



ELSEVIER

Available online at [www.sciencedirect.com](http://www.sciencedirect.com)

SCIENCE @ DIRECT®

Operations Research Letters 31 (2003) 28–34

**Operations  
Research  
Letters**

[www.elsevier.com/locate/dsw](http://www.elsevier.com/locate/dsw)

# On local search for the generalized graph coloring problem

Tjark Vredeveld\*, Jan Karel Lenstra

*Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, P.O.Box 513,  
5600 MB Eindhoven, Netherlands*

Received 26 February 2002; accepted 6 May 2002

## Abstract

Given an edge-weighted graph and an integer  $k$ , the generalized graph coloring problem is the problem of partitioning the vertex set into  $k$  subsets so as to minimize the total weight of the edges that are included in a single subset. We recall a result on the equivalence between Karush–Kuhn–Tucker points for a quadratic programming formulation and local optima for the simple flip-neighborhood. We also show that the quality of local optima with respect to a large class of neighborhoods may be arbitrarily bad and that some local optima may be hard to find. © 2002 Elsevier Science B.V. All rights reserved.

*Keywords:* Graph coloring; Local search; Worst-case analysis

## 1. Introduction

Consider the following problem. Given a graph  $G = (V, E)$ , a weight function  $w : E \rightarrow \mathbb{Z}$  on its edges, and an integer  $k \geq 2$ , find a color assignment  $c : V \rightarrow \{1, \dots, k\}$  of the vertices that minimizes the total weight of the *monochromatic edges*, i.e., edges that have end points with the same color. The problem was first stated by Carlson and Nemhauser [1], who write about ‘scheduling to minimize interaction cost’. The problem may occur when one wishes to partition a set of items into a given number of groups so as to minimize the total pairwise interaction cost. The problem is also referred to as the generalized graph coloring problem (GGCP) [8], graph  $k$ -partitioning [5], and  $k$ -min cluster [10]. For  $k = 2$  the problem is equivalent to the well-known max cut problem, as for two colors minimizing the total weight of

the monochromatic edges is equivalent to maximizing the weight of the cut edges. For general  $k$ , the problem is equivalent to the max  $k$ -cut problem. As max cut is NP-hard [6], the GGCP is also NP-hard, even for fixed  $k$ . If  $k$  is not part of the input, we denote the problem by  $k$ -GGCP.

From an approximation point of view, the GGCP is not equivalent to the max  $k$ -cut problem. Under the assumption that  $P \neq NP$ , Kann et al. [5] show that for  $k > 2$  and every  $\varepsilon > 0$  there exists a constant  $\alpha > 0$  such that the GGCP cannot be approximated in polynomial time within a factor  $\alpha|V|^{2-\varepsilon}$  of optimal. For the 2-GGCP, Garg et al. [2] gave a polynomial-time  $\mathcal{O}(\log n)$ -approximation algorithm. On the negative side, as a direct consequence of the hardness of approximating max cut by Håstad [3], there cannot exist a polynomial-time algorithm for 2-GGCP such that the solution is guaranteed to have value within  $\frac{18}{17}$  times the optimal solution value, unless  $P = NP$ .

A way to find approximate solutions is through *local search*, which iteratively searches through the set of feasible solutions. Starting from an initial solution, a

\* Corresponding author.

*E-mail addresses:* [tjark@dis.uniroma1.it](mailto:tjark@dis.uniroma1.it) (T. Vredeveld),  
[jkl@win.tue.nl](mailto:jkl@win.tue.nl) (J.K. Lenstra).

local search procedure moves from a feasible solution to a *neighboring* solution until some stopping criteria are met. The choice of a suitable neighborhood function has an important influence on the performance of local search. The simplest form of local search is *iterative improvement*. This method iteratively chooses a better solution in the neighborhood of the current solution; it stops when no better solution is found. We say that the final solution is a *local optimum*.

In this note, we recall a result that is implicit in [1] on the relation between the Karush–Kuhn–Tucker conditions and so-called FLIP-optimal solutions. Moreover, we show that the quality of local optima may be bad, and we mention some results on the time required to find locally optimal solutions.

## 2. Neighborhoods

In this section, we describe the neighborhood FLIP, its extension *m*-FLIP, and a variable-depth search variant of FLIP, called VD-FLIP.

