




# Efficient Minimization of Peakless Functions on Bounded-degree Graphs

Christoph Sandrock  (✉), Sebastian Lüderssen,  
Maximilian Thiessen , and Thomas Gärtner 

TU Wien, Vienna, Austria  
`christoph.sandrock@tuwien.ac.at`

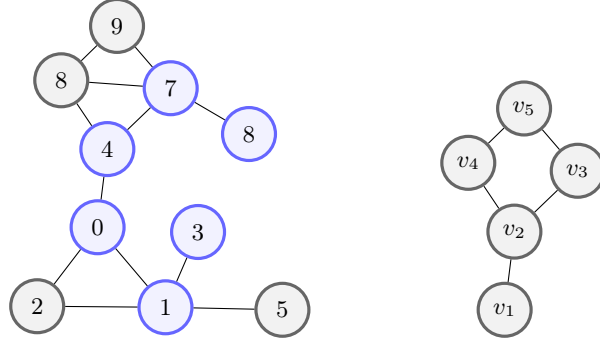
**Abstract.** We study the problem of query-efficiently minimizing peakless functions on graphs. Peakless functions are a notion of convex functions on the nodes of a graph and are defined by having no local “peaks” along any shortest paths. We demonstrate that peakless functions form a rich class and are non-trivial to optimize on chordal graphs. Further, we show that these functions can be minimized using at most  $4\Delta \log n$  queries on graphs with  $n$  nodes and maximum degree  $\Delta$ , where each query reveals the function value of a selected node. We complement this result by providing a nearly matching lower bound on the query complexity of  $\Omega(\frac{\Delta}{\log \Delta} \log n)$ .

**Keywords:** Graph function optimization · Graph convexity.

## 1 Introduction

We study query-efficient optimization of functions defined on the nodes of a graph under convexity assumptions. While the network structure is often readily available, querying function values can be costly. Query-efficient algorithms address this problem by querying only informative nodes. Existing approaches leverage ideas from Bayesian optimization [2, 10, 13, 15] to solve problems like detecting patient zero in infection networks, identifying bottlenecks in infrastructure networks, and analyzing social networks. However, they lack theoretical guarantees regarding their *query complexity*, i.e., the number of function evaluations required to find the minimum of a function. To close this gap, we initiate the theoretical study of query-efficient function optimization on graphs, where queries select one node in the graph and get the function value of this node as an answer.

One of the most fundamental classes of functions in continuous optimization is convex functions. While there are multiple notions of convex functions on graphs, they are usually defined for specific graph classes such as trees or grids, and there is no established notion of convex functions on general graphs. In this work, we study functions that have no *peaks* along shortest paths, i.e., no values that exceed the values of both endpoints of the path. They were introduced for



**Fig. 1. Left:** Example of a peakless function on a chordal graph, where the numbers describe the function values  $f(v)$ . The blue nodes form a shortest path. Hence, the function values must be peakless along this path. **Right:** Example of a non-chordal graph. The walk  $(v_1, v_2, v_3, v_5, v_4, v_2, v_1)$  is a locally-shortest walk but no shortest path. The only totally convex sets are  $\{v_2, v_3, v_4, v_5\}$  and  $\{v_1, v_2, v_3, v_4, v_5\}$ , which implies that  $v_1$  can only be minimal for constant peakless functions on this graph.

graphs as *peakless functions* by Chepoi [6]. These functions generalize strictly quasiconvex functions on graphs [1] and thus are more widely applicable. Furthermore, they inherit important properties like convex sublevel sets. Our main contributions are:

1. We show that peakless functions can be minimized with  $4\Delta \log n$  queries on graphs with  $n$  nodes and maximum degree  $\Delta$ .
2. We show that this query complexity is near-optimal by proving a lower bound of  $\Omega\left(\frac{\Delta}{\log \Delta} \log n\right)$ .

