096, 1979. → 3

[36] S. Khot, G. Kindler, E. Mossel, and R. O'Donnell. Optimal inapproximability results for max-cut and other two-variable csps. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*, pages 146–154, 2004. DOI: 10.1109/FOCS.2004.49. → 9

[37] V. Klee and P. Kleinschmidt. The d-step conjecture and its relatives. *Mathematics of Operations Research*, 12(4):718–755, 1987. URL: http://www.jstor.org/stable/10.2307/3689926. → 3

[38] V. Klee and G. Minty. How good is the simplex algorithm? *Inequalities*, 3:159–175, 1972. → 2, 3

[39] M. W. Krentel. Structure in locally optimal solutions. In *Proceedings of the 30th Annual Syposium on Foundations of Computer Science (FOCS)*, pages 216–221. IEEE, 1989. DOI: 10.1109/SFCS.1989.63481. → 8, 10, 98

[40] M. W. Krentel. On finding and verifying locally optimal solutions. *SIAM Journal on Computing*, 19(4):742–749, 1990. DOI: 10.1137/0219052. → 10

[41] R. Ladner. The circuit value problem is log space complete for P. *SIGACT News*, 7 (1):18–20, 1975. DOI: 10.1145/990518.990519. → 22

[42] M. Loebl. Efficient maximal cubic graph cuts. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, volume 510 of *LNCS*, pages 351–362. Springer, 1991. DOI: 10.1007/3-540-54233-7_147. → 5, 9

[43] G. S. Lueker. Unpublished manuscript. Princeton University, Princeton, NJ, 1975. → 3

[44] J. Matousek, M. Sharir, and E. Welzl. A subexponential bound for linear programming. *Algorithmica*, 16(4/5):498–516, October / November 1996. DOI: 10.1007/BF01940877. → 3

[45] D. Monderer and L. S. Shapley. Potential games. *Games and Economic Behavior*, 14 (1):124–143, 1995. DOI: 10.1006/game.1996.0044. → 11

[46] B. Monien and T. Tscheuschner. On the power of nodes of degree four in the local max-cut problem. In *Proceedings of the 7th International Conference on Algorithms and Complexity, Rome, Italy*, volume 6078 of *LNCS*, pages 264–275. Springer, 2010. DOI: 10.1007/978-3-642-13073-1_24. → 8

[47] B. Monien, D. Dumrauf, and T. Tscheuschner. Local search: simple, successful, but sometimes sluggish. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP)*, number 6198 in LNCS, pages 1–17. Springer, 2010. DOI: 10.1007/978-3-642-14165-2_1. → 2, 8

[48] J. B. Orlin, A. P. Punnen, and A. S. Schulz. Approximate local search in combinatorial optimization. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pages 587–596. SIAM, 2004. → 9

[49] S. Poljak. Integer linear programs and local search for max-cut. *SIAM Journal on Computing*, 24(4):822–839, 1995. DOI: 10.1137/S0097539793245350. → 5, 7, 9, 27

[50] S. Poljak and Z. Tuza. Maximum cuts and largest bipartite subgraphs. In *Combinatorial Optimization*, pages 181–244. American Mathematical Society, Providence, RI, 1995. → 5

[51] G. Reinelt. TSPLIB - a traveling salesman problem library. *INFORMS Journal on Computing*, 3(4):376–384, 1991. DOI: 10.1287/ijoc.3.4.376. → 3

[52] R. W. Rosenthal. A class of games possessing pure-strategy nash equilibria. *International Journal of Game Theory*, 2:65–67, 1973. DOI: 10.1007/BF01737559. → 10, 120

[53] C. H. Roth Jr. *Fundamentals of logic design*. Brooks/Cole Pub Co, 2009. → 17

[54] A. A. Schäffer and M. Yannakakis. Simple local search problems that are hard to solve. *SIAM Journal on Computing*, 20(1):56–87, 1991. DOI: 10.1137/0220004. → 4, 5, 7, 8, 10, 14, 93, 98, 101, 107, 109, 110, 111, 119, 120, 121

[55] S. Smale. On the average number of steps in the simplex method of linear programming. *Mathematical Programming*, 27:241–262, 1983. DOI: 10.1007/BF02591902. → 9

[56] R. Solis-Oba. Local search. In T. F. Gonzalez, editor, *Handbook of Approximation Algorithms and Metaheuristics*. Chapman & Hall/CRC, 2007. → 9

[57] D. A. Spielman and S.-H. Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM*, 51(3):385–463, 2004. DOI: 10.1145/990308.990310. → 9

[58] M. Todd. The many facets of linear programming. *Mathematical Programming*, 91 (3):417–436, 2001. DOI: 10.1007/s101070100261. → 2, 3

[59] A. Vattani. k-means requires exponentially many iterations even in the plane. In *Proceedings of the 25th ACM Symposium on Computational Geometry (SCG)*, pages 324–332, 2009. DOI: 10.1145/1542362.1542419. → 4

[60] M. Yannakakis. Node-and edge-deletion np-complete problems. In *Proceedings of the tenth annual ACM symposium on Theory of computing (STOC)*, pages 253–264. ACM, 1978. DOI: 10.1145/800133.804355. → 9

[61] M. Yannakakis. The analysis of local search problems and their heuristics. In *Proceedings of the 7th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 415 of *LNCS*, pages 298–311. Springer, 1990. DOI: 10.1007/3-540-52282-4_52. → 9

[62] M. Yannakakis. Equilibria, fixed points, and complexity classes. *Computer Science Review*, 3(2):71–85, 2009. DOI: 10.1016/j.cosrev.2009.03.004. → 9