Given a solution, a FLIP neighbor is obtained by choosing a single vertex and assigning it a different color. A solution is *FLIP-optimal* if flipping any single vertex does not decrease the total weight of monochromatic edges.

To obtain an *m*-FLIP neighbor of a given solution, we choose at most *m* vertices and flip them. A solution is *m*-FLIP-optimal if it has no *m*-FLIP neighbor of smaller objective value.

The third neighborhood, VD-FLIP, is a form of variable-depth search, introduced by Kernighan and Lin [7] for the graph partitioning problem. We obtain a neighbor of a given solution as follows. To facilitate the exposition, we define tie-breaking rules, based on complete orderings of the vertices and the colors. We start by labeling all vertices ‘unflipped’. We iteratively choose the first unflipped vertex that is best to flip, assign it the first best new color, and label it ‘flipped’. After  $|V|$  iterations all vertices have been flipped and we have obtained a series of  $|V|$  solutions, of which we choose the first best one as our neighbor. Note that if there are only two colors, the last solution in the series is equivalent to the first solution. We say that a solution is *VD-FLIP-optimal* if its VD-FLIP neighbor does not have a smaller objective value.

## 3. KKT conditions and FLIP-optimality

Carlson and Nemhauser [11] gave a quadratic programming formulation for the GGCP. It uses binary variables  $x_{hi}$  for  $i \in V$ ,  $h = 1, \dots, k$ :  $x_{hi} = 1$  if and only if vertex  $i$  is colored with color  $h$ . The weight function is extended to the complete graph on  $V$  by setting  $w_{ij} = 0$  whenever  $\{i, j\}$  is not an edge in  $E$ . The quadratic formulation is then the following:

$$(QP) \quad \min \frac{1}{2} \sum_h \sum_{i,j} w_{ij} x_{hi} x_{hj} \tag{1}$$

$$\text{s.t.} \quad \sum_h x_{hi} = 1, \quad i \in V,$$

$$x_{hi} \in \{0, 1\}, \quad i \in V, \quad h = 1, \dots, k. \tag{2}$$

As there is a one-to-one correspondence between feasible solutions to (QP) and a color assignment of the vertices, we can denote a feasible color assignment  $c$  by its corresponding feasible solution  $x \in \{0, 1\}^{k|V|}$  to (QP).

Let us replace the integrality constraint (2) by  $x_{hi} \geq 0$ . Carlson and Nemhauser showed that there exists an optimal solution to this program that is integral. The Karush–Kuhn–Tucker (KKT) conditions for this quadratic program are

$$\sum_j w_{ij} x_{hj} - \lambda_i = \mu_{hi}, \quad i \in V, \quad h = 1, \dots, k, \tag{3}$$

$$\sum_h x_{hi} = 1, \quad i \in V,$$

$$\mu_{hi} \geq 0, \quad i \in V, \quad h = 1, \dots, k, \tag{4}$$

$$x_{hi} \geq 0, \quad i \in V, \quad h = 1, \dots, k, \tag{5}$$

$$\mu_{hi} x_{hi} = 0, \quad i \in V, \quad h = 1, \dots, k. \tag{6}$$

The following result is implicit in Carlson and Nemhauser [1], and also mentioned by Lenstra [9].

**Theorem 1.** *An integral solution satisfies the KKT conditions if and only if it is FLIP-optimal.*

**Proof.** Suppose  $x \in \{0, 1\}^{k|V|}$  satisfies the KKT conditions, for some  $\lambda \in \mathbb{R}^{|V|}$  and  $\mu \in \mathbb{R}_+^{k|V|}$ , and let

$x'$  be the solution obtained by flipping a vertex, say vertex  $j$ . Assume w.l.o.g. that this vertex has color  $g$  in  $x$  and color  $g'$  in  $x'$ . The change in costs due to this flip is

$$\begin{aligned} & \frac{1}{2} \sum_h \sum_{p,q} w_{pq} x'_h p x'_{hq} - \frac{1}{2} \sum_h \sum_{p,q} w_{pq} x_h p x_{hq} \\ &= \sum_p w_{pj} x_{g'p} - \sum_p w_{pj} x_{gp} \\ &\stackrel{(3)}{=} (\lambda_j + \mu_{g'j}) - (\lambda_j + \mu_{gj}) \\ &\stackrel{(6)}{=} \mu_{g'j} \geq 0. \end{aligned}$$

Thus, this new solution is not better than  $x$  and hence  $x$  is FLIP-optimal.

