Parameterized Algorithms for Load Coloring Problem

Gregory Gutin and Mark Jones
Department of Computer Science
Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK
gutin|markj@cs.rhul.ac.uk

Abstract

One way to state the Load Coloring Problem (LCP) is as follows. Let $G = (V, E)$ be a graph and let $f : V \to \{\text{red, blue}\}$ be a 2-coloring. An edge $e \in E$ is called red (blue) if both end-vertices of $e$ are red (blue). For a 2-coloring $f$, let $r'_f$ and $b'_f$ be the number of edges of red and blue edges and let $\mu_f(G) = \min\{r'_f, b'_f\}$. Let $\mu(G)$ be the maximum of $\mu_f(G)$ over all 2-colorings.

We introduce the parameterized problem $k$-LCP of deciding whether $\mu(G) \geq k$, where $k$ is the parameter. We prove that this problem admits a kernel with at most $7k$. Ahuja et al. (2007) proved that one can find an optimal 2-coloring on trees in polynomial time. We generalize this by showing that an optimal 2-coloring on graphs with tree decomposition of width $t$ can be found in time $O^*(2^t)$. We also show that either $G$ is a Yes-instance of $k$-LCP or the treewidth of $G$ is at most $2k$. Thus, $k$-LCP can be solved in time $O^*(4^k)$.

1 Introduction

For a graph $G = (V, E)$ with $n$ vertices, $m$ edges and maximum vertex degree $\Delta$, the load distribution of a 2-coloring $f : V \to \{\text{red, blue}\}$ is a pair $(r_f, b_f)$, where $r_f$ is the number of edges with at least one end-vertex colored red and $b_f$ is the number of edges with at least one end-vertex colored blue. We wish to find a coloring $f$ such that the function $\lambda_f(G) := \max\{r_f, b_f\}$ is minimized. We will denote this minimum by $\lambda(G)$ and call this problem LOAD COLORING PROBLEM (LCP). The LCP arises in Wavelength Division Multiplexing, the technology used for constructing optical communication networks [1, 9]. Ahuja et al. [1] proved that the problem is NP-hard and gave a polynomial time algorithm for optimal colorings of trees. For graphs $G$ with genus $g > 0$, Ahuja et al. [1] showed that a 2-coloring $f$ such that $\lambda_f(G) \leq \lambda(G)(1 + o(1))$ can be computed in $O(n + g \log n)$-time, if the maximum degree satisfies $\Delta = o\left(\frac{m^2}{n^2}\right)$ and an embedding is given.

For a 2-coloring $f : V \to \{\text{red, blue}\}$, let $r'_f$ and $b'_f$ be the number of edges whose end-vertices are both red and blue, respectively (we call such edges red and blue, respectively). Let $\mu_f(G) := \min\{r'_f, b'_f\}$ and let $\mu(G)$ be the maximum of $\mu_f(G)$ over all 2-colorings of $V$. It is not hard to see (and it is proved in Remark 1.1 of [1]) that $\lambda(G) = m - \mu(G)$ and so the LCP is equivalent to maximizing $\mu_f(G)$ over all 2-colorings of $V$.
In this paper we introduce and study the following parameterization of LCP.

\[ \text{\textbf{k-Load Coloring Problem (k-LCP)}} \]

\textbf{Input:} A graph \( G = (V,E) \) and an integer \( k \).

\textbf{Parameter:} \( k \)

\textbf{Question:} Is \( \mu(G) \geq k \)? (Equivalently, is \( \lambda(G) \leq m - k \)?)

We provide basics on parameterized complexity and tree decompositions of graphs in the next section. In Section 3, we show that \( k \)-LCP admits a kernel with at most \( 7k \) vertices. Interestingly, to achieve this linear bound, only two simple reduction rules are used. In Section 4, we generalise the result of Ahuja et al. \[1\] on trees by showing that an optimal 2-coloring for graphs with tree decomposition of width \( t \) can be obtained in time \( 2^t n^{O(1)} \). We also show that either \( G \) is a Yes-instance of \( k \)-LCP or the treewidth of \( G \) is at most \( 2k \). As a result, \( k \)-LCP can be solved in time \( 4^k n^{O(1)} \).