## 2 Preliminaries

Our goal is to find a minimum of a function  $f : V \rightarrow \mathbb{R}$  defined on the nodes of a graph  $G = (V, E)$ . Throughout this paper, we assume that graphs are connected and undirected. We denote by  $n = |V|$  the number of nodes and by  $\Delta$  the maximum degree of  $G$ . For edges  $\{u, v\} \in E$  we write  $u \sim v$ . We denote by  $d(u, v)$  the shortest-path distance of two nodes  $u, v \in V$ . A *walk* is a sequence of nodes  $(w_0, w_1, \dots, w_m)$  with  $w_i \sim w_{i+1}$  for all  $0 \leq i \leq m-1$ . A *locally-shortest walk* is a walk with  $d(w_{i-1}, w_{i+1}) = 2$  for all  $1 \leq i \leq m-1$ . Note that locally-shortest walks can have repeated nodes (as shown in the right graph in Figure 1). A set of nodes  $X \subseteq V$  is *totally convex* if it contains every locally-shortest walk in  $G$  whose endpoints are both in  $X$ .

We want to optimize  $f$  *efficiently*, that is, with as few function evaluations as possible. We call the minimum number of queries required to minimize a fixed but unknown function the *query complexity*. Since obtaining a non-trivial query complexity is impossible for arbitrary functions, we assume some properties of

the function  $f$ . We say that  $f : V \rightarrow \mathbb{R}$  is *peakless* [6] if for all  $u, v, w \in V$  with  $u \sim v \sim w$  and  $d(u, w) = 2$  it holds that  $f(v) \leq \max(f(u), f(w))$ , with equality holding only if  $f(u) = f(w)$ . Equivalently,  $f$  is peakless if  $f(v_i) \leq \max(f(v_0), f(v_m))$  holds for all shortest paths  $(v_0, \dots, v_m)$  and  $1 \leq i \leq m - 1$ , again with equality holding only if  $f(v_0) = f(v_m)$ . The left graph in Figure 1 shows an example of a peakless function. As shown by Chepoi [6], totally convex sets correspond to sublevel sets of peakless functions. More precisely, a set  $X \subseteq V$  is totally convex if and only if it is a sublevel set  $\{v \in V \mid f(v) \leq \alpha\}$  of some peakless function  $f$  with  $\alpha \in \mathbb{R}$ . A graph is *chordal* if every cycle of length greater than three has a *chord*, i.e., an edge connecting two non-consecutive nodes.

A related problem is *graph search with edge queries*. Given a graph, the goal is to find a target node  $t \in V$ . The algorithm can ask edge queries  $\{v_1, v_2\} \in E$ , and an oracle returns the node  $u \in \{v_1, v_2\}$  which is closer to  $t$  (with ties broken arbitrarily). We also say that  $u$  is the oracle answer in this case. This problem was first studied for trees [3, 8]. Later Dereniowski et al. [7] showed that  $2\Delta \log n$  queries are sufficient for general graphs.<sup>1</sup> By establishing a connection between peakless optimization and graph search, we can use their algorithm to minimize peakless functions. The algorithm uses the following idea: First, initialize the weight  $\mu(v)$  of all nodes  $v \in V$  with 1. Then, iteratively query an edge

$$\{u, v\} \in \arg \min_{\{u, v\} \in E} \sum_{w \in V} \mu(w) \min(d(u, w), d(v, w))$$

with minimum summed distance to all nodes, weighted by  $\mu$ . After receiving the node  $u$  closer to the target as the oracle answer, decrease the weight of all nodes that are closer to the other endpoint  $v$  by a fixed fraction. When a node has more than a  $\frac{1}{\Delta+1}$  fraction of the total weight, query all incident edges to check whether this node is the target.

### 3 Optimizing Peakless Functions

We start this section by showing that peakless functions form a diverse class of functions on chordal graphs and are non-trivial to optimize. In the second part, we show that peakless functions can be optimized efficiently on general bounded-degree graphs.

#### 3.1 Peakless Functions on Chordal Graphs

Since constant functions on any graph are peakless, every graph admits peakless functions. However, some graphs only admit constant peakless functions. This is because all sublevel sets must be totally convex. For example, on cycle graphs

<sup>1</sup> Dereniowski et al. [7] consider various noisy settings. In their Theorem 7, they show that with linearly bounded error and precision parameter  $\varepsilon \in (0, 1]$ , the target can be found with  $2\varepsilon^{-2}\Delta \log n$  queries. Setting  $\varepsilon = 1$  for the noiseless case yields the bound we use.

with more than three nodes, the only non-empty totally convex set is  $V$  (the set of all nodes). Hence, these graphs only admit constant peakless functions. Moreover, many other graphs (such as the right graph in Figure 1) only admit peakless functions that can be minimized trivially without any queries. The following lemma, which is similar to a fact observed by Chepoi [6], shows that this is not the case for chordal graphs.

