Popular Conjectures as a Barrier for Dynamic Planar Graph Algorithms

Amir Abboud and Søren Dahlgaard

Final Report by Rahul Kumar CS21M049

1 Introduction

Dynamic Shortest path problem ask us to preprocess the graph G such it supports

i insertion or removal of an edge (u,v).

ii we can answer to the query that asks for length of shortest path from between two nodes u and v.

The best algorithm that can perform query and update both takes $\tilde{O}(n^{2/3})$ time, based on ideas of a seminal paper by Fakcharoenphol and Rao. A $(1+\epsilon)-$ approximation algorithm takes $\tilde{O}(\sqrt{n})$ time in update and query. In this paper, they have followed a recent and very active line of work on showing lower bounds for polynomial time problems based on popular conjectures, obtaining the first such results for natural problems in planar graphs. Such results were not obtained because of non-planar nature of known reductions. In this Paper they have used a graph construction technique which is based on grid construction of matrix and distance labeling. They have used this framework to prove that no algorithm for dynamic APSP or maximum bipartite matching can perform update and query in amortized time $O(n^{1/2-\epsilon})$ for any $\epsilon > 0$ unless the All Pair shortest Path Conjecture fails.

2 Previous Works

- Using naive algorithm we perform query on planar graph in $\tilde{O}(n)$ time.
- The first sublinear bound was obtained in the seminal paper by Fakcharoenphol and Rao[1]. The amortized time per operation was $O(n^{2/3}log^{7/3}(n))$ if negative weights are not allowed. If negative weights are allowed then it is $O(n^{2/3}log^{13/5}(n))$.
- Follow up works by Klein[2] ,Italiano et al[3] and Kaplan et al[4] reduced this time to $O(n^{\frac{2}{3}}\frac{log^{5/3}n}{log^{4/3}log(n)})$.

- In a recent SODA'16 paper, Abraham et al. [5]study worst case bounds under a restricted but realistic model of dynamic updates in which a base graph G is given and one is allowed to perform only weight updates subject to the following constraint: For any updated graph $G^{'}$ it must hold that $d_G(u,v) \leq d_{G^{'}(u,v) \leq M d_G(u,v)}$ for all u,v and some parameter M. In this model, the authors obtain a $(1+\epsilon)-$ approximation algorithm that maintains updates in time $O(poly(logn)M^4/\epsilon^3)$. Without this restriction, the best known $(1+\epsilon)-$approximation algorithms use $\tilde{O}(\sqrt{n})$.

3 Results

Theorem 1. *No algorithm can solve the dynamic APSP problem in planar graphs on N nodes with amortized query time q(N) and update time u(N) such that $q(N).u(N) = O(N^(1-\epsilon))$ for any $\epsilon > 0$ unless APSP conjecture is false. This holds even if we only allow weight updates to G.*

Theorem 2 *No algorithm can solve the dynamic APSP problem in unit weight planar graphs on N nodes with amortized query time $q(N)$ and update time $u(N)$ such that $max(q(N)^2.u(N), q(N).u(N)^2) = O(N^{1-\epsilon})$ for any $\epsilon > 0$ unless the OMv conjecture is false. This holds even if we only allow weight updates*

Theorem 3.*No algorithm can solve the dynamic maximum weight matching problem in bipartite planar graphs on N nodes with amortized update time $u(N)$ and query time $q(N)$ such that $max(q(N), u(N)) = O(N^{\frac{1}{2}-\epsilon})$ for any $\epsilon > 0$ unless APSP Conjecture is false. Furthermore, if $q(N) \geq u(N)$ the algorithm cannot have $q(N).u(N) = O(N^{1-\epsilon})$. This holds even if the planar embedding of G never changes.*

Theorem 4. *No algorithm can solve the dynamic APSP problem in planar graphs on N nodes with amortized update and query time $O(N^{\frac{1}{2}-\epsilon})$ for any $\epsilon > 0$ unless APSP Conjecture is false. This holds even if only weight updates are allowed*

4 Conjectures used

Conjecture 1(APSP Conjecture). *There exists no algorithm for solving the all pairs shortest paths (APSP) problem in general weighted (static) graphs in time $O(n^{3-\epsilon})$ for any $\epsilon > 0$*

Conjecture 2(OMv Conjecture)*For any constant $\epsilon > 0$ there is no algorithm that can solve OMv problem in $O(n^{3-\epsilon})$ time. It takes at least $O(n^{3-o(1)})$ time.*

