Algorithmic aspects of graph-indexed random walks

Jan Bok

Computer Science Institute, Charles University, Malostranské náměstí 25, 11800, Prague, Czech Republic, email: bok@iuuk.mff.cuni.cz.

Abstract. We study three problems regarding the so called graph-indexed random walks (or equivalently Lipschitz mappings of graphs).
1. Computing the average range of graph-indexed random walk of a graph.
2. Computing the maximum range of graph-indexed random walk for a given graph.
3. Deciding if we can extend partial GI random walk into full GI random walk for a given graph.
We show that while the first problem is \#P-complete, the other two problems can be solved in polynomial time.

1 Introduction

Graph-indexed random walks (or also M-Lipschitz mapping of graphs) are a generalization of standard random walk on \(\mathbb{Z}\). Also, this concept has an important connections to statistical physics, namely to gas models (as is described by Zhao [26] and Cohen et al. [6]). Understanding the structure of all M-Lipschitz mappings of a given graph and corresponding parameters is also a point of interest because it can describe the expected behavior of a random homomorphism to a suitable graph.

Graph-indexed random walks and average range were studied for example in [1,8,2,25,21]. However, we emphasize that algorithmic aspects of graph-indexed random walks were, by our best knowledge, not studied yet.

In the following text we will not mix the terms M-Lipschitz mapping and graph-indexed random walk and we will use only the first term.

1.1 Preliminaries

In this text we use the standard notation as for example in Diestel’s monograph [7].

We will often use the definition of graph diameter.

Definition 1. The diameter \( \text{diam}(G) \) of a graph \( G \) is the maximum distance between any two vertices of \( G \). We define the distance of two vertices \( u,v \in V(G) \), \( d(u,v) \), as the length of the shortest path between these two vertices. The distance of vertex to itself is defined as zero. The distance between two vertices from different components is defined as \( \infty \).
A graph homomorphism between digraphs $G$ and $H$ is a mapping $f : V(G) \to V(H)$ such that for every edge $uv \in E(G)$, $f(u)f(v) \in E(H)$. That means that graph homomorphism is an adjacency-preserving mapping between the vertex sets of two digraphs. The set $I := \{w \in V(H) \mid \exists v \in V(G) : f(v) = w\}$ for a graph homomorphism $f$ is called the homomorphic image of $f$.

For a comprehensive and more complete source on graph homomorphisms, the reader is invited to see [13]. A quick introduction is given in [11] as well.

**Definition 2.** For $M \in \mathbb{N}$, an $M$-Lipschitz mapping of a connected graph $G = (V, E)$ with root $v_0 \in V$ is a mapping $f : V \to \mathbb{Z}$ such that $f(v_0) = 0$ and for every edge $(u, v) \in E$ it holds that $|f(u) - f(v)| \leq M$. The set of all $M$-Lipschitz mappings of a graph $G$ is denoted by $\mathcal{L}_M(G)$.

By the term Lipschitz mappings of graph we mean the union of sets of $M$-Lipschitz mappings for every $M \in \mathbb{N}$.

The importance of having rooted graphs is the following. We want to have finitely many Lipschitz mappings for a fixed graph $G$. Mappings with $f(v_0) \neq 0$ are just linear shifts of some mapping with $f(v_0) = 0$. Formally, consider a mapping $f'$ with $f'(v_0) = a$. Then we can define a linear transformation $T_a$ as $T_a(f) \mapsto f - a$. Applying $T_a$ to $f'$ yields a Lipschitz mapping of $G$ with $v_0$ as its root.

We note that we are interested in connected graphs only. Components without the root would also allow infinitely many new $M$-Lipschitz mappings.

In literature, we will often meet a slightly different definition of $1$-Lipschitz mappings. In it the restriction $|f(u) - f(v)| \leq 1$, for all $uv \in E$, is removed and instead, the restriction $|f(u) - f(v)| = 1$, for all $uv \in E$, is added. In [21] authors call these mappings strong Lipschitz mappings. We generalize this in the following definition.

**Definition 3.** For $M \in \mathbb{N}$, a strong $M$-Lipschitz mapping of a connected graph $G = (V, E)$ with root $v_0 \in V$ is a mapping $f : V \to \mathbb{Z}$ such that $f(v_0) = 0$ and for every edge $(u, v) \in E$ it holds that $|f(u) - f(v)| = M$. The set of all $M$-Lipschitz mappings of a graph $G$ is denoted by $\mathcal{L}_{\pm M}(G)$.

Note that strong $M$-Lipschitz mappings are a special case of $M$-Lipschitz mappings of graph. Also, $B$-Lipschitz mappings are a superset of $A$-Lipschitz mapping whenever $B \geq A$. See Figure[1] for the Hasse diagram of various types of Lipschitz mappings.

We emphasize the following lemma.

**Lemma 1.** A graph has a strong $M$-Lipschitz mapping if and only if it is bipartite.

**Proof.** First of all, observe the fact that in strong $M$-Lipschitz mapping, all vertices are mapped to some number of the form $k \cdot M$, where $k \in \mathbb{Z}$.