We conclude the paper in Section 5 by stating some open problems.

2 Basics on Fixed-Parameter Tractability, Kernelization and Tree Decompositions

A \textit{parameterized problem} is a subset \( L \subseteq \Sigma^* \times \mathbb{N} \) over a finite alphabet \( \Sigma \). \( L \) is \textit{fixed-parameter tractable} if the membership of an instance \( (x,k) \) in \( \Sigma^* \times \mathbb{N} \) can be decided in time \( f(k)|x|^{O(1)} \), where \( f \) is a function of the parameter \( k \) only. It is customary in parameterized algorithms to often write only the exponential part of \( f(k) : O^*(t(k)) := O(t(k)(kn)^{O(1)}) \).

Given a parameterized problem \( L \), a \textit{kernelization} of \( L \) is a polynomial-time algorithm that maps an instance \( (x,k) \) to an instance \( (x',k') \) (the \textit{kernel}) such that (i) \( (x,k) \in L \) if and only if \( (x',k') \in L \), (ii) \( k' \leq g(k) \), and (iii) \( |x'| \leq g(k) \) for some function \( g \). The function \( g(k) \) is called the \textit{size} of the kernel.

It is well-known that a parameterized problem \( L \) is fixed-parameter tractable if and only if it is decidable and admits a kernelization. Due to applications, low degree polynomial size kernels are of main interest. Unfortunately, many fixed-parameter tractable problems do not have kernels of polynomial size unless the polynomial hierarchy collapses to the third level, see, e.g., \[2\ \[3\ \[4\]. For further background and terminology on parameterized complexity we refer the reader to the monographs \[4\ \[8\.

\textbf{Definition 1.} A \textit{tree decomposition} of a graph \( G = (V,E) \) is a pair \( (\mathcal{X},\mathcal{T}) \), where \( \mathcal{T} = (I,F) \) is a tree and \( \mathcal{X} = \{X_i : i \in I\} \) is a collection of subsets of \( V \) called \textit{bags}, such that:

1. \( \bigcup_{i \in I} X_i = V \);
2. For every edge \( xy \in E \), there exists \( i \in I \) such that \( \{x,y\} \subseteq X_i \);
3. For every \( x \in V \), the set \( \{i : x \in X_i\} \) induces a connected subgraph of \( \mathcal{T} \).

The \textit{width} of \( (\mathcal{T},\mathcal{X}) \) is \( \max_{i \in I} |X_i| - 1 \). The \textit{treewidth} of a graph \( G \) is the minimum width of all tree decompositions of \( G \).
To distinguish between vertices of $G$ and $T$, we call vertices of $T$ nodes. We will often speak of a bag $X_i$ interchangeably with the node $i$ to which it corresponds in $T$. Thus, for example, we might say two bags are neighbors if they correspond to nodes in $T$ which are neighbors. We define the descendants of a bag $X_i$ as follows: every child of $X_i$ is a descendant of $X_i$, and every child of a descendant of $X_i$ is a descendant of $X_i$.

**Definition 2.** A nice tree decomposition of a graph $G = (V,E)$ is a tree decomposition $(X,T)$ such that $T$ is a rooted tree, and each node $i$ falls under one of the following classes:

- **$i$ is a Leaf node**: Then $i$ has no children;
- **$i$ is an Introduce node**: Then $i$ has a single child $j$, and there exists a vertex $v \not\in X_j$ such that $X_i = X_j \cup \{v\}$;
- **$i$ is a Forget node**: Then $i$ has a single child $j$, and there exists a vertex $v \in X_j$ such that $X_i = X_j \setminus \{v\}$;
- **$i$ is a Join node**: Then $i$ has two children $h$ and $j$, and $X_i = X_h \cup X_j$.

It is known that any tree decomposition of a graph can be transformed into a tree decomposition of the same width.

**Lemma 1.** [7] Given a tree decomposition with $O(n)$ nodes of a graph $G$ with $n$ vertices, we can construct, in time $O(n)$, a nice tree decomposition of $G$ of the same width and with at most $4n$ nodes.

