Perfect landmark sets in the ALT route planning algorithm

Michał Dębski , Konstanty Junosza-Szaniawski , Zbigniew Lonc Michał Tuczyński , — Warsaw University of Technology, Poland[1]

Abstract

A perfect landmark set is a set L of vertices such that every shortest path can be extended to a shortest path from a vertex in L to some other vertex. We denote the size of a minimum perfect landmark set in G by $\mathrm{mPLS}(G)$.

Landmark sets play a crucial role in the ALT algorithm that finds shortest paths in weighted graphs. For a fixed perfect landmark set L in a graph G, after some preprocessing using $O(|L||V(G)|)$ memory, it is possible to determine the distance between any two vertices in time $O(|L|)$. Therefore, it is desirable to find small perfect landmark sets. We show that determining $\mathrm{mPLS}(G)$ is NP-hard.

We also introduce the parameter $\mathrm{mPLS}^*(G)$ equal to the minimum of $\mathrm{mPLS}(H)$ over all weighted graphs H such that G is an isometric subgraph of H. Sometimes $\mathrm{mPLS}^*(G)$ is much smaller than $\mathrm{mPLS}(G)$ and a further improvement of performance of route planning can be obtained.

There is a lot of ongoing research on routing planning algorithms in transportation networks. Modern systems handle huge transportation networks and need to answer a large number of queries in a short time. Therefore, it becomes increasingly more important to develop methods with the best possible performance. We refer to the survey by Bast et al. [1] for the state of the art in the field.

Dijkstra's algorithm is the classic solution for finding a shortest path between two given vertices in a weighted graph. It scans all vertices in the order of increasing distances from the starting point until the target is reached. It means that in the worst case the whole graph is scanned. However, it is possible to reduce the amount of computation needed by using some additional properties of the network. The A^* search is a modification of Dijkstra's algorithm that reduces the number of vertices scanned. It requires a function called a potential which assigns to each vertex v a lower bound on the distance from v to the target. Using this information the algorithm is able to skip scanning vertices that are known to be too far from the target to be of any relevance. If the potential of every vertex is close to its real distance from the target, the method is much faster than Dijkstra's algorithm; in the extreme case when the potential is exactly equal to the desired distance, only vertices along a shortest path are scanned. Usually the potential is based on some additional properties of a graph, e.g its geometrical representation.

A universal approach to constructing a useful potential function is to choose an arbitrary subset L of the vertices, called a landmark set, and precompute the distances from each landmark to every vertex of the graph. Using the triangle inequality we obtain $\mathrm{dist}(v,t) \geq \mathrm{dist}(v,l) - \mathrm{dist}(t,l)$ for every vertex v and landmark l, where t is the target. It follows that $\pi(v) = \max\{|\mathrm{dist}(s,l) - \mathrm{dist}(t,l)| : l \in L\}$, is a viable potential function. This approach is called ALT (A for A^*, L for landmarks and T for triangle inequality) [3]. Note that adding more vertices to L may result in a more useful potential π, but π is computed in time $O(|L|)$ and uses auxiliary data that takes $O(|L|\,|V(G)|)$ memory, so we want to keep the number of added vertices as low as possible.

If we fix the number of landmarks then the problem of choosing optimal landmark set is NP-hard [2]. Our approach is complementary: to find a perfect landmark set L of minimum cardinality. We say that L is perfect if for any two vertices s, t there exists a vertex $l \in L$ such that

[1]email: {m.debski, k.szaniawski, z.lonc, m.tuczynski}@mini.pw.edu.pl

$\mathrm{dist}(s,t) = |\mathrm{dist}(s,l) - \mathrm{dist}(t,l)|$. It means that the distance between any vertices s, t is given by the formula $\mathrm{dist}(s,t) = \max\{|\mathrm{dist}(s,l) - \mathrm{dist}(t,l)| : l \in L\}$ and can be computed in $O(|L|)$ time using $O(|L|\,|V(G)|)$ precomputed data. If L is small then we obtain a very fast implementation of the ALT algorithm finding a shortest path. Note that a naive approach applied to dense graphs – to precompute all distances – would give an algorithm that gives an answer in constant time, but uses $O(|V(G)|^2)$ memory for precomputed data, which is by far too much. Notice that our notation of perfect landmark set is different than metric basis, used in the context of metric dimension of graphs. Therefore, our understanding of landmarks should not be confused with landmarks used in that context.

Consider as an example a rectangular grid – a Cartesian product of two paths – with all edges of weight 1. It turns out that two corners on one side e.g. two left corners form a 2-element perfect landmark set. This shows that the minimum landmark set can be arbitrary small comparing to the number of vertices in a graph.