Now, consider a FLIP-optimal solution  $x \in \{0, 1\}^{k|V|}$ . Let  $\lambda_i$  be the total weight of monochromatic edges incident to vertex  $i \in V$ , i.e., if  $x_{gi} = 1$ , then

$$\lambda_i = \sum_j w_{ij} x_{gj}, \quad i \in V.$$

Let  $\mu_{hi}$  denote the change in the weight of monochromatic edges incident to vertex  $i \in V$  when we change its color to  $h$ , i.e.,

$$\mu_{hi} = \sum_j w_{ij} x_{hj} - \lambda_i, \quad h = 1, \dots, k, \quad i \in V.$$

As  $x$  is FLIP-optimal,  $\mu_{hi} \geq 0$  and  $\mu_{gi} = 0$  if  $x_{gi} = 1$ , and thus  $(x, \lambda, \mu)$  satisfies (3), (4), and (6). As  $x$  is a feasible solution, it certainly satisfies (1) and by definition of  $\lambda$  and  $\mu$ ,  $(x, \lambda, \mu)$  satisfies (3). Hence,  $(x, \lambda, \mu)$  is a KKT-point.  $\square$

In their paper, Carlson and Nemhauser propose an iterative improvement procedure that always moves to the best FLIP neighbor, i.e., the one yielding the highest decrease in the objective value; one may escape from local optima by making a zero-cost FLIP. They report that this method is efficient and frequently attains global minima. For an instance with 45 vertices, Kolen and Lenstra [8] report that iterative improvement over the FLIP neighborhood always finds the same local minimum in the case of two colors, while for three and four colors several local minima are being found. In

the subsequent sections, we prove worst-case results on the quality of local optima and the running time of iterative improvement.

#### 4. Local optima may be bad

We will now show that a large class of local optima can be arbitrarily bad. The underlying neighborhood functions for this class are so-called *polynomially searchable neighborhoods*, which are neighborhoods for which in polynomial time either a better neighbor will be found if one exists or it is determined that the current solution is locally optimal.

The proofs in this section are based on graphs with optimum GGCP value equal to 0. Our arguments can be trivially extended to graphs with a strictly positive optimum.

**Theorem 2.** *Consider the GGCP with  $k \geq 3$ . For any constant  $\rho > 1$ , a local optimum w.r.t. a polynomially searchable neighborhood is not guaranteed to have value at most  $\rho$  times the optimum, unless  $P = NP$ .*

**Proof.** Consider a graph  $G = (V, E)$  with unit weights on the edges. For  $k \geq 3$ , the problem of deciding whether  $G$  is  $k$ -colorable, i.e., it can be colored with  $k$  colors without monochromatic edges, is NP-complete [6]. If  $G$  is  $k$ -colorable, then the optimal value for GGCP is 0 and otherwise it will be at least 1. If there exists a constant  $\rho > 1$  such that a local optimum is guaranteed to have value at most  $\rho$  times the optimal value, then any locally optimal solution for a  $k$ -colorable graph has value 0 and it would be a global optimum, whereas a local optimum for a graph that cannot be properly colored with  $k$  colors has value at least 1. Hence, any procedure that finds a local optimum decides on the  $k$ -colorability of  $G$ . An arbitrary coloring has value at most  $|E|$ , because of the unit weights. Thus, an iterative improvement procedure needs at most  $|E|$  iterations to find a local optimum. As the time spent to find each neighbor is by assumption polynomially bounded in the input size, the total time spent by iterative improvement is polynomially bounded.  $\square$

For the three neighborhoods defined in Section 2, we show stronger results, as these results hold without the assumption  $P \neq NP$  and are also true for  $k = 2$ .

**Theorem 3.** For any constant  $\rho > 1$ , there exists a class of instances and a FLIP-optimal solution for each instance, such that the value of this FLIP-optimal solution is larger than  $\rho$  times the optimal solution value.

**Proof.** Consider the graph  $G = (V, E)$  with  $V = \{1, 2, 3, 4\}$  and  $E = \{\{1, 2\}, \{2, 3\}, \{3, 4\}\}$  and let the weights on the edges be 1. As this graph is bipartite, the optimum for the 2-GGCP has value 0.