## 3 Linear Kernel

For a vertex $v$ of a graph $G = (V,E)$ and set $X \subseteq V$, let $\deg_X(v)$ denote the number of neighbors of $v$ in $X$. If $X = V$, we will write $\deg(v)$ instead of $\deg_V(v)$.

**Lemma 2.** Let $G = (V,E)$ be a graph with no isolated vertices, with maximum degree $\Delta \geq 2$ and let $|V| \geq 5k$. If $|V| \geq 4k + \Delta$, then $(G,k)$ is a Yes-instance of $k$-LCP.

**Proof.** Suppose that $|V| \geq 4k + \Delta$, but $(G,k)$ is a No-instance of $k$-LCP.

Let $M$ be a maximum matching in $G$ and let $Y$ be the set of vertices which are not end-vertices of edges in $M$. If $M$ has at least $2k$ edges, then we may color half of them blue and half of them red, so we conclude that $|M| < 2k$.

For an edge $e = uv$ in $M$, let $\deg_Y(e) = \deg_Y(u) + \deg_Y(v)$, that is the number of edges between a vertex in $Y$ and a vertex of $e$.

**Claim** For any $e$ in $M$, $\deg_Y(e) \leq \max\{\Delta - 1, 2\}$.

**Proof of Claim:** Suppose that $\deg_Y(e) \geq \Delta$ and let $e = uv$. As $u$ and $v$ are adjacent, $d_Y(u)$ and $d_Y(v)$ are each less than $\Delta$. But as $\deg_Y(u) + \deg_Y(v) = \deg_Y(e) \geq \Delta$, it follows that $\deg_Y(u) \geq 1$ and $\deg_Y(v) \geq 1$. Then either $u$ and $v$ have only one neighbor in $Y$, which is adjacent to both of them (in which case $\deg_Y(e) = 2$), or there exist vertices $x \neq y \in Y$ such that $x$ is adjacent to $u$ and $y$ is adjacent to $v$. 


Then $M$ is not a maximum matching, as $xuvy$ is an augmenting path, which proves the claim.

Now let $M'$ be a subset of edges of $M$ such that

$$
\sum_{e' \in M'} \deg_Y(e') \geq k - |M'|, \quad (1)
$$

and

$$
\left[ \sum_{e' \in M'} \deg_Y(e') \right] - \deg_Y(e) < k - |M'|, \text{ for any } e \in M'. \quad (2)
$$

To see that $M'$ exists observe first that $M'$ satisfies (1). Indeed, suppose it is not true. Then $|V| < |V(M)| + k - |M| = k + |M| < 5k$, a contradiction with our assumption that $|V| \geq 5k$. Now let $M'$ be the minimal subset of $M$ that satisfies (1), and observe that by minimality $M'$ also satisfies (2).

Observe that $|M'| \leq k$. Then by the Claim, we have that $\sum_{e' \in M'} \deg_Y(e') \leq k - |M'| + \Delta$. Color $M'$ and all neighbors of $M'$ in $Y$ red, and note that there are at most $k - |M'| + \Delta + 2 |M'| = k + |M'| + \Delta \leq 2k + \Delta$ such vertices. The number of red edges is at least $k - |M'| + |M'| = k$.

Color the remaining vertices of $G$ blue. By assumption there are at least $4k + \Delta - 2k \geq 2k$ such vertices. As $G$ contains no isolated vertices and $M$ is a maximum matching, the blue vertices in $Y$ have neighbors in the vertices of $M \setminus M'$. Thus, every blue vertex has a blue neighbor. It follows that there are at least $2k/2 = k$ blue edges. Thus, $(G, k)$ is a Yes-instance of $k$-LCP.

We will use the following reduction rules for a graph $G$.

**Reduction Rule 1.** Delete isolated vertices.

**Reduction Rule 2.** If there exists a vertex $x$ and set of vertices $S$ such that $|S| > k$ and every $s \in S$ has $x$ as its only neighbor, delete a vertex from $S$.

**Theorem 1.** The problem $k$-LCP has a kernel with at most $7k$ vertices.

**Proof.** Assume that $G$ is a graph reduced by Rules 1 and 2. Assume also that $G$ is a No-instance. We will prove that $G$ has at most $7k$ vertices.