Grids are also interesting for practical reasons as they have a structure similar to the street graphs of cities. If we consider a weighted grid such that shortest paths in the sense of weights coincide with shortest paths in the corresponding unweighted graph, then still two corners on one side form a 2-element perfect landmark set, which gives us efficient algorithm for the shortest path. The distance between two given vertices can be computed in a constant time and the shortest path can be found in linear time.

We have also observed that sometimes the size of a minimum perfect landmark set can be significantly reduced by adding some extra vertices to the graph, which gives a further increase in performance of the computation of shortest paths. There are examples of graphs with arbitrary large minimum perfect landmark set that are isometric subgraphs of a graph with a small (e.g 2-element) perfect landmark set. This means that the ALT algorithm can be efficiently used for much larger classes of graphs.

By $G = (V, E)$ we denote simple graph with the vertex set V and edge set E. By G_w we denote weighted graph i.e. a graph G with a weight function $w : E \to \mathbb{R}_+ \cup \{0\}$. We treat unweighted graphs as a weighted graph with all edges of weight one.

By $\mathrm{dist}_G(u,v)$ in G_w we denote the minimum of $\{\sum_{e\in E(P)} w(e) : P \text{ is } u - v - \text{path}\}$. We omit index in dist_G if the graph G is clear from the context. For any two vertices u, v holds $\mathrm{dist}(u,v) = \mathrm{dist}(v,u)$. We say that a path P is a shortest $u-v$-path if $\sum_{e\in E(P)} w(e) = \mathrm{dist}(u,v)$. A path P is called a *geodesic* in a graph G or G_w if it is a shortest path joining its ends. A geodesic is *maximal* if it can not be extended to a longer geodesic. A set $L \subseteq V(G)$ is a *perfect set of landmarks* for G, if every maximal geodesic has an end in L. Let $\mathrm{mPLS}(G)$ denote the size of the smallest perfect landmark set in a simple or weighted graph G. Consider any geodesic joining u and v. Let P be geodesic joining u and v and let P' be a maximal geodesic containing P. Let l be an end of P' belonging to a perfect landmark set (from the definition at leas one end of maximal geodesic belongs to L). Then $\mathrm{dist}(u,v) = |\mathrm{dist}(v,l) - \mathrm{dist}(u,l)|$ and we obtain

Remark 1. *Let L be a perfect landmark set in G then for any two vertices u, v holds $dist(u,v) = \min\{|dist(v,l) - dist(u,l)| : l \in L\}$*

By $\widehat{G}$ we denote a graph $(V, \widehat{E})$ where $uv \in \widehat{E}$ if and only if u and v are the ends of maximal geodesic. Directly from definitions we get:

Remark 2. *A set L is a minimal perfect landmark set in G if and only if it is minimal vertex cover in the graph $\widehat{G}$.*

Example 3. *1.* $\widehat{C}_{2n} = nK_2$ *and* $mPLS(C_{2n}) = n$,

2. $\widehat{C}_{2n+1} = C_{2n+1}$ *and* $mPLS(C_{2n+1}) = n + 1$,

3. $\widehat{K}_{n,m} = K_n \cup K_m$ *and* $mPLS(K_{n,m}) = n + m - 2$,

4. $\widehat{P_m \times P_n} = 2K_2 \cup (mn - 4)K_1$ *and* $mPLS(P_m \times P_n) = 2$,

5. $\widehat{T} = K_t \cup (|T| - t)K_1$, *and* $mPLS(T) = t - 1$,*where* T *is a tree with* t *leaves.*

By a *twin* in a graph we mean a pair of vertices whose sets of neighbors are the same. Clearly, the vertices in a twin are nonadjacent.

Theorem 4. *The problem of determining if* $mPLS(G)$ *does not exceed* k *is NP-complete, when* k *is a part of the instance.*

Proposition 5. *Let* C *be a cycle in* G *of length* m *such that every path of length at most* $\frac{1}{2}m$ *contained in* C *is a geodesic. Then each set of landmarks in* G *has at least* $\lceil \frac{m}{2} \rceil$ *elements.*

Although it may seem counterintuitive at first, sometimes a perfect landmark set – and therefore, an amount of computation needed to find shortest paths in a graph – may be reduced by adding some artificial vertices. However, we need to take some extra care so that the shortest paths in the original graph are still shortest in the extended graph. This motivates the following definitions.

We say that a graph G is an *isometric* subgraph of H if for every $u, v \in V(G)$ $\mathrm{dist}_G(u, v) = \mathrm{dist}_H(u, v)$. We define $\mathrm{mPLS}^*(G)$ to be the minimum of $\mathrm{mPLS}(H)$ over all weighted graphs H such that G is an isometric subgraph of H.