Consider a non-bipartite graph $G$ and a strong $M$-Lipschitz mapping $f$ of $G$. There is a well-known characterization of bipartite graphs:
A graph is bipartite if and only if it does not contain a cycle of odd length as a subgraph.

Therefore, $G$ contains some odd cycle $C$ with edges $v_1v_2, \ldots, v_lv_1$. Let us denote

$$e_i := f(v_{(i+1 \mod l)}) - f(v_{(i \mod l)}), \forall i \in \{1, \ldots, n\}.$$ 

We see that $e_i \in \{+M, -M\}$. Moreover, $\sum_{i=0}^{n} e_i = 0$ from the definition of $e_i$. However, this sum has an odd number of summands and thus we get a contradiction. $\Box$

Analogously, by the term \textit{strong Lipschitz mappings of graph} we mean the union of sets of strong $M$-Lipschitz mappings for every $M \in \mathbb{N}$.

![Diagram](Fig. 1: The Hasse diagram of different types of Lipschitz mappings of graphs.)

First of all, let us define the \textit{range} of a mapping.

\textbf{Definition 4.} The \textit{range} of a Lipschitz mapping $f$ of $G$ is the size of the homomorphic image of $f$. Formally:

$$r_G(f) := |\{z \in \mathbb{Z} | z = f(v) \text{ for some } v \in V(G)\}|.$$

We define the \textit{average range} and the \textit{maximum range} of graph $G$ as follows.

\textbf{Definition 5.} (Average range) The \textit{average range} of graph $G$ over all $M$-Lipschitz mappings is defined as

$$r_M(G) := \frac{\sum_{f \in \mathcal{L}(G)} r(f)}{|\mathcal{L}(G)|}.$$
Definition 6. (Maximum range) The maximum range over all $M$-Lipschitz mappings of graph $G$ is defined as

$$r_{max}^m(G) = \max_{f \in \mathcal{L}_M(G)} r(f).$$

We can view this quantity as the expected size of the homomorphic image of an uniformly picked random $M$-Lipschitz mapping of $G$.

Whenever we want to talk about the counterparts of these definitions for strong Lipschitz mappings, we denote it with ± in subscript. For example, $r_{\pm M}$ is the average range of strong $M$-Lipschitz mapping of graph.

Whenever we write average range or maximum range without saying which $M$-Lipschitz mappings we use, it should be clear from the context what $M$ do we mean.

It is worth noting that for computing the average range and the maximum range, the choice of root does not matter. That is why we often omit the details of picking the root.

2 Average range

Our aim is to settle the complexity of the following natural problem associated with the average range.

**Problem:** Average range computation problem – AvGRANGE

**Input:** A connected graph $G$.

**Question:** What is the average range of 1-Lipschitz mappings of $G$, i.e. $r_1(G)$?

We will use the recent result of Focke et al. [10] to prove that this problem is #P-complete. To state their result, we need to define the problem of counting surjective homomorphisms of a graph.

**Problem:** #SurHom(H)

**Input:** A connected irreflexive graph $G$.

**Question:** What is the number of surjective homomorphisms from $G$ to the fixed graph $H$.

Focke et al. proved the following result.

**Theorem 1.** [10] Let $H$ be a graph. If every connected component of $H$ is a reflexive clique or an irreflexive biclique, then #SurHom(H) is in FP. Otherwise, #SurHom(H) is #P-complete.

**Theorem 2.** The problem AvGRANGE is #P-complete.
Proof. We can without the loss of generality assume that the diameter of $G$ – the input graph – is equal to four. We will reduce the problem $\#\text{SurHom}(P_4)$ to $\text{AvgRange}$.

We denote by $\text{Output(Pr)}$ the result of computing a problem $\text{Pr}$. With this notation in hand, we observe that:

$$\text{Output(\text{AvgRange}(G))} = \frac{1 + \sum_{i=1}^{4} i \cdot \text{Output(\#\text{SurHom}(P_i)(G))}}{1 + \sum_{i=1}^{4} \text{Output(\#\text{SurHom}(P_i)(G))}}.$$  

Since $\#\text{SurHom}(P_i)(G)$ is in FP for $i \leq 3$ we can compute $\#\text{SurHom}(P_4)(G)$ from $\text{AvgRange}(G)$ and thus we finished the reduction. \hfill $\blacksquare$

As for the case of strong Lipschitz mappings we can proceed analogously for the problem $\text{StrongAvgRange}$. Thus we conclude.

Theorem 3. The problem $\text{StrongAvgRange}$ is $\#P$-complete.

3 Maximum range

In this section we will show how can we algorithmically compute the maximum range of a given graph. Also, we will show the relation of this parameter to other existing results.

3.1 Diameter

In this section we observe one important fact giving us an upper bound on the range of a graph. Then we will show that this upper bound is tight.

We will first prove an important, yet easy lemma.