It is easy to see that the solution  $c$  in which vertices 1 and 4 are colored blue and the other two vertices are colored red, is FLIP-optimal for the 2-GGCP and that this solution has value 1.

This graph can be extended to the general case of  $k$  colors by adding  $k - 2$  vertices all adjacent to the vertices 2 and 3. This extended graph has unit weights on the edges. As this graph is  $k$ -colorable, the optimal value is 0. In the FLIP-optimal solution, the coloring of  $V$  remains as in  $c$  and the  $k - 2$  new vertices are matched to the  $k - 2$  unused colors.  $\square$

Theorems 4 and 5 extend this result to  $m$ -FLIP-optimal and  $VD$ -FLIP-optimal solutions, respectively.

**Theorem 4.** For any constant  $\rho > 1$ , there exists a class of instances and an  $m$ -FLIP-optimal solution for each instance, such that the value of this  $m$ -FLIP-optimal solution is larger than  $\rho$  times the optimal solution value.

**Proof.** Choose a positive integer  $m \geq 2$ . Define a graph  $G = (V, E)$  by  $V = \{1, 2, \dots, 2m + 2\}$ ,  $E = \{\{1, 2\}, \{2, 3\}, \dots, \{2m + 1, 2m + 2\}\}$ , let it have unit weights on the edges, and let  $k = 2$ . This graph is bipartite and therefore the optimum has value 0.

Consider a coloring  $c$  in which the odd numbered vertices between 1 and  $m + 1$  and the even vertices between  $m + 2$  and  $2m + 2$  are colored red and the other vertices are colored blue. Because of the edge  $\{m + 1, m + 2\}$ , the coloring  $c$  has value 1. We claim that  $c$  is  $m$ -FLIP-optimal.

Suppose we flip at most  $m$  vertices to obtain a neighboring coloring  $c'$ . If both  $m + 1$  and  $m + 2$  are flipped, or if neither of them is, then  $c'$  is no better than  $c$ . If exactly one of  $m + 1$  and  $m + 2$  is flipped, say  $m + 1$ , then consider the connected component of the subgraph induced by the flipped vertices containing ver-

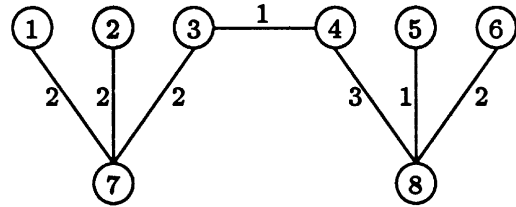


Fig. 1. Graph.

tex  $m + 1$ . Obviously, there is an unflipped vertex  $m'$  with  $1 \leq m' < m + 1$ . Hence, in this case  $c'$  is no better than  $c$  either.

The graph and the locally optimal solution in the above proof can be extended to the general problem with  $k$  colors by adding a  $(k - 2)$ -clique to the graph, and making all vertices of this clique adjacent to the  $2m + 2$  vertices of the bipartite graph. All additional edges have unit weight. The  $m$ -FLIP-optimal solution  $c$  is extended by matching the  $k - 2$  new vertices to the  $k - 2$  unused colors.  $\square$

**Theorem 5.** For any constant  $\rho > 1$ , there exists a class of instances and a  $VD$ -FLIP-optimal solution for each instance, such that the value of this locally optimal solution is more than  $\rho$  times the optimal solution value.

**Proof.** Consider the graph  $G = (V, E)$ , with  $V = \{1, \dots, 8\}$  and  $E = \{\{1, 7\}, \{2, 7\}, \{3, 4\}, \{3, 7\}, \{4, 8\}, \{5, 8\}, \{6, 8\}\}$ . The weights on the edges are depicted in Fig. 1. This graph is bipartite and has optimal value 0. The coloring  $c$  in which we color vertices 7 and 8 red and the other vertices are colored blue has weight 1. We claim that this solution is  $VD$ -FLIP-optimal.

In Fig. 2, we show how the variable-depth search will proceed. All unflipped vertices are denoted by circles and the flipped vertices are denoted by squares. The value next to an unflipped vertex denotes the increase in the objective value if this vertex is flipped. We iteratively choose the best unflipped vertex, which is denoted by an extra circle. In Fig. 2(a), the coloring  $c$  is shown. The best vertex to flip is vertex 3 and we proceed as shown in Figs. 2(b–i). The intermediate solution in Fig. 2(f) and the starting and final solution all have objective value 1; the other intermediate solutions have all value at least 2. Thus, the coloring  $c$  is  $VD$ -FLIP-optimal.

