Triangle Inequality for Prim distances

Mark Kliger, Steve Damelin, P. Olvier and A. Hero

September 30, 2007

Abstract

Triangle inequality for Symmetric Prim distance is established.

Keywords: Minimum Spanning Tree, Prim's algorithm, triangle inequality, dimensionality reduction, spectral clustering.

1 Introduction

2 Prim's Algorithm

Prim's algorithm

3 and Action!

Let $V = \{v_1, \dots, v_n\} \subset \mathbb{R}^d \subseteq S$, where S is a *nice* set (see definition of the nice set at [**?**]). Let G=$\{V, E\}$ be the fully connected weighted graph, where E set of edges e_{ij} between each pair of vertices $v_i, v_j \in V$, $1 \leq i, j \leq n$, where edge weight equal to Euclidean distance between the points. One of standard techniques for finding *Minimum Spanning Tree* (MST) of this graph is *Prim's algorithm*. Prim's algorithm operates by continually trying to attach new edge of minimum weight (in our case minimum length) to the existing tree. Prim's algorithm may start from any arbitrary vertex of V. We will use Prim algorithm to compute new "clustering motivated" distance between vertices of G.

For each pair of points $x, y \in V$ define the *one-sided Prim distance* $d(x \to y)$ as a total edge length of Prim tree rooted at x and stopped at y ($T_{x\to y}$). From the definition of the Prim tree, it is clear that this distance is not symmetric. Hence, we will define a symmetric version of it, *Symmetric Prim's Distance* (SPD) as:

$$d_{SP}(x, y) = d(x \to y) + d(y \to x) \tag{1}$$

It is easy to see that if one-sided Prim distance satisfy triangle inequality, hence the same is true for symmetric Prim distance. Indeed, if

$$d(x \to z) + d(z \to y) \geq d(x \to y) \tag{2}$$

for any point $z \in V$, then

$$\begin{aligned} d_{SP}(x, z) + d_{SP}(z, y) &= d(x \to z) + d(z \to x) + d(z \to y) + d(y \to z) \\ &\geq d(x \to y) + d(y \to x) = d_{SP}(x, y) \end{aligned} \tag{3}$$

Hence it is enough to prove (**??**) in order to show that SDP is a *metric*. Let $T_{x\to y} = \{V_{x\to y}, E_{x\to y}\}$ be a Prim tree rooted at x and stopped at y. Let $V_{x\to y} = \{x_0 = x, x_1, \dots, x_{m-1}, x_m = y\} \subset V$ and $E_{x\to y} = \{e_1, \dots, e_m\} \subset V$ be the sets of vertices and edges of Prim tree respectively, arranged in order of their appearance in Prim algorithm, *i.e.* at step k of the algorithm we attach point x_k to the tree by edge e_k.

In the case where $z \in V \setminus V_{x\to y}$ it is easy to see that $d(x \to y) \leq d(x \to z)$ and hence (**??**) is satisfied. Indeed the Prim tree $T_{x\to z}$ is growing from tree $T_{x\to y}$ by attaching to latter vertices and edges, till the former will include z. Hence, $E_{x\to y} \subset E_{x\to z}$ and total length of $T_{x\to z}$ is greater then total length of $T_{x\to y}$.

The situation is more complicated in the case where $z \in V_{x\to y}$. We need some additional definitions: For each vertex $x_i \in V_{x\to y}$, $i > 0$, we let $x_{\pi(i)}$, $\pi(i) < i$, denote the "parent" of vertex x_i. Here, the function π symbolized parent-child relationship (π^2 grandparent, *e.c.*). Thus, at step i, edge e_i connects x_i to the vertex $x_{\pi(i)}$, which is already belongs to Prim tree from previous steps of the algorithm. Although we are borrowing our terminology from directed graphs world, we should remind that G is undirected graph. However, rooted Prim tree can be treated as a directed one.

Let $T_{z\to} = \{V_{z\to}, E_{z\to}\}$ denote Prim tree rooted at z. In the following we demonstrate growing process of this tree in case where $z \in V_{x\to y}$. Let $z = x_k$, $1 \leq k \leq n - 1$ (situations where $z = x$ or $z = y$ are trivial). Our goal is to check whether a vertices x_j, $k < j \leq n$ will belong to $T_{z\to y}$.