Lemma 2. For any connected graph $G$ with $\text{diam}(G)$ and every $M$-Lipschitz mapping $f$ of $G$, holds that

$$r(f) \leq M \cdot (\text{diam}(G) + 1).$$

Proof. The existence of $f$ with $r(f) > M \cdot (\text{diam}(G) + 1)$ would imply the existence of a path subgraph in $G$ with endpoints $u, v$ such that their images would be in distance $|f(u) - f(v)| = M \cdot (\text{diam}(G) + 1)$. However, that would mean that all paths between $u$ and $v$ have to map to some connected subgraph of $Z$ of size greater than $M \cdot (\text{diam}(G) + 1)$ which is a contradiction with the definition of diameter. \hfill $\blacksquare$

Now we show that we can always construct a mapping where equality holds and thus we conclude that the diameter and the maximum range are tightly connected.

Theorem 4. For any connected graph $G$, $r^\text{max}_M(G) = M \cdot (\text{diam}(G) + 1).$
Proof. From the definition of the diameter, there must exist vertices \( u_1 \) and \( u_2 \) such that their distance is equal to \( \text{diam}(G) \). Without loss of generality we set \( r := u_1 \). Now let us define the mapping \( f : V(G) \rightarrow \mathbb{Z} \) so that for every \( v \in V \) we have \( f(v) := M \cdot d(r, v) \).

We see that \( f(r) = 0 \), and \( f(u_2) = M \cdot d(r, u_2) \) so the image of the shortest path connecting \( u_1 \) and \( u_2 \) has the size \( \text{diam}(G) + 1 \). On the top of that, for every \( uv \in E(G) \), \( |f(u) - f(v)| \leq M \), otherwise we would get a contradiction with the definition of the distance. Thus \( f \) is an \( M \)-Lipschitz mapping and its range has to be at least \( M \cdot (\text{diam}(G) + 1) \). Combining this with Lemma 2, we get the claim we wanted to prove. \( \square \)

### 3.2 The case of strong Lipschitz mappings

By Lemma 1 we showed that strong Lipschitz mappings can exist on bipartite graphs only. We will now extend Theorem 4.

**Theorem 5.** For any bipartite connected graph \( G \), \( r_{\text{max}}^M(G) = M \cdot (\text{diam}(G) + 1) \).

**Proof.** We can take Lipschitz mapping \( f \) as in Theorem 4. However, we have to check if it is a strong \( M \)-Lipschitz mapping.

Suppose that \( f \) is not a strong \( M \)-Lipschitz mapping. That means that there exist two vertices \( a, b \in V(G) \) such that \( ab \in E(G) \) and \( f(a) = f(b) \). Furthermore, from the definition of \( f \), \( d(r, a) = d(r, b) \). Define \( l := d(r, a) \).

From the definition of the distance, we get that there exist an \((r, a)\)-path and \((r, b)\)-path, both of length \( l \). Since \( G \) is bipartite, parts to which vertices belong have to alternate along the \((r, a)\)-path and along the \((r, b)\)-path as well. Additionally, parts to which \( a \) and \( b \) belong are determined by the parity of their distance from root. But that means that \( a \) and \( b \) belong to the same part. Since they are neighbors, we get a contradiction.

Finally, we note that the previous argument works also in the case of \( r \) being either the vertex \( a \) or \( b \). \( \square \)

### 3.3 Application

We will apply our results to prove Theorem 6 which was left unproven. We will first need the so called “cherry lemma”.

**Theorem 6.** For every connected graph \( G = (V, E) \) and for every two vertices \( a, b \in V \) such that \( ab \not\in E \), holds that \( \mathcal{L}_1(G) \geq \mathcal{L}_1(G \cup \{a, b\}) \).

**Lemma 3 (Cherry lemma).** A graph \( G \) is a disjoint union of complete graphs if and only if it does not contain \( K_{1,2} \) as an induced subgraph.

Now we can prove Theorem 6.
Proof (Proof of Theorem 6). The graph $G$ cannot be a complete graph. Therefore, by Lemma 3 induced $K_{1,2}$ exists in $G$. Let vertices $a$ and $b$ from the statement of Theorem 6 be the two non-adjacent vertices of induced $K_{1,2}$.

We see that $G$ has the diameter at least 2, since $a$ and $b$ are in distance 2. Let us root $G$ in $a$ for auxiliary reasons.

By the construction of 1-Lipschitz mapping from Theorem 4 there must exist a mapping $f$ with $f(b) = d(a, b) = 2$.

Clearly, $\mathcal{L}_1(G \cup ab) \subseteq \mathcal{L}_1(G)$. However, $f$ cannot be a 1-Lipschitz mapping of $G \cup ab$ rooted in $a$. That implies

$$|\mathcal{L}_1(G \cup ab)| \leq |\mathcal{L}_1(G)| - 1,$$

and we are done. $\square$

Theorem further implies the following.

**Corollary 1.** Among connected graphs of order $n$, trees have the maximum number of 1-Lipschitz mappings and a complete graph $K_n$ has the minimum number of 1-Lipschitz mappings.

### 3.4 Algorithmic aspects

Let us consider the following algorithmic problems – $M$-MaxRange and $M$-Strong-MaxRange.

**Problem:** Maximum range problem – $M$-MaxRange  
**Input:** A connected graph $G$.  
**Question:** What is the maximum range of $M$-Lipschitz mapping of $G$, i.e. $r^\text{Max}_M(G)$?