**Lemma 1.** *Let  $G = (V, E)$  be chordal. Then, for each  $v \in V$  there exists a peakless function  $f : V \rightarrow \mathbb{R}$  with unique minimum  $v$ .*

*Proof.* To prove the lemma, we show that every node  $v \in V$  in a chordal graph forms a totally convex set  $\{v\}$ . Then,  $\{v\}$  is a sublevel set of some peakless function, implying that  $v$  is the unique minimum of this function.

Assume for contradiction that there is a node  $v \in V$  such that  $\{v\}$  is not totally convex. Then there exists a locally-shortest  $v$ - $v$ -walk  $W$  in  $G$ . Let  $c_1$  be the first repeated node on this walk. Since  $v$  is repeated, such a node must exist. Let  $C = (v_1, \dots, v_k, v_1)$  be the sub-walk of  $W$  between the first and second visit of  $v_1$ . Since  $v_1$  is the first repeated node,  $C$  is a cycle. We have  $k \geq 4$  because  $W$  is a locally-shortest walk. We now show that one of the shortcut edges  $\{v_i, v_{i+2}\}$  exists for some  $1 \leq i \leq k-2$ , contradicting the fact that  $W$  is locally-shortest.

*Claim.* For any cycle  $C' = (u_1, u_2, \dots, u_s, u_1)$  of length  $s \geq 4$  in a chordal graph  $G = (V, E)$  there is some  $1 \leq i \leq s-2$  such that  $\{u_i, u_{i+2}\} \in E$ .

*Proof of Claim.* If  $s = 4$ , we know that either  $\{u_1, u_3\} \in E$  or  $\{u_2, u_4\} \in E$  since  $G$  is chordal. Both cases imply the claim. Now, assume the claim holds for all cycles of length  $r < s$ . Since  $s \geq 4$ , there exists a chord  $\{u_a, u_b\} \in E$  in  $C$  for some  $1 \leq a, b \leq s$  with  $b \geq a+2$  and  $\{u_a, u_b\} \neq \{u_1, u_s\}$ . This yields a cycle  $C'' = (u_a, u_{a+1}, \dots, u_b, u_a)$  of size at most  $s-1$ . If  $C''$  is of length 3, we have  $a+2 = b$ , proving the claim. Otherwise, by induction, we know that one of the edges  $\{u_i, u_{i+2}\}$  exists for some  $a \leq i \leq b-2$ . This proves the existence of a shortcut edge in  $C'$ .  $\square$

The lemma implies that all chordal graphs (with more than one node) have interesting peakless functions since they cannot be optimized without queries. For a tighter characterization of peakless functions on chordal graphs, we can observe that peakless functions are constant along any induced cycle with more than three nodes. This is because an induced cycle with at least four nodes is a locally-shortest walk, and thereby every totally convex set containing one of the nodes must contain all nodes in the cycle. Since every sublevel set of a peakless function is totally convex, this implies that there cannot be any peakless function with a sublevel set that only contains some nodes of this cycle. This observation together with Lemma 1 gives the following corollary:

**Corollary 1.** *The following statements are equivalent for any graph  $G = (V, E)$ :*

1.  $G$  is chordal.
2. For each  $v \in V$  there exists a peakless function  $f : V \rightarrow \mathbb{R}$  with unique minimum  $v$ .

### 3.2 Query Complexity of Peakless Functions

In this section, we show an upper and lower bound on the query complexity of peakless functions. We start by showing that we can use graph search algorithms to minimize peakless functions in Theorem 1. Then, we show a near-tight lower bound in Theorem 2.