5 Techniques and Idea

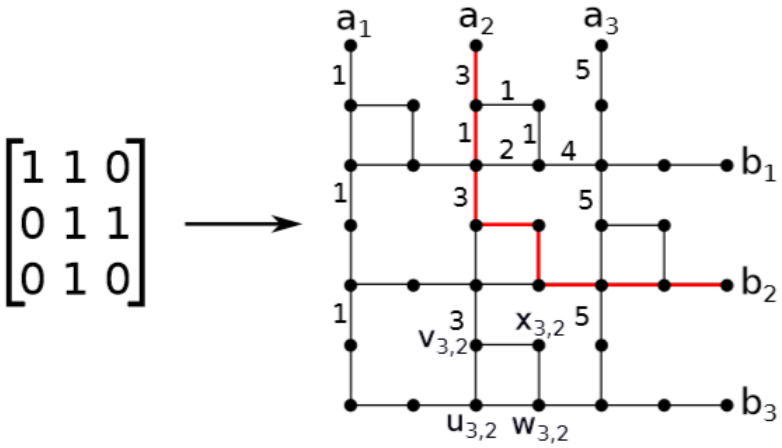
Techniques that are used in this paper are

- Grid construction: Given matrix M they have prepared to types of grid graph. one for the boolean matrix and other for normal matrix.
- Distance labeling scheme: They have used a distance labeling scheme to assign weights to edges of weighted graph

Idea in proving the theorem 1,3 and 4 is that it is going reduce from min-plus matrix product problem to dynamic APSP problem . while for proving theorem 2 we are going to use special version of OMv problem called OuMv problem. We will reduce from OuMv problem to dynamic APSP problem.

6 Grid Construction

Definition 1. given a boolean matrrix of size $R \times C$. We call the following construction the grid embedding of M. Let G_M be the grid graph with R rows and C column. $u_{i,j}$ is the node at the intersection of i_{th} row and j_{th} column. $u_{1,1}$ is on the top-left corner and $u_{R,C}$ is on the bottom-right corner. Add C nodes $a_1, a_2, ..., a_C$ and edges $(u_{1,j}, a_j)$ above G_M. Similarly add R nodes $b_1, b_2, ..., B_r$ and edges to the right of G_M.
Now subdivide each vertical edge adding the node $v_{i,j}$ above $u_{i,j}$, and subdivide each horizontal edge adding the node $w_{i,j}$ to the right of $u_{i,j}$. Finally, for each entry of M such that $M_{i,j} = 1$ add the node $x_{i,j}$ and edges $(v_{i,j}, x_{i,j})$ and $(w_{i,j}, x_{i,j})$ to the graph.
Each edge $(u_{i,j}, v_{i+1,j})$ and $(a_j, v_{1,j})$ has weight $2 \times j - 1$. Each edge $(w_{i,j}, u_{i,j+1})$ and $(w_{i,C}, b_i)$ has weight $2 \times R - 2$. Each edge $(u_{i,j}, w_{i,j})$ has weight 2. All the remaining edges has weight 1.
We will call the two-edge path $v_{i,j} \rightarrow x_{i,j} \rightarrow w_{i,j}$ a shortcut as it has length 1 less than the path $v_{i,j} \rightarrow u_{i,j} \rightarrow w_{i,j}$. Clearly the graph constructed is the subgraph of $(2R + 1) \times (2C + 1)$ rectangular grid.
Here is an example of graph constructed from a Matrix of size 3×3.



Proposition 1. Let M be a boolean $R \times C$ matrix and let G_M be its grid embedding as defined in Definition 1. Then for any $1 \leq i \leq R, 1 \leq j \leq C$ and $i < k \leq R$ the shortest path distance from $u_{i,j}$ to b_k is exactly
$(k-i).2j + 2R(C-j+1)$, if $M_{k,j} = 0$
$(k-i).2j + 2R(C-j+1) - 1$, Otherwise

Corollary 1. Let M be a $R \times C$ and let G_M be its grid embedding. Then for any $1 \leq j \leq C, 1 \leq k \leq R$, the distance between a_j and b_k in G_M is exactly
$2R.(C-j+1) + 2jk$, if $M_{i,j} = 0$
$2R.(C-j+1) + 2jk - 1$, if $M_{i,j} = 1$