**Problem:** Strong maximum range problem – $M$-Strong-MaxRange  
**Input:** A connected bipartite graph $G$.  
**Question:** What is the maximum range of strong $M$-Lipschitz mapping of $G$, i.e. $r^\text{Strong}_M(G)$?

Because of Theorem 4 we can use the existing algorithms for finding graph diameter and distance in graphs for both of these problems. The following table is a quick survey of them. In it we denote by $V$ and $E$ the order and the size of the input graph, respectively.

Observe that these algorithms are suitable for general graphs. We can achieve a better complexity for some classes. Take for example trees for which we can compute diameter by a linear-time algorithm using one clever depth-first search traversal.
| Name of algorithm | Complexity | Source |
|-------------------|------------|--------|
| Floyd-Warshall algorithm | $O(V^3)$ | [9] |
| Johnson’s algorithm | $O(V^2 \cdot \log V + V E)$ with Fibonacci heaps for Dijkstra subroutine | [10] |
| Seidel’s algorithm | $O(V^{2.376} \cdot \log V)$ | [23] |

Table 1: Summary of selected algorithms for graph diameter.

Connection to the surjective homomorphism problem

We saw that 1-STRONG-MAXRANGE is easily solvable by algorithms for graph diameter. However, we would like to show a broader context of this problem by pointing out a connection with problems of finding surjective graph homomorphisms. We need to formulate SURJECTIVE COLORING problem first. We note, that by surjective homomorphisms we mean vertex-surjective homomorphism, not the edge-surjective one.

**Problem:** SURJECTIVE COLORING

**Input:** Graphs $G$ and $H$.

**Question:** Does there exist a graph homomorphism of $G$ to $H$ that is surjective?

The graph $G$ is called the *guest graph* and the graph $H$ the *host graph*. If all the guest graphs are from a graph class $\mathcal{G}$ and all the host graphs are from a graph class $\mathcal{H}$, we speak about the SURJECTIVE ($\mathcal{G}, \mathcal{H}$)-HOMOMORPHISM problem.

Golovach et al. [12] proved the following theorem.

**Theorem 7.** [12, Proposition 1] The SURJECTIVE ($\mathcal{G}, \mathcal{H}$)-HOMOMORPHISM problem can be solved in polynomial time in the following two cases:

1. $\mathcal{G}$ is the class of complete graphs and $\mathcal{H}$ is the class of all graphs;
2. $\mathcal{G}$ is the class of all graphs and $\mathcal{H}$ is the class of paths.

We get the following corollary for 1-STRONG-MAXRANGE. Observe that we need to binary search for suitable $n^*$ such that a vertex-surjective homomorphism to a host graph $H$ isomorphic to $P_{n^*}$ exists.

**Corollary 2.** The problem 1-STRONG-MAXRANGE can be solved in polynomial time.

**Proof.** We have a connected bipartite graph $G$ as the input graph. Let us denote its order $n$. We need to binary search for suitable $n^*$. We start with the closed interval $I = [1, n]$ and we choose $n^* := \left\lceil \frac{n}{2} \right\rceil$. Start the instance of SURJECTIVE COLORING with $G$ and $P_{n^*}$. Depending on the result, we will continue to binary search in some corresponding subinterval and set $n^*$ to a new value, i.e. we binary search for the maximum $n^* \in I$ such that $G$ admits a surjective homomorphism to $P_{n^*}$. At the end of the algorithm, we will output the resulting $n^*$ as the answer.

$\square$
For more information on surjective homomorphism problems, we refer to [4,13].

4 Extending partial Lipschitz mappings

While studying Lipschitz mappings we came up with an algorithmic problem which falls into widely studied paradigm of a partial structure extension. We give two examples of such problems to show a broader context.

4.1 Related problems

Precoloring extension The following problem was introduced in the series of papers [3,14,15].

**Problem:** Precoloring Extension

**Input:** An integer $k \geq 2$, a graph $G = (V, E)$ with $|V| \geq k$, a vertex subset $W \subseteq V$, and a proper $k$-coloring of $G_W$.

**Question:** Can this $k$-coloring be extended to a proper $k$-coloring of the whole graph $G$?

To current date, more than twenty papers on the precoloring extension problem were published. No up-to-date survey is available, but Daniel Marx gathers an unofficial list of relevant papers on his webpage: [http://www.cs.bme.hu/~dmarx/prext.php](http://www.cs.bme.hu/~dmarx/prext.php).

The partial representation extension problem. The reader surely knows a planar drawing of graph. A particular drawing of the underlying graph can be seen as one of the possible representations. Studying the representations of various graph classes is a wide area of graph theory and we refer reader to the comprehensive monograph of Spinrad [24]. One can ask for a given graph $G$ and some partial representation $R'$ of $G$ if it can be extended to some full representation $R$ of $G$ such that $R' \subseteq R$. This problem was studied for various graph classes, for example intersection graph classes [18,19,17] or planar graphs [5]. For a presentation of state of the art in partial representation extension problems, consult the PhD thesis of Klavík [20].

4.2 Definition of our problem

We will define two similar problems in the setting of integer homomorphisms.