**Theorem 1.** *Peakless functions on graphs can be minimized with  $4\Delta \log n$  queries.*

*Proof.* We show how we can reduce the optimization problem to a graph search problem. The main difference is that the former uses edge queries with function values as answers, whereas the latter uses edge queries with a direction toward a target as an answer. We can simulate edge queries  $\{u, v\}$  by evaluating  $f$  at both adjacent nodes  $u$  and  $v$ . Let  $f$  be peakless. Then, it holds for adjacent nodes  $u, v$  and all minimizers  $t \in \arg \min_{w \in V} f(w)$  that  $f(u) > f(v)$  implies there cannot exist any shortest  $v - t$  path via  $u$  (because  $u$  would be a peak), and hence  $d(u, t) \geq d(v, t)$ . For  $f(u) = f(v)$ , the same argument yields  $d(u, t) = d(v, t)$ . This means that we can interpret  $f(u) > f(v)$  as an edge oracle answer  $v$ , and we can interpret  $f(u) = f(v)$  arbitrarily as an answer  $v$  or  $u$ . If we run any graph search algorithm with this simulation, the found node is a minimum of  $f$ . Since every query requires two function evaluations, the query complexity is twice the query complexity of the graph search algorithm. The algorithm proposed by Dereniowski [7] needs  $2\Delta \log n$  queries. This implies that we can optimize peakless functions with  $4\Delta \log n$  function evaluations.  $\square$

Next, we show that this strategy is near-optimal:

**Theorem 2.** *For arbitrarily large  $n, \Delta \in \mathbb{N}$  there exists a graph  $G$  with  $n$  nodes and degree  $\Delta$  such that every optimization algorithm needs at least  $\frac{\Delta}{\log \Delta} \log n$  queries to find a minimum of a peakless function on  $G$ .*

To prove the theorem, we construct a tree (similar to Ben-Asher and Archi [3] for the graph search setting) where each strategy requires  $\Omega\left(\frac{\Delta}{\log \Delta} \log n\right)$  queries. We prove this by induction on the height of the tree. For simplicity, we show that the bound even holds for the easier problem where the value of the root node is known.

**Lemma 2.** *Let  $G$  be a complete  $k$ -ary tree of height 1 (i.e., a star graph with  $k + 1$  nodes). Assume that the value  $f(r)$  of the root node  $r$  is known. Then, every strategy must query all  $k$  leaf nodes to find the minimum in the worst case.*

*Proof.* Assume that the root node  $r$  and all leaf nodes except for one node  $u$  are queried. In the worst case, we have  $f(r) < f(v)$  for all known leaf nodes  $v$ . Then our only information about  $u$  is that  $f(r) < \max(f(u), f(v))$  for all other leaves  $v$ . Since  $f(r) < f(v)$  holds for all leaves  $v \neq u$ , the value  $f(u)$  can be arbitrary, and both  $u$  and  $r$  could be minimizers.  $\square$

After showing the base case of our induction, we now continue with the induction step. For simplicity, we again assume the easier problem where the value of the root node is known.

**Lemma 3.** *Let  $G$  be a complete  $k$ -ary tree of height  $h + 1$  for some  $h \geq 1$ ,  $k \geq 2$ . Assume that the value  $f(r)$  of the root node  $r$  is known. Then, the query complexity of every strategy is at least  $k - 1$  plus the query complexity on a complete  $k$ -ary tree of height  $h$  with known root value.*

*Proof.* Assume that  $f$  has a unique minimum attained in one leaf node. Since all subtrees adjacent to the root node without a queried node are indistinguishable for an algorithm, we can assume, as a worst case, that the first  $k - 1$  queries are all in wrong subtrees, i.e., no query is in the subtree  $T$  containing the minimum. Then, all that we know is that the target node is either the root node or within the subtree  $T$ . This situation is more difficult than a tree of reduced height since we can transform it into this case by additionally querying  $r'$ . Hence, we conclude that no algorithm can have a query complexity less than  $k - 1$  plus the query complexity of the tree with reduced height.  $\square$

*Proof of Theorem 2.* Consider a complete  $k$ -ary tree with  $n$  nodes. Then the height is bounded as follows:

$$\begin{aligned} h &\geq \lceil \log_k((k-1)n + 1) - 1 \rceil = \log_k((k-1)n) - 1 \\ &\geq \log_k(k-1) + \log_k(n) - 1 \geq \log_k(n) - 1. \end{aligned}$$

Using the previous lemmas, we get  $h(k-1) + k \geq (k-1)\log_k(n) - 1 + k$  as a lower bound on the number of queries. Since the maximum degree is  $\Delta = k + 1$ , we get  $(\Delta - 2)\log_{\Delta-1}(n) - 1 + \Delta - 1 = \Omega\left(\frac{\Delta}{\log \Delta} \log n\right)$  as a lower bound.  $\square$

## 4 Related Work

The related work falls into two main categories: Theoretical concepts for convex functions on graphs and empirical methods for optimizing graph functions. The two fields are largely disconnected since the former does not address optimization methods, while the latter lacks theoretical guarantees on the query complexity.