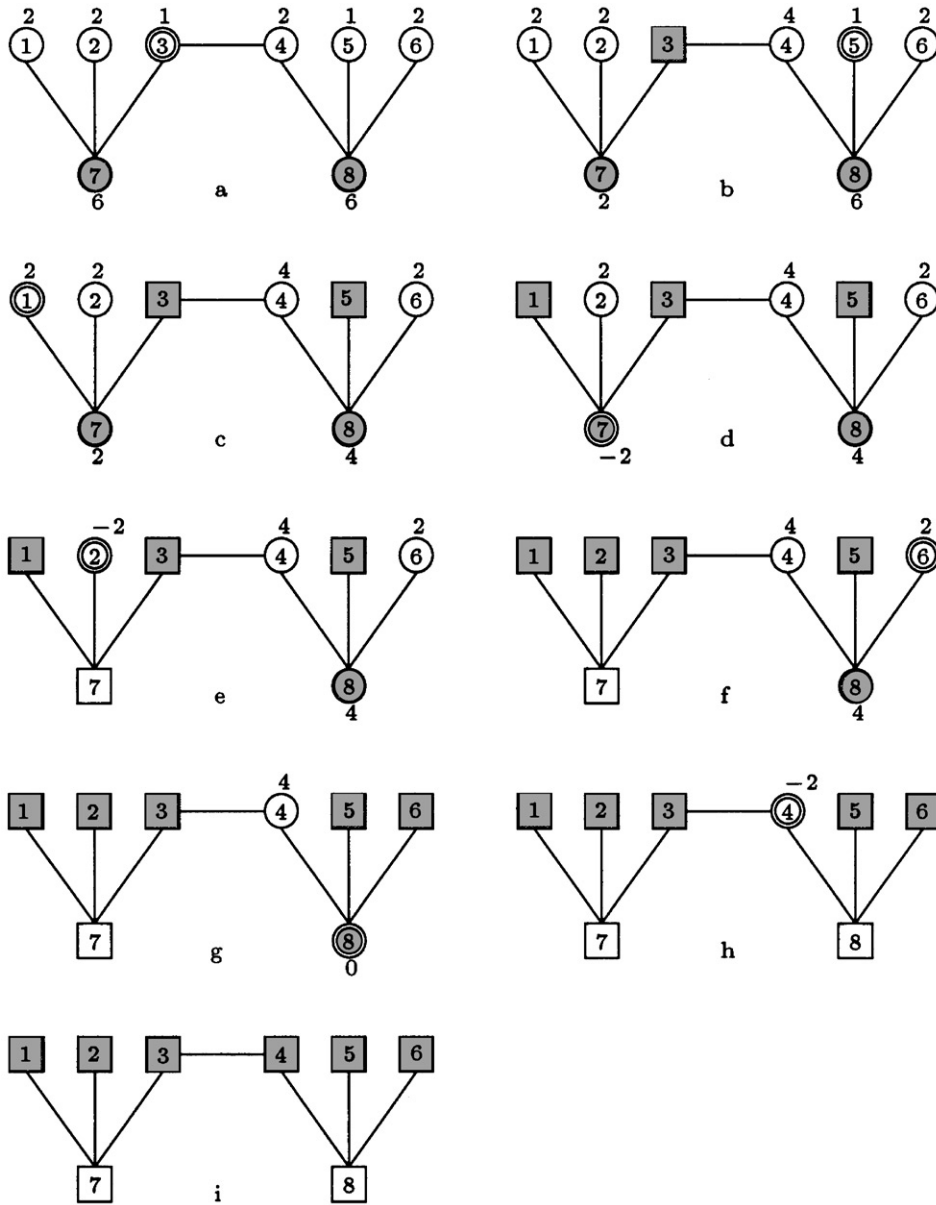


Fig. 2. Variable-depth flip.