**Problem:** Partial $M$-Lipschitz mapping extension - $M$-LipExt

**Input:** A connected graph $G = (V, E)$, a subset $V' \subseteq V$ with a function $f' : V' \to \mathbb{Z}$.

**Question:** Does there exist an $M$-Lipschitz mapping $f$ of $G$ such that $f' \subseteq f$?
PROBLEM: Partial strong $M$-Lipschitz mapping extension - STRONG $M$-LipExt

INPUT: A connected bipartite graph $G = (V, E)$, a subset $V' \subseteq V$ with a function $f' : V' \rightarrow \mathbb{Z}$.

QUESTION: Does there exist a strong $M$-Lipschitz mapping $f$ of $G$ such that $f' \subseteq f$?

If the answer for a given instance of $M$-LipExt (or STRONG $M$-LipExt) is YES, we say that $f'$ is extendable for the given $G$ and the given type of problem. We often say only that $f'$ is extendable when it is clear from the context which problem we are trying to solve.

See Figure 2 for an initial example. Clearly, this mapping cannot be extended to a 1-Lipschitz mapping but it can be extended to an $L$-Lipschitz mapping for every $L \geq 2$.

Fig. 2: An example of a partial mapping with three prescribed vertices.

4.3 Partial strong $M$-Lipschitz mappings

We will show that STRONG $M$-LipExt can be solved by a special linear program (LP) with the property that all its feasible solution are integral and a feasible solution exists if and only if $f'$ is extendable.

Theorem 8. STRONG $M$-LipExt is solvable in polynomial time.

Proof. We prove the theorem by constructing a linear program for the given instance with polynomially many inequalities of polynomial size. We will further show that we are interested in feasible solutions only and that if a feasible solution exists, it is always integral. As we know, LP can be solved in polynomial time with respect to the size of the program, as is explained e.g. in [22]. We note that it might be possible to simplify the following theorem by employing total unimodularity but we were unable to do that.
**Step 1: Checking the mapping \( f' \).**

For the images of mapping \( f' \), we can easily check the necessary conditions on the difference of their images in quadratic time.

Note that if \( V' = V(G) \) then it suffices to check the differences on every edge, check that at least one vertex maps to zero, and we are done.

So assume that \( V(G) \setminus V' \) is nonempty and for all \( u, v \in V' \), if \( uv \in E(G) \) then \( |f'(u) - f'(v)| = M \).

**Step 2: Creating the LP.**

We denote by \( N_G(v) \) the set of vertices adjacent to \( v \) in \( G \). We have the variables \( y_v \) for every vertex \( v \in V(G) \setminus V' \). We need feasible solutions only, so the objective function is of no interest to us.

\[
\begin{align*}
\text{minimize} & \quad 0 \\
\text{subject to} & \quad |f'(i) - y_j| = M \quad \forall i, j : i \in V', j \in N_G(i) \cap (V(G) \setminus V') \\
& \quad |y_k - y_l| = M \quad \forall k, l \in V(G) \setminus V', k \neq l
\end{align*}
\]

However this is not a linear program yet. Absolute values violate the definition of LP. However, these can be changed to linear inequalities by the standard trick of adding two additional variables for each of inequalities (see for example [22]).

**Step 3: Enforcing the existence of a root.**

We still have a problem. We need to ensure that at least one vertex is mapped to zero, i.e. that we have some root. If there is no such vertex in \( V' \), i.e. \( f'^{-1}(0) = \emptyset \), then, unfortunately, we do not have a strong Lipschitz mapping yet.

However, this can be fixed rather easily. We iterate over the vertices \( r' \in V(G) \setminus V' \) and extend \( f' \) in the following way:

\[ f' := f' \cup (r', 0). \]

Then we continue building our LP as previously. If there is no feasible solution, remove \((r', 0)\) from \( f' \) and try to add some other vertex \( r'' \in V(G) \setminus V \).

**Step 4: Complexity.**

We will compute our LP program at most \( O(V(G)) \) times to ensure that we have \( G \) rooted. Every LP program consists of polynomially many inequalities, because we had one \( O(E(G)) \) inequalities with size bounded by a constant and by striping the absolute values away, we increased the number of inequalities only by some constant factor.

Thus the size of our program is polynomial in \( V(G) \) and \( E(G) \) and we indeed have a polynomial running time of our algorithm due to the polynomiality of linear programming.

**Step 5: Correctness.**

We claim that every output of our program is integral. Every \( f'' \) for which we build our LP has some vertex mapped to zero. Inequalities ensure that neighboring vertices get values either \(+M\) or \(-M\); an integer value. We can proceed inductively and integrality follows.

To finish, we need to prove the following. For a given instance of STRONG \( M\)-LIPEXT, our algorithm outputs a complete \( M\)-Lipschitz mapping \( f \), such that \( f' \subseteq f \), if and only if \( f' \) is extendable.
Every output of our program has to satisfy that some vertex from \( V(G) \) is mapped to zero and it also has to satisfy that for every \( uv \in E(G) \), \( |f(u) - f(v)| = M \). Thus it an \( M \)-Lipschitz mapping of \( G \). That finishes the proof of the if part.