We will start from vertex x_{k+1} - the next vertex attached to Prim tree $T_{x\to y}$ immediately after z. If $e_{k+1} < e_k$ then we can conclude that $\pi(k+1) = k$, *i.e.* x_{k+1} is a child of x_k. Otherwise, x_{k+1} would be attached to Prim tree before x_k, that contradicts to our definitions. Moreover, from the definition of the Prim's algorithm, x_{k+1} is a closest neighbor of x_k. Hence, we will attach x_{k+1} to $T_{z\to}$. We add x_{k+1} to $V_{z\to}$ and e_{k+1} to $E_{z\to}$. Then we will proceed to next vertex, x_{k+2}. In contrary, if $e_{k+1} > e_k$, then the closest neighbor of x_k is it parent, $x_{\pi(k)}$. We will attach $x_{\pi(k)}$ to $T_{z\to}$. Does x_{k+1} will be attached to $T_{z\to}$ on latter steps?

If $e_{k+1} < e_{\pi(k)}$, and accidently $\pi(k+1) = k$, *i.e.* x_{k+1} is a child of x_k, we will attach it to $T_{z\to}$ and will proceed to a next vertex x_{k+2}. However, if x_{k+1} is not a child of x_k, *i.e.* $\pi(k+1) < k$, it can be easily shown that $\pi(k) \leq \pi(k+1)$, *i.e.* parent of x_{k+1} attached to Prim tree $T_{x\to y}$ after $x_{\pi(k)}$ or it $x_{\pi(k)}$ itself. Otherwise x_{k+1} would be attached to Prim tree $T_{x\to y}$ before $x_{\pi(k)}$, that contradict to our definitions. Moreover, since $\pi(k) \leq \pi(k+1) < k$ then $e_{\pi(k+1)} < e_k$ (parent of x_k was attached before $x_{\pi(k+1)}$, but x_k itself was attached latter) and therefore $e_{\pi(k+1)} < e_{\pi(k)}$. The last inequality shows that $\pi(k) \leq \pi^2(k+1)$, otherwise $x_{\pi(k+1)}$ would be attached to Prim tree $T_{x\to y}$ before $x_{\pi(k)}$, that also leads to contradiction. We may continue to perform the same operation till it will converge, *i.e.* $\pi(k) = \pi^m(k+1)$, for some $m \geq 1$. In other words, x_{k+1} is a descendant of $x_{\pi(k)}$. Hence, since the distance from x_k to its next closest neighbor is at least e_{k+1} we will attach to $T_{z\to}$ all appropriate descendants of $x_{\pi(k)}$ including x_{k+1}. Then we will proceed to next vertex, x_{k+2}.

If $e_{k+1} > e_{\pi(k)}$, we will attach to $T_{z\to}$ all descendant of $x_{\pi(k)}$ which a closer to it then x_k, and after this we will attach it parent, $x_{\pi^2(k)}$ (the distance from x_k to its next closest neighbor is at least e_{k+1}). Then we will check once again if $e_{k+1} > e_{\pi^2(k)}$ and we will recursively continue the procedure of attaching ancestors of x_k and their appropriate descendants, till we will find an ancestor $x_{\pi^l(k)}$ such that $e_{k+1} < e_{\pi^l(k)}$. Then by repeating the same steps as in previous paragraph, we can show that x_{k+1} is a descendant of $x_{\pi^l(k)}$. Hence, we will attach to $T_{z\to}$ all all appropriate descendants of $x_{\pi^l(k)}$ including x_{k+1}. Then once again we will proceed to next vertex, x_{k+2}.

From the above discussion, it is clear that all verteces x_j, $k \leq j \leq n$ are attached to $T_{z\to y}$, and hence $\{e_j, k \leq j \leq n\} \subseteq E_{z\to y}$. Hence,

$$d(z \to y) \geq \sum_{j=k+1}^{n} e_j \tag{4}$$

However by definition

$$d(x \to z) = \sum_{j=1}^{k} e_j \tag{5}$$

Finally,

$$d(x \to z) + d(z \to y) \geq \sum_{j=1}^{n} e_j = d(x \to y) \tag{6}$$

References

[1] R. C. Prim, “Shortest connection networks and some generalisations.” *Bell System Technical Journal*, 36, pp. 13891401, 1957.

[2] S. Grikschat, J. Costa, A.O. Hero and O. Michel, “Dual rooted-diffusions for clustering and classification on manifolds,” *Proc. 2006 IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, Toulouse France, 2006.

[3] J. M. Steele, *Probability Theory and Combinatorial Optimization*, SIAM, 1997.