*Discrete convex functions.* Several authors [5, 11, 12, 16] proposed different notions of discrete convex functions on the integer grid  $\mathbb{Z}^d$ . While these notions can be transferred to (infinite) grid graphs, they do not directly generalize to other graph classes. Hirai [9] defines discrete L-convex functions on certain restricted graph classes, which they call tree-grids. The closest notion of convexity to peakless functions is that of (strictly) quasiconvex functions [1, 4, 14, 16]. They were first defined on integer intervals as follows [4, 16]: A function  $f : X \rightarrow \mathbb{R}$  defined on an integer interval  $X \subset \mathbb{Z}$  is quasi-convex if for all  $x, y \in X$ ,  $x \neq y$  it holds  $f(z) \leq \max(f(x), f(y))$  for all  $z \in \mathbb{Z}$  with  $\min(x, y) < z < \max(x, y)$ . Bapat et al. [1] transferred the concept to trees. A function  $f : V \rightarrow \mathbb{R}$  defined on the nodes  $V$  of a tree is quasiconvex if for all distinct  $u, v, w \in V$  it holds  $u \sim v \sim w$  implies  $f(v) \leq \max(f(u), f(w))$ . Using strict inequality yields strictly quasiconvex functions. Peakless functions are more general than strictly quasiconvex functions but more restricted than quasiconvex functions. From an

optimization perspective, this brings advantages over both classes. On the one hand, non-chordal graphs such as cycle graphs do not admit any strictly quasi-convex functions, which limits their applicability compared to peakless functions. Quasiconvex functions, on the other hand, cannot be minimized efficiently even on very restricted graph classes. For example, consider a path graph where all nodes except for one have value 1 and only one node has value 0. Then, it is not possible to find this node without querying all nodes in the worst case.

*Bayesian optimization of graph functions.* In the last few years, some authors have started using Bayesian optimization techniques to optimize graph functions. According to their paper, Baptista et al. [2] are the first to propose a query-efficient algorithm for optimizing over discrete structured domains. Oh et al. [13] propose a method for Bayesian optimization of discrete variables. They define a combination graph of the variables and optimize the resulting function. However, their method is restricted to a specific graph class (product graphs of multiple variables). Wan et al. [15] propose a more general method for Bayesian optimization of graph functions, where the graph structure does not have to be known in advance. Liang et al. [10] extend this to optimization over node subsets, i.e., the goal is to find a subset of a given size with optimal value. While these methods demonstrate strong empirical performance, none of them have theoretical guarantees regarding their query complexity.

## 5 Future Work

One open question for this setting is how noise (or not perfectly peakless functions) can be modeled. There exist different noise models and adversarial settings in the graph search literature [7]. One common model is that each oracle answer is wrong independently or adversarially with probability  $p < 0.5$ . Another model is that an oracle has a fixed number of mistakes. Translated into the optimization setting, both models mean that (1) the underlying function is still peakless, but the observations are noisy, and that (2) queries can be repeated to obtain correct answers. While this is interesting for some cases, an open question is how the problem can be modeled for functions that are only close to a peakless function, and where repeated queries for the same node always return the same value.

Another open problem is the  $\log \Delta$  gap in the query complexity. While it is known that graph search on trees is possible with  $\frac{\Delta}{\log \Delta} \log n$  queries, it is unknown whether this also holds for general graphs.

A third interesting open problem is to extend the analysis to graphs which are not completely known in advance, as assumed by Wan et al. [15]. In this case, the classical graph search algorithms (for example, [7, 8]) do not work since they rely on computing a central edge. Dereniowski et al. [7] took a first step in this direction by investigating graph search in unbounded integer ranges.