If for every assignment of integers to the program variables there is no feasible solution, then some of the inequalities must be violated. Inequalities of our LP are precisely the conditions from the definition of strong \( M \)-Lipschitz mapping and therefore \( f' \) is not extendable.

\[ \square \]

4.4 Partial \( M \)-Lipschitz mappings

A simple generalization of the previous algorithm does not work for \( M \)-Lipschitz mappings, or at least we were unable to prove it. However, we came up with a different polynomial algorithm for the general case of \( M \text{-ParExt} \). This subsection will start with the special case of trees and only then we will show the result on general graphs.

Trees

We will use Algorithm 1 for solving \( M \text{-ParExt} \) on trees.

We will now prove the correctness and complexity of this algorithm.

**Lemma 4 (Correctness).** Algorithm 1 is correct. It finds an \( M \)-Lipschitz mapping \( f \) that extends \( f' \) if and only if \( f' \) is extendable.

**Proof.** We will write \( V \) for \( V(G) \) and we will denote the iterations of code between the lines 4 and 10 DFS phases and the code executed between the lines 19 and 22 the BFS phase.

Suppose that the algorithm returns a mapping \( f \). We claim that it is an \( M \)-Lipschitz mapping extending \( f' \). Obviously, there exists a vertex mapped to zero under \( f \) – the vertex \( r \). Furthermore, the condition

\[
|f(u) - f(v)| \leq M, \forall uv \in E(G)
\]

holds, otherwise the algorithm would stop on Line 19. Finally, we observe that for every \( v' \in V' \), interval \( P(v') \) is equal to \([f'(v'), f'(v')]\) at the end of the algorithm so \( f \) extends \( f' \). That finishes the only if part of the equivalence.

Now let us prove the if part. We will prove that if the algorithm does not find an \( M \)-Lipschitz mapping \( f \) extending \( f' \), then \( f' \) is not extendable.

Algorithm can stop and fail to find such \( f \) exactly from the following reasons:

1. **Algorithm could not find a candidate for the root. (Line 13)**
   If at the end of the algorithm for every vertex \( v \in V \), \( 0 \notin P(v) \), then for every \( v \in V \) exists some vertex \( v' \in V' \) such that \( |f(v')| > M \cdot d(v, v') \). Clearly, \( f' \) is not extendable.

2. **There exists \( v \in V \) such that \( P(v) = \emptyset \). (Line 17)**
   If such \( v \) exists, then it implies that there exist two vertices \( c, d \in V' \) such that the intersection \( I = [f'(c) - M \cdot (c, v), f'(c) + M \cdot (c, v)] \cap [f'(d) - M \cdot (d, v), f'(d) + M \cdot (d, v)] \)
Algorithm 1 Wave algorithm for $M$-PARExt on trees.

Require: A tree graph $G$, a vertex set $V' \subseteq V(G)$, and a partial $M$-Lipschitz mapping $f' : V' \to \mathbb{Z}$.

1: Check if $|f'(v) - f'(u)| \leq M$ for all $u, v \in V'$. If not, $f'$ cannot be extended.
2: Set $P(v) := [f'(v), f'(v)]$ for every $v \in V'$.
3: Set $P(v) := [-\infty, \infty]$ for every $v \in V(G) \setminus V'$.

4: for every $v'$ in $V'$ do
5: Start the DFS on $G$ from $v'$.
6: In DFS, whenever you process vertex $v$ with $P(v) = [P(v), P(v)]$, do the following:
7: for every $w \in N_G(v)$ do
8: $P(w) := [P(v) - M, P(v) + M] \cap P(w)$.
9: end for
10: end for
11: Find $r \in V(G)$ such that $0 \in P(v)$.
12: if no such $r$ then
13: return The mapping $f'$ cannot be extended.
14: end if
15: Set $f(r) := 0$.
16: if $P(v) = \emptyset$ for some $v \in V(G)$ then
17: return The mapping $f'$ cannot be extended.
18: end if
19: Launch the BFS from $r$ and for every visited vertex $v \neq r$, set $f(v)$ so that for parent vertex $p$, $f(v) \in [f(p) - M, f(p) + M]$ holds.
20: if the previous BFS could not be completed then
21: return The mapping $f'$ cannot be extended.
22: end if
23: return The mapping $f : V(G) \to \mathbb{Z}$.

$(d, v), f'(d) + M \cdot (d, v)$ is empty. However, $I$ is exactly the set of all possible images that we can assign to $v$ if $c$ is set to $f'(c)$ and $d$ is set to $f'(d)$. We conclude that $f'$ is not extendable.

3. Algorithm could not complete the BFS phase. (Line 21)

We will actually show that this case is not possible since the only possibility how 3) can happen is that some final interval $P(v)$ for some $v \in V$ is empty and the algorithm will halt even before the BFS phase can start (more precisely, the algorithm will stop at line 17).