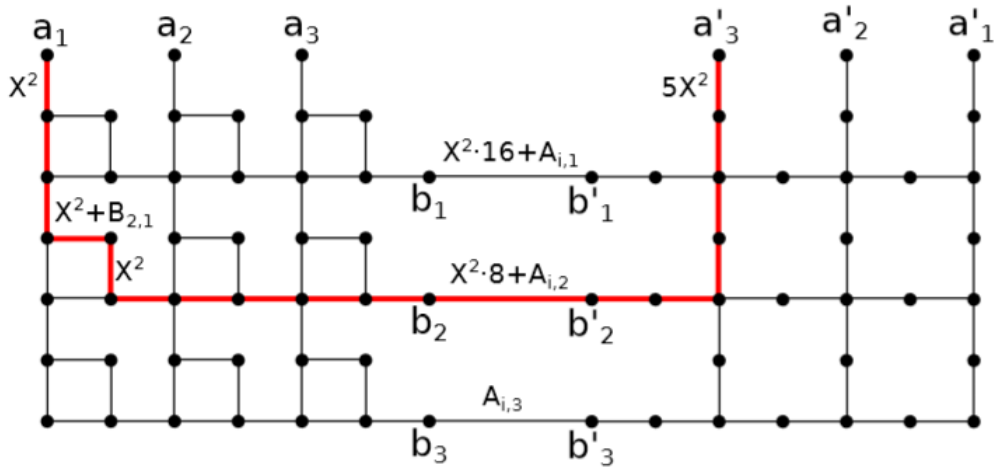
Definition 2. Let M be a $R \times C$ matrix with integer weights in $\{0, ..., X\}$ We will call the following construction the grid embedding of M. Let G_M be the grid embedding from Definition 1 for the all ones matrix of size $R \times C$ and multiply the weight of each edge by X^2. Furthermore, for each edge $(v_{i,j}, x_{i,j})$ increase its weight by $M_{i,j}$.

Corollary 2. Let M be a $R \times C$ matrix with integer weights in $\{0, ..., X\}$ and let G_M be its grid embedding. Then for any $1 \leq k \leq R$,$1 \leq j \leq C$, the distance between a_j and b_k in G_M is exactly
$X^2.(2R.(C-j+1) + 2jk - 1) + M_{k,j}$

7 Hardness of dynamic APSP in planar graphs

First we are going to prove Theorem 4. The main idea in proving Theorem 4 is to reduce from the APSP problem by first reducing to (min,+)-Matrix-Multiplication problem and use the grid construction from Section 2 to represent the matrices to be multiplied. We then perform several shortest paths queries to simulate the multiplication process. Given the Matrices A and B of size $n \times n$ for the $A \oplus B$ min-plus matrix multiplication. we assume that A and B have integer weight in range $\{0, ..., X\}$ such that $X = poly(n)$.We use matrix B to construct graph. Then we define graph G which consists of two part G_B and $G_B^{'}$. G_B is the graph constructed using definition 2 of grid construction using the matrix B.

let $G_B^{'}$ be the grid embedding of B mirrored along the vertical axis with all shortcuts removed. Now for each $1 \leq k \leq n$ add the edge $(b_k, b_k^{'})$ and define G to be this graph. The Graph for n=3 will look something like this.The red line is showing shortest path from a_1 to $a_3^{'}$.



Now we perform a phase for each row i of A as follows:

1. For each $1 \leq k \leq n$ set the weight of the edge $(b_k, b_n^{'})$ to $X^2.(2(n+1)(n-k)) + A_{i,k}$.
2. For each j query the distance between a_j and $a_{n-j+1}^{'}$.

Let $d_G(a_j, a_{n-j+1}^{'})$ represents the length of shortest distance between a_j and $a_{n-j+1}^{'}$. Shortest path from s_j to $a_{n-j+1}^{'}$ must go via one of the edge from $(b_k, b_k^{'})$. Therefore,

$$
\begin{aligned}
& d_G(a_j, a_{n-j+1}^{'}) = d(a_j, b_k) + w(b_k, b_k^{'}) + d(b_k^{'}, a_{n-j+1}^{'}) \\
& d(a_j, b_k) = X^2.(2R.(C-j+1) + 2jk - 1) + B_{k,j} \text{ From Corollary 2} \\
& w(b_k, b_k^{'}) = X^2.(2(n+1)(n-k)) + A_{i,k} \\
& d(b_k^{'}, a_{n-j+1}^{'}) = X^2.(2n.j + 2(n-j+1)k) \\
& \implies d_G(a_j, a_{n-j+1}^{'}) = X^2.4n(n+1) - X^2 + A_{i,k} + B_{k,j}
\end{aligned}
$$

Dominant term in $X^2.4n(n+1) - X^2 + A_{i,k} + B_{k,j}$ does not contains k. Therefor shortest will choose the k such that $A_{i,k} + B_{k,j}$ is minimum. and this is what we want in min-plus matrix multiplication. So for the min-plus matrix mult. we can get $C_{i,j}$ by just subtracting $X^2.4n(n+1) - X^2$ from queried distance. Number of update and query performed is $O(n^2)$ and so if some algorithm perform query and update in $O(N^{1/2-\epsilon})$ time then conjecture 1 will fail. Hence it proves the theorem 4. using the above discussion will prove theorem 1.