Finally, experiments on real-world data and an empirical comparison with Bayesian optimization techniques are interesting to investigate the applicability of our results.

## 6 Conclusion

In this paper, we theoretically studied the query-efficient minimization of peakless functions on graphs. We showed that all chordal graphs admit interesting (i.e., non-trivial to optimize) peakless functions. This extends the previously investigated notion of strictly quasiconvex functions, which are only defined on trees. As our second main result, we showed that peakless functions can be efficiently minimized on bounded-degree graphs. We also showed a lower bound on the query complexity. The bounds are near-tight, only a gap of  $\log \Delta$  remains.

**Acknowledgments.** This work was funded in part by the Vienna Science and Technology Fund (WWTF), project StruDL (ICT22-059); by the Vienna Science and Technology Fund (WWTF) [Grant ID: 10.47379/VRG23013]; by the Austrian Science Fund (FWF), project NanOX-ML (6728). MT acknowledges support from a DOC fellowship of the Austrian academy of sciences (ÖAW).

**Disclosure of Interests.** The authors have no competing interests to declare that are relevant to the content of this article.

## References

1. Bapat, R., Kalita, D., Nath, M., Sarma, D.: Convex and quasiconvex functions on trees and their applications. *Linear Algebra and its Applications* **533**, 210–234 (2017)
2. Baptista, R., Poloczek, M.: Bayesian optimization of combinatorial structures. In: *ICML. Proceedings of Machine Learning Research*, vol. 80, pp. 471–480. PMLR (2018)
3. Ben-Asher, Y., Farchi, E.: The cost of searching in general trees versus complete binary trees. Tech. rep., Citeseer (1997)
4. Cambini, R., Riccardi, R.: On discrete quasiconvexity concepts for single variable scalar functions. *Taiwanese Journal of Mathematics* **13**(2B), 837–853 (2009)
5. Cambini, R., Riccardi, R., Yüceer, Ü.: An approach to discrete convexity and its use in an optimal fleet mix problem. In: *Generalized Convexity and Related Topics*. pp. 133–148. Springer (2006)
6. Chepoi, V.: Peakless functions on graphs. *Discret. Appl. Math.* **73**(2), 175–189 (1997)
7. Dereniowski, D., Tiegel, S., Uznanski, P., Wolleb-Graf, D.: A framework for searching in graphs in the presence of errors. In: *SOSA. OASICs*, vol. 69, pp. 4:1–4:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2019)
8. Emamjomeh-Zadeh, E., Kempe, D., Singhal, V.: Deterministic and probabilistic binary search in graphs. In: *STOC*. pp. 519–532. ACM (2016)
9. Hirai, H.: Discrete convex functions on graphs and their algorithmic applications. In: *Combinatorial Optimization and Graph Algorithms: Communications of NII Shonan Meetings*. pp. 67–100. Springer (2017)
10. Liang, H., Wan, X., Dong, X.: Bayesian optimization of functions over node subsets in graphs. In: *NeurIPS* (2024)
11. Miller, B.L.: On minimizing nonseparable functions defined on the integers with an inventory application. *SIAM Journal on Applied Mathematics* **21**(1), 166–185 (1971)



12. Murota, K., Shioura, A.: Quasi  $m$ -convex and  $l$ -convex functions—quasiconvexity in discrete optimization. *Discret. Appl. Math.* **131**(2), 467–494 (2003)
13. Oh, C., Tomczak, J.M., Gavves, E., Welling, M.: Combinatorial bayesian optimization using the graph cartesian product. In: *NeurIPS*. pp. 2910–2920 (2019)
14. Pezzo, L.M.D., Frevenza, N., Rossi, J.D.: Convex and quasiconvex functions in metric graphs. *Networks Heterog. Media* **16**(4), 591–607 (2021)
15. Wan, X., Osselin, P., Kenlay, H., Ru, B., Osborne, M.A., Dong, X.: Bayesian optimisation of functions on graphs. In: *NeurIPS* (2023)
16. Yüceer, Ü.: Discrete convexity: convexity for functions defined on discrete spaces. *Discret. Appl. Math.* **119**(3), 297–304 (2002)