Assume that all intervals $P(v)$ are nonempty. Consider an edge $xy \in E(G)$. Assume further without loss of generality that in the last DFS phase (line 6), $x$ was processed before $y$. Consider intervals $P'(x), P'(y)$ defined as the intervals $P(x), P(y)$, respectively, before the last DFS phase. Clearly, when $x$ was processed in the last DFS phase, $P'(y) \cap [P'(x) - M, P'(x) + M]$ was
set to a nonempty interval and therefore,
\[ \forall i \in P(x), \exists j \in P(y) : |i - j| \leq M. \]

And conversely,
\[ \forall j \in P(y), \exists i \in P(x) : |i - j| \leq M. \]

We conclude that the case 3) cannot occur.

This proves the if part and we are done. \( \square \)

Lemma 5 (Complexity). Algorithm \( \square \) is quadratic, i.e., its time complexity is \( O(n^2) \), given that \( n \) is the number of vertices of the input graph.

Proof. We are running \( O(|V(G)|) \) times DFS on \( G \) plus we perform a constant number of linear traversals of data structure for \( G \). That concludes that the algorithm runs in quadratic time. \( \square \)

From these two lemmas we conclude the following theorem.

Theorem 9. \( \text{M-ParExt for trees is solvable in quadratic time and linear space.} \)

General graphs

The main result of this section is to prove polynomiality in the general case.

Theorem 10. The problem \( \text{M-ParExt} \) is solvable in polynomial time on general graphs.

It will be useful to define a new property for integer functions on vertex sets.

Definition 7 (M-reachability). We call a mapping \( f : V' \to \mathbb{Z} \) of graph \( G \), with \( V' \subseteq V(G) \), \( M \)-reachable if every pair of vertices \( u, v \in V' \) satisfies
\[ |f(u) - f(v)| \leq M \cdot d(u, v), \]

Definition 8. We call a mapping \( f : V' \to \mathbb{Z} \) of graph \( G \), with \( V' \subseteq V(G) \), zero-containing if \( f^{-1}(0) \neq \emptyset \).

We can now state and prove the full characterization of extendable situations.

Theorem 11. For a graph \( G = (V, E) \), subset \( V' \subseteq V \), and a partial mapping \( f' : V' \to \mathbb{Z} \), the following statements are equivalent:

1. The mapping \( f' \) is extendable to a \( M \)-Lipschitz mapping.
2. One of the following holds:
   (a) The mapping \( f' \) is \( M \)-reachable and zero-containing.
   (b) There exist \( r \in V \setminus V' \) with \( f'' \) defined as
   \[ f'' := f' \cup (r, 0), \]
   such that \( f'' \) is \( M \)-reachable.
Proof. (1) \(\Rightarrow\) (2): We will prove it by contradiction. For brevity, we will handle the (a) case, (b) case is similar. Choose a pair of vertices \((u, v)\) certifying that \(f'\) is not \(M\)-reachable such that the distance between \(u\) and \(v\) is the smallest. There exists a path between \(u\) and \(v\) with free vertices. Clearly, for every completion of this path the conditions for \(M\)-Lipschitz mapping will not be satisfied. We have a contradiction.

(2) \(\Rightarrow\) (1): Again, we will only prove the (a) case, (b) can be done in analogous way. We will show that we can extend \(f'\) mapping by one vertex and preserve \(M\)-reachability. By applying this inductively we will prove that (2) implies (1).

Choose a vertex \(a\) that is adjacent to some vertex \(b\) such that \(f'(b)\) is defined and \(f'(a)\) is not defined. For every vertex \(c \in f'(V')\), the vertex \(b\) is reachable. Thus we can always find a number for \(a\) such that \(c\) will be reachable from \(a\). For every \(c\) we can define interval \(I_c\) containing all possible values for \(a\) such that \(c\) is reachable from \(a\). This is clearly a closed connected interval in \(\mathbb{Z}\). Furthermore, the set system \(\{I_c | \forall c \in f'(V')\}\) has the Helly property. Finally, for every two \(c_1, c_2 \in f'(V')\) there is a nonempty intersection. Otherwise we would get contradiction with \(M\)-reachability of \(f'\). Thus we can pick a suitable \(k \in \bigcap_{c \in f'(V')} I_c\) and extend \(f'\) by setting \(f' := f' \cup \{(a, k)\}\). \(\square\)

Now we can finally prove polynomiality.

Proof (of Theorem 11). We can check in linear time if the given mapping is zero-containing. If it is, we can use Floyd-Warshal algorithm to check \(M\)-reachability. If the given mapping is not zero-containing, we try one free vertex after another, set it to zero and again we check \(M\)-reachability. \(\square\)

The proof of Theorem 11 yields an algorithm for constructing the extended mapping. We omit the details of this construction.

Application

We will apply Theorem 11 to derive yet another proof of Theorem 6.

Proof (of Theorem 6). Use Lemma 8 and let \(a\) and \(b\) be the two non-adjacent vertices in induced \(K_{1,2}\). Let us define the mapping \(f' : a, b \rightarrow \mathbb{Z}\) with:

\[
f'(a) = 0 \quad \text{and} \quad f'(b) = 2.
\]

By Theorem 11 \(f'\) is \(M\)-reachable and zero-containing and thus extendable. However, if we add the edge \(ab\) into \(G\), \(f'\) becomes non-extendable. Combined with the observation that Lipschitz mappings of \(G\) with deleted edge are a subset of Lipschitz mappings of \(G\), we get that there is at least one extra mapping if \(ab\) is not in the edge set of \(G\) and the claim follows. \(\square\)
5 Concluding remarks

We studied algorithmic aspects of Lipschitz mappings of graphs. We showed that while the problem of finding the average range is \#P-complete, the problem of finding the maximum range and extending partial Lipschitz mapping are both solvable in polynomial time.