**Claim A.** There is no pair $x, y$ of distinct vertices such that $\deg(x) > 2k$ and $\deg(y) > k$.

**Proof of Claim:** Suppose such a pair $x, y$ exists. Color $y$ and $k$ of its neighbors, not including $x$, red. This leaves $x$ and at least $k$ of its neighbors uncolored. Color $x$ and $k$ of its neighbors blue.

Construct $G' = (V', E')$ as follows. Let $x$ be a vertex in $G$ of maximum degree. Let $S$ be the vertices of $G$ whose only neighbor is $x$. Then let $G' = G - (S \cup \{x\})$. The next claim follows from the definition of $G'$.

**Claim B.** The graph $G'$ has no isolated vertices.

4
The next claim follows from the definition of \( G' \) and Claim A.

**Claim C.** If the maximum degree in \( G' \) is at least 2\( k \), then \( G \) is a Yes-instance of \( k \)-LCP.

The next claim follows from the definition of \( G' \) and Rule 2.

**Claim D.** We have \(|V| \leq |V'| + k + 1\).

Observe that if \( G' \) was a Yes-instance of \( k \)-LCP then so would be \( G \). Thus, \( G' \) is a No-instance of \( k \)-LCP. If the maximum degree in \( G' \) is 1, then we may assume that \(|V'| < 4k\) as otherwise by Claim B \( G' \) is a matching with at least 2\( k \) edges and so \((G', k)\) is a Yes-instance. So, the maximum degree of \( G' \) is at least 2. By Claim C and Lemma 2 we may assume that \(|V'| \leq 4k + 2k - 1 = 6k - 1\). Then by Claim D, \(|V| \leq 6k - 1 + k + 1 = 7k\).

Using the 7\( k \) kernel of this section we can get a simple algorithm that tries all 2-colourings of vertices of the kernel. The running time is \( O^*(2^{7k}) = O^*(128^k) \). In the next section, we obtain an algorithm of running time \( O^*(4^k) \).

### 4 Load Coloring Parameterized by Treewidth

**Theorem 2.** Given a tree decomposition of \( G \) of width \( t \), we can solve LCP in time \( O(2^{t+1}(k+1)^4n^6) \).

**Proof.** Let \( G = (V, E) \) be a graph and let \((\mathcal{X}, T)\) be a tree decomposition of \( G \) of width \( t \), where \( T = (I, F) \) and \( \mathcal{X} = \{X_i : i \in I\} \). By Lemma 4 we may assume that \((\mathcal{X}, T)\) is a nice tree decomposition.

Let \( \psi(X_i) \) denote the set of vertices in \( V \) which appear in either \( X_i \) or a descendant of \( X_i \). For each \( i \in I \), each \( S \subseteq X_i \) and each \( r, b \in \{0, 1, \ldots, k\} \), define the boolean-valued function \( F(X_i, S, r, b) \) to be true if there exists a 2-coloring \( f : \psi(X_i) \rightarrow \{\text{red}, \text{blue}\} \) such that \( f^{-1}(\text{red}) \cap X_i = S \) and there are at least \( r \) red edges and at least \( b \) blue edges in \( G[\psi(X_i)] \). We will say such an \( f \) satisfies \( F(X_i, S, r, b) \).

Let \( X_0 \) denote the bag which is the root of \( T \), and observe that \( G \) is a Yes-instance if and only if \( F(X_0, S, k, k) \) is true for some \( S \subseteq X_0 \). We now show how to calculate \( F(X_i, S, r, b) \) for each \( X_i, S, r \) and \( b \). Assume we have already calculated \( F(X_j, S', r', b') \) for all descendants \( j \) of \( i \) and all values of \( S', r', b' \). Our calculation of \( F(X_i, S, r, b) \) depends on whether \( i \) is a Leaf, Introduce, Forget or Join node.

**\( i \) is a Leaf node:** As \( \psi(X_i) = X_i \) there is exactly one 2-coloring \( f : \psi(X_i) \rightarrow \{\text{red}, \text{blue}\} \) such that \( f^{-1}(\text{red}) \cap X_i = S \). It is sufficient to set \( F(X_i, S, r, b) \) to be true if and only if this coloring gives at least \( r \) red edges and at least \( b \) blue edges.