We extend this graph to the general case of  $k$  colors by adding a  $(k - 2)$ -clique of which all vertices are adjacent to the vertices in  $V$ . All added edges have weight at least 7 and in the coloring  $c$ , the  $k - 2$  new vertices are matched to the  $k - 2$  unused colors.  $\square$

### 5. Local optima may be hard to find

For the computational complexity of finding local optima, Johnson et al. [4] introduced the class of polynomial-time local search (PLS) problems; see

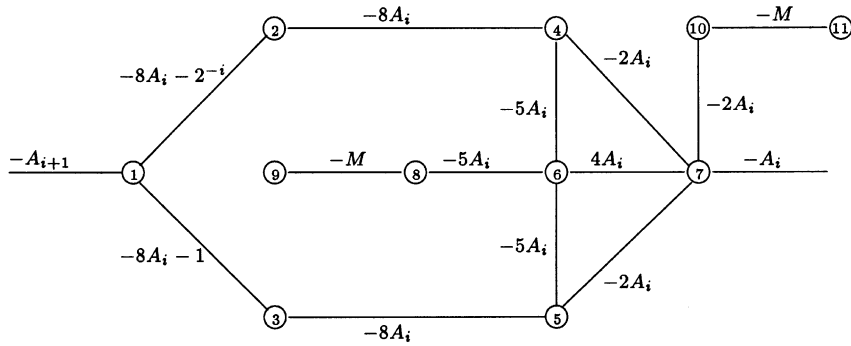


Fig. 3. Module  $i$ ;  $A_i = 20^{i-1}$

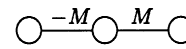


Fig. 4. Chain.

also Yannakakis [12]. This class contains local search problems whose neighborhoods are polynomially searchable. The local search problems of the GGCP with the FLIP,  $m$ -FLIP, or the VD-FLIP neighborhood are in PLS, as the number of FLIP and  $m$ -FLIP neighbors are polynomially bounded in the input size and the procedure to obtain a VD-FLIP neighbor runs in polynomial time. Johnson et al. also defined a reduction among problems in this class and showed that there exist PLS-complete problems. If a local optimum for such a complete problem can be found in polynomial time by whatever means, then for all problems in PLS a local optimum can be found in polynomial time. This is generally not believed to be true, as it would require a general approach to finding local optima at least as clever as the ellipsoid algorithm, since linear programming with the simplex neighborhood is in PLS. On the other hand, Johnson et al. showed that if a PLS problem is NP-hard, then  $NP = co-NP$ .

Schäffer and Yannakakis [11] showed that the max cut problem with the FLIP neighborhood is PLS-complete. It is easy to verify that a FLIP-optimal solution for the max cut problem on a given graph  $G$  also corresponds to a FLIP-optimal solution for the 2-GGCP on the same graph. Furthermore, we can extend the graph  $G$  and the solution so that we obtain a FLIP-optimal solution for the general case. We also know that an  $m$ -FLIP and a VD-FLIP-optimal solution both are FLIP-optimal. This leads us to the following theorem.

**Theorem 6.** *The GGCP with the FLIP,  $m$ -FLIP, or VD-FLIP neighborhood is PLS-complete.*

Schäffer and Yannakakis introduced the notion of *tight* PLS reductions. If there is a tight PLS reduction from a problem  $\Pi_1$  to a problem  $\Pi_2$  and  $\Pi_1$  contains instances and starting solutions for which iterative improvement needs an exponential number of iterations, then there exist instances and starting solutions for  $\Pi_2$  with the same property. By constructing a tight PLS reduction, they showed that finding a FLIP-optimal solution for max cut by iterative improvement may take an exponential number of iterations, regardless of the neighbor selecting rules. Hence, finding a FLIP-optimal solution for the GGCP by iterative improvement takes an exponential number of iterations in the worst case as well. As the reductions for the GGCP with the  $m$ -FLIP and with the VD-FLIP neighborhoods are not tight, this result does not extend to iterative improvement procedures for finding  $m$ -FLIP- and VD-FLIP-optimal solutions. This does not imply that there does not exist a tight PLS reduction for these problems.

To illustrate the exponential number of iterations needed for finding a FLIP-optimal solution, we give an example of a graph and an initial solution for 2-GGCP for which *best improvement*, i.e., always flipping the best vertex, needs an exponential number of iterations to find a FLIP-optimal solution. This graph consists of  $K$  modules with weights on the edges as shown in Fig. 3 for  $i = 1, \dots, K$  and a chain of three additional vertices as shown in Fig. 4.