6 Acknowledgments

This research was supported by the Charles University Grant Agency, project GA UK 1158216.

References

1. Benjamini, I., H"aggstr"om, O., and Mossel, E. On random graph homomorphisms into $\mathbb{Z}$. Journal of Combinatorial Theory, Series B 78, 1 (2000), 86–114.
2. Benjamini, I., and Schechtman, G. Upper bounds on the height difference of the Gaussian random field and the range of random graph homomorphisms into $\mathbb{Z}$. Random Structures and Algorithms 17, 1 (2000), 20–25.
3. Biro, M., Hujter, M., and Tuza, Z. Precoloring extension I: Interval graphs. Discrete Mathematics 100, 1-3 (1992), 267–279.
4. Bodirsky, M., Kára, J., and Martin, B. The complexity of surjective homomorphism problems—a survey. Discrete Applied Mathematics 160, 12 (2012), 1680–1690.
5. Chaplick, S., Dorbec, P., Kratochvíl, J., Montassier, M., and Stacho, J. Contact representations of planar graphs: extending a partial representation is hard. In International Workshop on Graph-Theoretic Concepts in Computer Science (2014), Springer, pp. 139–151.
6. Cohen, E., Perkins, W., and Tetali, P. On the widom–rowlinson occupancy fraction in regular graphs. Combinatorics, Probability and Computing 26, 2 (2017), 183–194.
7. Diestel, R. Graph theory (Graduate texts in mathematics; 173). Springer-Verlag Berlin and Heidelberg GmbH, 2000.
8. Erschler, A. Random mappings of scaled graphs. Probability theory and related fields 144, 3-4 (2009), 543–579.
9. Floyd, R. W. Algorithm 97: shortest path. Communications of the ACM 5, 6 (1962), 345.
10. Focke, J., Goldberg, L. A., and Zivny, S. The Complexity of Counting Surjective Homomorphisms and Compactions. 1–35.
11. Godsil, C., and Royle, G. F. Algebraic graph theory, vol. 207. Springer Science & Business Media, 2013.
12. Golovach, P. A., Lidický, B., Martin, B., and Paulusma, D. Finding vertex-surjective graph homomorphisms. Acta Informatica 49, 6 (2012), 381–394.
13. Hell, P., and Nešetřil, J. Graphs and homomorphisms. Oxford Lecture Series in Mathematics and its Applications (2004).
14. Hujter, M., and Tuza, Z. Precoloring extension II: Graph classes related to bipartite graphs. Acta Mathematica Universitatis Comenianae 62, 1 (1993), 1–11.
15. **Hujter, M., and Tuza, Z.** Precoloring extension III: Classes of perfect graphs. *Combinatorics, Probability and Computing* 5, 1 (1996), 35–56.

16. **Johnson, D. B.** Efficient algorithms for shortest paths in sparse networks. *Journal of the ACM (JACM)* 24, 1 (1977), 1–13.

17. **Klavík, P., Kratochvíl, J., Otachi, Y., Rutter, I., Saitoh, T., Saumell, M., and Vyskočil, T.** Extending partial representations of proper and unit interval graphs. *Algorithmica* 77, 4 (2017), 1071–1104.

18. **Klavík, P., Kratochvíl, J., Otachi, Y., and Saitoh, T.** Extending partial representations of subclasses of chordal graphs. In *Algorithms and Computation, ISAAC 2012* (2012), vol. 7676 of *Lecture Notes in Computer Science*, pp. 444–454.

19. **Klavík, P., Kratochvíl, J., Otachi, Y., Saitoh, T., and Vyskočil, T.** Extending partial representations of interval graphs. *Algorithmica* (2016), 1–23.

20. **Klavík, P.** *Extension Properties of Graphs and Structures*. Charles University, 2017.

21. **Loebl, M., Nešetřil, J., and Reed, B.** A note on random homomorphism from arbitrary graphs to $\mathbb{Z}$. *Discrete mathematics* 273, 1 (2003), 173–181.

22. **Matoušek, J., and Gärtner, B.** *Understanding and using linear programming*. Springer Science & Business Media, 2007.

23. **Seidel, R.** On the all-pairs-shortest-path problem. In *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing* (1992), ACM, pp. 745–749.

24. **Spinrad, J. P.** *Efficient graph representations*. American Mathematical Society, 2003.

25. **Wu, Y., Xu, Z., and Zhu, Y.** Average range of Lipschitz functions on trees. *Moscow Journal of Combinatorics and Number Theory* 1, 6 (2016), 96–116.

26. **Zhao, Y.** Extremal regular graphs: independent sets and graph homomorphisms. *arXiv preprint arXiv:1610.09210* (2016).