**\( i \) is an Introduce node:** Let \( j \) be the child of \( i \) and let \( v \) be the vertex such that \( X_i \setminus X_j = \{v\} \). If \( v \in S \), let \( r^* \) be the number of neighbors of \( v \) in \( S \). Then for any 2-coloring of \( \psi(X_i) \), the number of red edges in \( G[\psi(X_i)] \) is exactly the number of red edges in \( G[\psi(X_j)] \) plus \( r^* \), and the number of blue edges is the same in \( G[\psi(X_j)] \) and \( G[\psi(X_i)] \). Therefore we may set \( F(X_i, S, r, b) \) to be true if and only if \( F(X_j, S \setminus \{v\}, \max(r - r^*, 0), b) \) is true. Similarly if \( v \notin S \), we may set \( F(X_j, S, r, b) \) to be true if and only if \( F(X_j, S \setminus \{v\}, r, \max(b - b^*, 0)) \) is true, where \( b^* \) is the number of neighbors of \( v \) in \( X_i \setminus S \).
\textbf{i is a Forget node}: Let \(j\) be the child of \(i\) and let \(v\) be the vertex such that \(X_j \setminus X_i = \{v\}\). Observe that \(\psi(X_i) = \psi(X_j)\), and so any 2-coloring of \(\psi(X_i)\) has exactly the same number of red edges or blue edges in \(G[\psi(X_i)]\) and \(G[\psi(X_j)]\). Therefore we may set \(F(X_i, S, r, b)\) to be true if and only if \(F(X_j, S, r, b)\) or \(F(X_j, S \cup \{v\}, r, b)\) is true.

\textbf{i is a Join node}: Let \(h\) and \(j\) be the children of \(i\). Let \(r^*\) be the number of red edges in \(X_i\).

Then observe that if there is a 2-coloring on \(\psi(X_i)\) consistent with \(S\) such that there are \(r_h\) red edges in \(G[\psi(X_h)]\) and \(r_j\) red edges in \(G[\psi(X_j)]\), then \(r_h, r_j \geq r^*\), and the number of red edges in \(G[\psi(X_i)]\) is \(r_h + r_j - r^*\).

Let \(r' = \min(r^*, k)\). Then if there are at least \(r \leq k\) red edges in \(G[\psi(X_i)]\), there are at least \(r_h\) red edges in \(G[\psi(X_h)]\) for some \(r_h\) such that \(r' \leq r_h \leq k\), and there are at least \(r - r_h + r'\) red edges in \(G[\psi(X_i)]\). Similarly let \(b^*\) be the number of blue edges in \(X_i\), and let \(b' = \min(b^*, k)\). Then if there are at least \(b \leq k\) blue edges in \(G[\psi(X_i)]\), there are at least \(b_h\) blue edges in \(G[\psi(X_h)]\) for some \(b_h\) such that \(b' \leq b_h \leq k\), and there are at least \(b - b_h + b'\) blue edges in \(G[\psi(X_i)]\).

Therefore, we may set \(F(X_i, S, r, b)\) to be true if and only if \(r_h, b_h\) such that \(r' \leq r_h \leq k\), \(b' \leq b_h \leq k\), and both \(F(X_h, S, r_h, b_h)\) and \(F(X_j, S, \max(r - r_h + r', 0), \max(b - b_h + b', 0))\) are true.

It remains to analyse the running time of the algorithm.

We first analyse the running time of calculating a single value \(F(X_i, S, r, b)\) assuming we have already calculated \(F(X_j, S', r', b')\) for all descendants \(j\) of \(i\) and all values of \(S', r', b'\). In the case of a Leaf node, we can calculate \(F(X_i, S, r, b)\) in \(O(n + m)\) time.

In the case of an Introduce node, we need to check how many values need to be calculated. As there are at most \((k + 1)^2\) such choices and so we can calculate \(F(X_i, S, r, b)\) in \(O((k + 1)^2(n + m))\) time.