7.1 Trade-offs

proof theorem 1:

Instead of balance version of min-plus matrix mult. consider the unbalanced version. Let A and B be $n \times n^{\beta}$ and $n^{\beta} \times n^{\alpha}$ matrices respectively for some $0 < \alpha, \beta \leq 1$ We define the initial graph G from B in the same

manner as in Theorem 4. We then have a phase for each row i of A as follows:

1 For each $1 \leq k \leq n^{\beta}$ set the weight of edge $(b_k, b_k^{'})$ to $X^2.(2(n^{\alpha}+1)(n^{\beta}-k)) + A_{i,k}$.

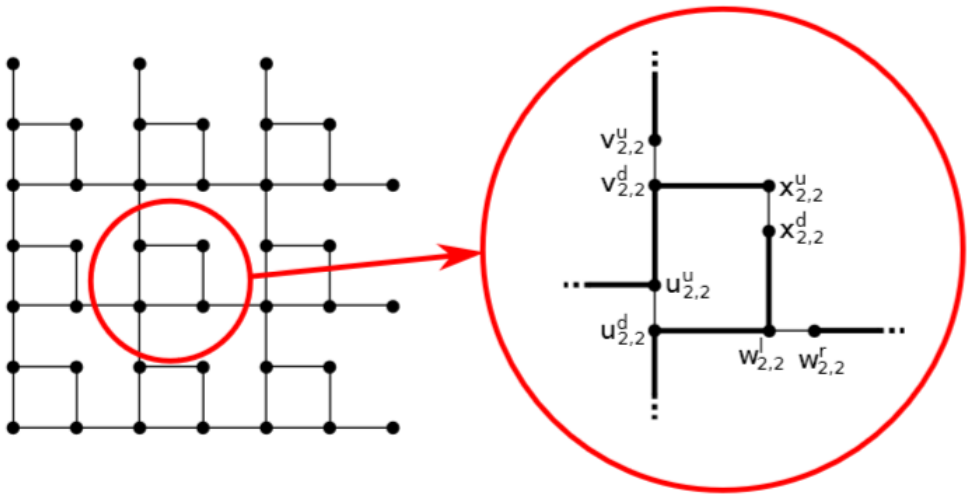
2 For each $1 \leq j \leq n^{\alpha}$ query the distance between a_j and $a_{n^{\alpha}-j+1}^{'}$

The entry $C_{i,j}$ is exactly the distance $d_G(a_j, a_{n^{\alpha}-j+1}^{'})$ from the ith phase minus $X^2.(4n^{\beta}(n^{\alpha}+1)-1)$ the correctness of above reduction follow from the proof of theorem 4.
Now, Number of nodes in the graph is $\theta(n^{\alpha+\beta})$.Number of update performed is $O(n^{1+\beta})$ and number of query performed is $O(n^{1+\alpha})$. Any algorithm solving this problem must use total time $n^{1+\alpha+\beta-o(1)}$ time unless Conjecture 1 is false. It follows that either updates must take $n^{\alpha-o(1)}$ amortized time or queries must take $n^{\beta-o(1)}$ amortized time.

Assume now that an algorithm exists such that queries take $O(N^{\gamma})$ amortized time for any $0 < \gamma < 1$. We wish to show that this algorithm cannot perform updates in amortized time $O(N^{1--\epsilon})$for any $\epsilon > 0$. Pick $\beta = +\epsilon/2$ and set $\alpha = 1-\beta$. We now use the above reduction to create a dynamic graph G with $N = O(n)$ nodes. Since queries do not take $n^{\beta-o(1)}$ time it follows from the above discussion that updates must take $n^{\alpha-o(1)} = n^{1-\gamma-\frac{\epsilon}{2}-o(1)}$ time. Since this is polynomially greater than $O(N^{1-\gamma-\epsilon})$ the claim follows.