Vertex 1 is called the input node and vertex 7 is called the output node of a module. The input node of module  $i$  is adjacent to the output node of module  $i+1$ , for  $i=K-1, \dots, 1$ , and the input node of module  $K$  is adjacent to the right most vertex of the chain of Fig. 4. The output node of module 1 is only adjacent to vertices 4, 5, 6, and 10 of this module. An edge of weight  $-M$ , where  $M$  is some large positive value, makes sure that the two vertices incident to this edge have the same color. We claim that the best improvement procedure starting from the solution in which all vertices are colored red, flips the output node of the first module  $2^K$  times.

In our starting solution, only flipping the right most vertex of the chain yields an improvement. This flip results in a solution in which the input node of module  $K$  is *unhappy*, i.e., flipping this vertex improves the solution. The claim is now a direct result of the following lemma.

**Lemma 1.** *If the input node of module  $K$  is the only unhappy node, the output node of module 1 flips  $2^K$  times.*

**Proof.** We show this by induction on  $K$ . For  $K=0$ , the output node is the right most vertex of the chain and it flips once.

Assume the claim is true for  $K-1$  modules. Consider a graph on  $K$  modules of which the only unhappy vertex is the input node of module  $K$ . Flipping this vertex yields a solution in which vertices 2 and 3 of module  $K$  are unhappy. Changing vertex 2 yields an improvement of  $2^{-K}$  and by our choice of edge weights, best improvement will only change the color of this vertex when all other vertices are happy. Hence, vertices 3, 5 and 7 are flipped, which results in a solution in which the input node of module  $K-1$  is unhappy. By induction we know that the output node of module 1 will now flip  $2^{K-1}$  times and then we have found a solution in which all vertices in the modules

$1, \dots, K-1$  are happy and the only unhappy vertex is vertex 2 of module  $K$ . Thus, this vertex is flipped and then successively vertices 4 and 6 and the output node of module  $K$  are flipped. This yields a solution in which the input node of module  $K-1$  is unhappy. By induction, we know that the output node of module 1 flips another  $2^{K-1}$  times and then we have found a FLIP-optimal solution. Hence, the number of times that the output node of module 1 flips is  $2^K$ .  $\square$

## References

- [1] R.C. Carlson, G.L. Nemhauser, Scheduling to minimize interaction cost, *Oper. Res.* 14 (1966) 52–58.
- [2] N. Garg, V.V. Vazirani, M. Yannakakis, Approximate max-flow min-(multi)cut theorems and their applications, *SIAM J. Comput.* 25 (1996) 235–251.
- [3] J. Håstad, Some optimal inapproximability results, in: *Proceedings of the 29th ACM Symposium on Theory of Computing*, 1997, pp. 1–10.
- [4] D.S. Johnson, C.H. Papadimitriou, M. Yannakakis, How easy is local search? *J. Comput. System Sci.* 37 (1988) 79–100.
- [5] V. Kann, S. Khanna, J. Lagergren, A. Panconesi, On the hardness of approximating max  $k$ -cut and its dual, *Chicago J. Theoret. Comput. Sci.*, 1997, <http://cjtcs.cs.uchicago.edu/>.
- [6] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller, J.W. Thatcher (Eds.), *Complexity of Computer Computations*, Plenum Press, New York, 1972, pp. 85–103.
- [7] B.W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, *Bell System Tech. J.* 49 (1970) 291–307.
- [8] A.W.J. Kolen, J.K. Lenstra, Combinatorics in operations research, in: R.L. Graham, M. Grötschel, L. Lovász (Eds.), *Handbook of Combinatorics*, Elsevier Science, Amsterdam, 1995, pp. 1875–1910.
- [9] J.K. Lenstra, Sequencing by enumerative methods, Ph.D. Thesis, Universiteit van Amsterdam, 1976.
- [10] S. Sahni, T. Gonzalez,  $P$ -Complete approximation problems, *J. ACM* 23 (1976) 555–565.
- [11] A.A. Schäffer, M. Yannakakis, Simple local search problems that are hard to solve, *SIAM J. Comput.* 20 (1991) 56–87.
- [12] M. Yannakakis, Computational complexity, in: E.H.L. Aarts, J.K. Lenstra (Eds.), *Local Search in Combinatorial Optimization*, Wiley, Chichester, 1997, pp. 19–55 (Chapter 2).