It remains to check how many values need to be calculated. As there are at most \(O(n)\) bags \(X_i\), at most \(2^{k+1}\) choices of \(S \subseteq X_i\), and at most \(k + 1\) choices for each of \(r\) and \(b\), the number of values \(F(X_i, S, r, b)\) we need to consider is \(O(n2^{k+1}(k + 1)^2)\). As each value can be calculated in polynomial time, overall we have running time \(O(2^{k+1}(k + 1)^4n(n + m)) = O(2^{k+1}(k + 1)^4n^3)\).

We will combine Theorem 2 with the following lemma to obtain Theorem 3.

\textbf{Lemma 3}. For a graph \(G\), in polynomial time, we can either determine that \(G\) is a Yes-instance of \(k\)-LCP, or construct a tree decomposition of \(G\) of width at most \(2k\).

\textbf{Proof}. If every component of \(G\) has at most \(k - 1\) edges then we may easily construct a tree decomposition of \(G\) of width at most \(k - 1\) (as each component has at most \(k\) vertices).

Now assume that \(G\) has a component with at least \(k\) edges. By starting with a single vertex in a component with at least \(k\) edges, and adding adjacent vertices one at a time, construct a minimal set of vertices \(X\) such that \(G[X]\) is connected and \(|E(X)| \geq k\), where \(E(X)\) is the set of edges with both end-vertices in \(X\). Let \(v\) be
the last vertex added to $X$. Then $G[X \setminus \{v\}]$ is connected and $|E(X \setminus \{v\})| < k$. It follows that $|X \setminus \{v\}| \leq k$ and so $|X| \leq k + 1$.

Now if $|E(V \setminus X)| \geq k$, then we may obtain a solution for $k$-$\text{LCP}$ by coloring all of $X$ red and all of $V \setminus X$ blue. Otherwise, we may construct a tree decomposition of $G[V \setminus X]$ of width at most $k - 1$. Now add $X$ to every bag in this tree decomposition. Observe that the result satisfies the conditions of a tree decomposition and has width at most $k - 1 + |X| \leq 2k$. □

**Theorem 3.** There is an algorithm of running time $O^*(4^k)$ to solve $k$-$\text{LCP}$.

### 5 Open Problems

Our kernel and fixed-parameter algorithm seem to be close to optimal: we do not believe that $k$-$\text{LCP}$ admits $o(k)$-vertex kernel or $2^{o(k)}$ running time algorithm unless the Exponential Time Hypothesis fails. It would be interesting to prove or disprove it. It would also be interesting to obtain a smaller kernel or faster algorithm for $k$-$\text{LCP}$.

**Acknowledgments** We are grateful to the referees for some suggestions that improved results of our paper.

### References

[1] N. Ahuja, A. Baltz, B. Doerr, A. Privitivy and A. Srivastav, On the Minimum Load Coloring Problem. J. Discr. Alg. 5(3): 533–545 (2007).

[2] H.L. Bodlaender, R.G. Downey, M.R. Fellows, and D. Hermelin, On problems without polynomial kernels. J. Comput. Syst. Sci. 75(8):423–434, 2009.

[3] H.L. Bodlaender, B.M.P. Jansen and S. Kratsch, Cross-Composition: a New technique for kernelization lower bounds. In STACS 2011, LIPICS 9:165176, 2011.

[4] H.L. Bodlaender, S. Thomassé, and A. Yeo, Kernel bounds for disjoint cycles and disjoint paths. In ESA 2009, Lect. Notes Comput. Sci. 5757:635–646, 2009.

[5] R.G. Downey and M.R. Fellows, Parameterized Complexity, Springer, 1999.

[6] J. Flum and M. Grohe, Parameterized Complexity Theory, Springer, 2006.

[7] T. Kloks, Treewidth: Computations and Approximations, Lect. Notes Comput. Sci., vol. 842. Springer, 1994.

[8] R. Niedermeier, Invitation to Fixed-Parameter Algorithms. Oxford UP, 2006.

[9] C. Siva Ram Murthy and M. Guruswamy, WDM Optical Networks, Concepts, Design, and Algorithms, Prentice Hall, 2001.