8 Hardness of dynamic maximum weight matching in bipartite planar graphs

Proof of Theorem 3:
Proof of this theorem uses same type of reduction. except that each vertex is replace by two new vertex. $a_j, u_{i,j}, x_{i,j}, v_{i,j}$ get replace by two new vertices with super script u(up) and d(down). b_i and $w_{i,j}$ gets replace by two new vertices with superscript l(left) and r(right). they are connected by edges of weight 0. We gets graph something like this,



To prove theorem 3 we create a graph as similar to that we created in proof of theorem 1. except that we will make the changes mentioned above. We now add two additional nodes s and t to the initial graph and perform a phase for each row i of A as follows:

1 For each $1 \leq k \leq n^{\beta}$ set the weight of edge $(b_k, b_k^{'l})$ to be $X^2.(2(n^{\alpha}+1)(n^{\beta}-k)+A_{i,k}$.

2 For each $1 \leq j \leq n^{\alpha}$ do the following three steps; 1) add the edge (s, a_j^u) and $(t, a^{'l}_{n^{\alpha}-j+1})$, 2)query the minimum weight perfect matching , 3) delete the two edges added in 1.

It follows that we get the same trade-offs for minimum weight perfect matching as for APSP with the exception that the trade-off only holds when $q(N) \geq u(N)$ since we perform $O(1)$ updates for each query. Hence it proves theorem 3.

9 Unweighted

proof of Theorem 2:

To prove this it uses OuMv problem. It uses the Matrix M to construct the graph and then perform the following algorithm,

We perform a phase as follows for each vector pair (u^i, v^i),

1 For each k such that $u^i_k = 1$ connect b_k and $b^{'}_k$ to their respective path.

2 For each j such that $v^i_j = 1$ query the distance from a_j to $a^{'}_{n^{\alpha}-j+1}$.

3 Remove all the edges added in step 1.

If the answer to any of the queries during the ith phase is $4n(n+1)-1$ the answer to the ith product is 1 and otherwise the answer is 0. Number of nodes $N = O(n^{2\beta+\alpha} + n^{2\alpha+\beta})$.Number of update $O(n^{1+\beta})$, number of query $O(n^{1+\alpha})$. Any algorithm solving this problem must take $n^{1+\alpha+\beta-o(1)}$ time unless OMv conjecture is false. thus either updates take $n^{\alpha-o(1)}$ time or queries take $n^{\beta-o(1)}$ time. We will assume that $q(N) \geq u(N)$ and note that the other case follows symmetrically. Assume that some algorithm can perform queries in N^{γ} for some $\frac{1}{3} \leq \gamma \leq \frac{1}{2}$. We wish to show that this algorithm cannot perform updates in time $N^{1-2\gamma-\epsilon}$ for any $\epsilon > 0$. To do this, pick $\beta = \gamma + \epsilon/3$ and $\alpha = 1 - 2\beta$. Note that $\beta \geq \alpha$. Thus the graph has $N = O(n)$ nodes. It now follows by the above discussion that the algorithm cannot perform updates faster than $m^{\alpha-o(1)} = N^{1-2\gamma-\frac{2\epsilon}{3}-o(1)}$ which proves the claim. Hence theorem 2 proved .

10 Conclusion

All the algorithm proves lower bound on update and query time in Dynamic APSP or Maximum bipartite perfect matching problem on planar . Dynamic update and query on a planar graph is important problem for companies like Uber, Ola and google.. so any improvement these problem will benefit people on large scale.

11 References

[1] Jittat Fakcharoenphol and Satish Rao. Planar graphs, negative weight edges, shortest paths, and near linear time. Journal of Computer and System Sciences, 72(5):868–889, 2006. See also FOCS'01

[2] Philip N Klein. Multiple-source shortest paths in planar graphs. In SODA, volume 5, pages 146–155, 2005

[3] Giuseppe F Italiano, Yahav Nussbaum, Piotr Sankowski, and Christian Wulff-Nilsen. Improved algorithmsformincutandmaxflowinundirectedplanargraphs. InProceedings of the forty-third annual ACM symposium on Theory of computing, pages 313–322. ACM, 2011.

[4] Haim Kaplan, Shay Mozes, Yahav Nussbaum, and Micha Sharir. Submatrix maximum queries in monge matrices and monge partial matrices, and their applications. In Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms, pages 338–355. SIAM, 2012

[5] Ittai Abraham, Shiri Chechik, Daniel Delling, Andrew V. Goldberg, and Renato F. Werneck. On dynamic approximate shortest paths for planar graphs with worst-case costs. In Proc. 27th ACM/SIAM Symposium on Discrete Algorithms (SODA), pages 740–753, 2016