Geometric Partial Matching

Pankaj K. Agarwal, Hsien-Chih Chang, Allen Xiao

Problem Statement

Geometric Bipartite Matching: Given two equal-sized point sets A, B in the plane, find a *perfect matching* minimizing the sum of lengths of matching edges.
Geometric Partial Matching: Given two point sets A, B in the plane, find a *size-k matching* minimizing the sum of lengths of matching edges.

Background

Let $n = \max(|A|, |B|)$ and $m = |A \times B| = O(n^2)$. Primal-dual algorithms for general graphs can solve the problem in $O(km + k^2 \log n) = O(kn^2 + k^2 \log n)$ time [4].
However, there are faster algorithms for geometric bipartite matching versus perfect matching in general graphs, using dynamic data structures for **bichromatic closest pair** and **nearest neighbors**, which query/update in $O(\operatorname{polylog} n)$ time [2]. Roughly speaking, we can replace the $O(m)$ with $O(n \operatorname{polylog} n)$ or $O(k \operatorname{polylog} n)$ in many instances. If $O(m)$ is no longer the running time bottleneck, can we design a faster algorithm for partial matching?

Results

Building from existing primal-dual augmentation algorithms,

1. An exact algorithm running in time $O((n + k^2) \operatorname{polylog} n)$, using the Hungarian algorithm [3].
2. A $(1 + \varepsilon)$-approximation algorithm running in time $O((n + k\sqrt{k}) \operatorname{polylog} n \log(1/\varepsilon))$, using a **cost-scaling** algorithm for unit-capacity minimum-cost flow by Goldberg, Hed, Kaplan, and Tarjan [1].

Think of these running times as $O((n + f(k) \cdot k) \operatorname{polylog} n)$ (modulo scaling):
"After $O(n \operatorname{polylog} n)$ preprocessing time, find $f(k)$ augmenting paths in $O(k \operatorname{polylog} n)$ time each."

Primal-Dual Augmentation Algorithms

A classical algorithm for min-cost matching and min-cost flow: grow the matching using the least-cost augmenting path. Essentially, solve a **single-source shortest paths** problem (Dijkstra). Improves with geometry — can query the "next shortest edge" as a bichromatic closest pair for A, B in the plane.

Techniques

1. **Rewinding**: It takes $O(n \operatorname{polylog} n)$ time to construct the dynamic data structures from scratch. We can't afford to do this every augmentation, but we can build it from the *initial data structure of the previous augmentation* in $O(\operatorname{polylog} n)$ time. This costs $O(k \operatorname{polylog} n)$ time to "rewind" changes and recover the initial data structure state.
2. **Null vertices**: Certain vertices cannot be part of any augmenting path. After filtering these out, each BCP query "makes progress" towards finding an augmenting path so the search for an augmenting path takes $O(k \operatorname{polylog} n)$ time.

References

[1] Andrew V. Goldberg et al. "Minimum-Cost Flows in Unit-Capacity Networks". In: *Theory Comput. Syst.* 61.4 (2017), pp. 987–1010. DOI: `10.1007/s00224-017-9776-7`. URL: `https://doi.org/10.1007/s00224-017-9776-7`.

[2] Haim Kaplan et al. "Dynamic Planar Voronoi Diagrams for General Distance Functions and their Algorithmic Applications". In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19.* 2017, pp. 2495–2504. DOI: `10.1137/1.9781611974782.165`. URL: `https://doi.org/10.1137/1.9781611974782.165`.

[3] Harold W Kuhn. "The Hungarian method for the assignment problem". In: *Naval Research Logistics (NRL)* 2.1-2 (1955), pp. 83–97.

[4] Lyle Ramshaw and Robert E. Tarjan. "A Weight-Scaling Algorithm for Min-Cost Imperfect Matchings in Bipartite Graphs". In: *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012.* 2012, pp. 581–590. DOI: `10.1109/FOCS.2012.9`. URL: `https://doi.org/10.1109/FOCS.2012.9`.