As an example, consider the graph G from Figure 1 drawn with solid lines; we have $\mathrm{mPLS}(G) = 12$, because $\widehat{G}$ is $2K_{6,6}$. When we add two extra vertices and dashed edges of weight 5, we obtain an extended graph that behaves as a graph satisfying the assumptions of Theorem **??**, so $\mathrm{mPLS}^*(G) = 2$.

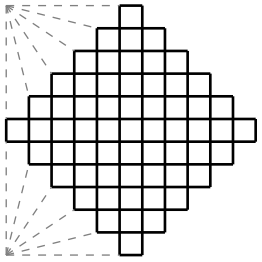


Figure 1: A graph G with $\mathrm{mPLS}(G) = 12$ and $\mathrm{mPLS}^*(G) = 2$

Let us start with a simple lower bound on the parameter $\mathrm{mPLS}^*(G)$. We observe that the proof of Proposition 5 can be easily (with almost no changes) adapted to prove the following statement.

Proposition 6. *Let* m *be the length of a longest cycle in* G *such that every path of length at most* $\frac{1}{2}m$ *contained in the cycle is a geodesic. Then*

$$mPLS^*(G) \geq \left\lceil \frac{1}{2}m \right\rceil.$$

The lemma beneath tells how to add a small set of vertices that will "handle" all maximal geodesics that end in a larger set S, provided that S has some nice property.

Lemma 7. *Let G be a simple graph and $d \in \{1, 2\}$. Consider a set $S \subseteq V(G)$ such that every two vertices of S are at distance d in G and every vertex from S is adjacent to every vertex from $N(S) \setminus S$. There exists a graph H such that*

(i) G is an isometric subgraph of H,

(ii) $|V(H) \setminus V(G)| = \lceil \log_2(|S| + 1) \rceil$,

(iii) there is no maximal geodesic path in H with both ends in S,

(iv) every maximal geodesic path in H with one end in S has the other end in $V(H) \setminus V(G)$.

Using Lemma 7 we show an upper bound on mPLS* of a clique, a star and a bipartite graph.

Proposition 8. *For every $n \geq 1$ we have $mPLS^*(K_{1,n}) \leq \lceil \log_2(n + 1) \rceil$.*

Proposition 9. *For every $n \geq 1$ we have $mPLS^*(K_n) \leq \lceil \log_2(n + 1) \rceil$.*

Proposition 10. *For every $n, m \geq 1$ we have $mPLS^*(K_{n,m}) \leq \lceil \log_2(n + 1) \rceil + \lceil \log_2(m + 1) \rceil$.*

Lemma 11. *Let T_w be a weighted tree, r the radius of T and Δ the maxiumum degree of a vertex of T. There exists a graph H_w such that*

(i) $|V(H_w) - V(T_w)| = r \lceil \log_2(\Delta) \rceil$,

(ii) T_w is an isometric subgraph of H_w,

(iii) every maximal geodesic path in H_w has at least one end in $V(H_w) - V(T_w)$.

Proposition 12. *Let T_w be a weighted tree, r the radius of T and Δ the maxiumum degree of a vertex of T. Then $mPLS^*(T_w) \leq r \lceil \log_2(\Delta) \rceil$.*

Lemma 13. *K_n can not be covered by less than $\Omega\left(\frac{\log n}{\log \log n}\right)$ triangle-free graphs.*

Theorem 14. *$mPLS^*(K_{1,n}) = \Omega\left(\frac{\log n}{\log \log n}\right)$.*

Theorem 15. *$mPLS^*(K_n) = \Omega\left(\frac{\log n}{\log \log n}\right)$.*

Corollary 16. *For every graph G with maximum degree Δ we have $mPLS^*(G) = \Omega\left(\frac{\log \log \Delta}{\log \log \log \Delta}\right)$.*

References

[1] H. Bast, D. Delling, A. V. Goldberg, M. Müller-Hannemann, T. Pajor, P. Sanders, D. Wagner and R. F. Werneck, Route Planning in Transportation Networks. Algorithm Engineering - Selected Results and Surveys, volume 9220 of LNCS, 19-80. Springer, 2016.

[2] R. Bauer, T. Columbus, B. Katz, M. Krug, D. Wagner, Preprocessing Speed-Up Techniques Is Hard. In: T. Calamoneri, J. Diaz (eds) Algorithms and Complexity. CIAC 2010. LNCS 6078.

[3] A. V. Goldberg and C. Harrelson. Computing the shortest path: A* search meets graph theory. In Proceedings of the 16th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'05), 156–165. SIAM, 2005.