On Graph Problems in a Semi-Streaming Model

Introduction and Motivation:

→ Huge graphs; unable to store edges.
→ Only polylog space: hard to tackle.
So, we take the $\underbrace{\text{middle ground}}$: $O(n \text{ polylog } n)$ space, $n = |V|$.
interesting area. "Semi-streaming".

→ Examples of massive graphs:
→ call graphs: nodes are telephone numbers; edges are edges are calls b/w nos.
→ web graphs: nodes → webpages
edges → links b/w webpages.
Need streaming to handle such massive graphs!

Main Results:

→ semi-streaming algorithm for computing $\left(\frac{2}{3}-\varepsilon\right)$- approx. in $O\left(\frac{\log(1/\varepsilon)}{\varepsilon}\right)$ passes for unweighted bipartite matching.

→ One-pass semi-streaming algorithm for $\frac{1}{6}$-approximating the maximum weighted graph matching.

→ $\log n / \log\log n$ approximations for diameter and shortest paths in weighted graphs
Also give $\Omega(\log^{(1-\varepsilon)} n)$ lower bounds for these problems in unweighted graphs.

Preliminaries:

• $G=(V,E)$ a graph; $V = \{v_1, \ldots, v_n\}$, $E = \{e_1, \ldots, e_m\}$ } n vertices, m edges

Def: (Graph Stream) : Sequences of edges $e_{i_1}, e_{i_2}, \cdots, e_{i_m}$ where $e_{i_j} \in E$ and $i_1, i_2, \cdots, i_m$ is a permutation of $[m]$.

$\Longrightarrow$ Graph is revealed one edge at a time.

Efficiency in the semi-streaming model depends on the following.

(1) Space complexity.
(2) Processing Time per edge.
(3) No. of passes over the graph stream.

Def (Space Requirements): Let A be a semi-streaming graph algo.

$S(n,m)$: Space complexity of A (bits)
$P(n,m)$: # of (one-way) passes over the stream.
$T(n,m)$: Time to process a single edge.

We want: $S(n,m) = O(n \cdot \text{polylog}(n))$

Graph Matching:

→ Unweighted Bipartite Matching:

An algorithm to approximate unweighted bipartite matching.

Simple Algorithm to find a bipartition

→ As edges stream in, use a DSU to maintain the CCs of the graph.
→ Color vertices black/white s.t every edge is not monochromatic. If not possible, graph is not bipartite.

Finding maximal matching (not maximum!).

→ If M is a matching, $v \in V$ is said to be free if no edge of M is incident on it

→ Initially, let $M' = \phi$.
→ As edges stream in, add the edge to M' iff. both

ends of the edge are free.

Lemma: The above algorithm finds a maximal matching.

Proof: Suppose $\exists$ matching M^* s.t $M' \subsetneq M^*$.

So, some edge $e \in M^*$ st $e \notin M'$. Our algorithm has to add this edge to M'. Contradiction.

Note: Above algo. works for any graph, not just bipartite ones

Proposition: Any maximal matching is an $\frac{1}{2}$-approximation to a maximum matching, i.e

$M_{max} \leq 2M$, $M \rightarrow$ maximal matching.

Pf.

$e \in M_{max} \Rightarrow \exists e' \in M$ sharing a vertex with e.

Make a list:

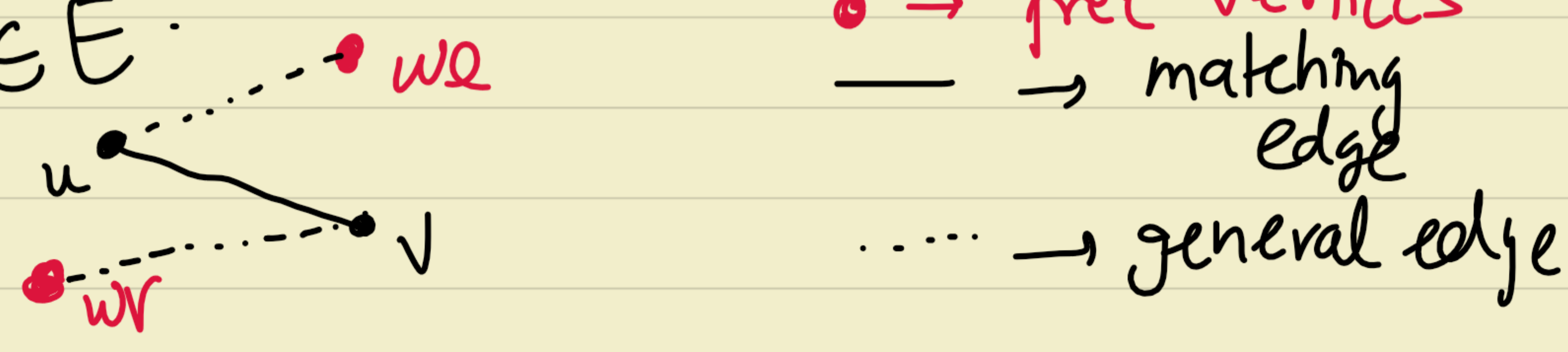
$e_1 : e_1', e_1''$
$e_2 : e_2', e_2''$
$\vdots \qquad \vdots$
$e_{max} : e_{max}', e_{max}''$

Because M_{max} is a matching, every edge $e' \in M$ appears in atmost 2 lists here. So,

$$\boxed{M_{max} \leq 2M}.$$

Cor: $\frac{1}{2}$-approx. to maximum matching can be computed (deterministically) using a single pass in the semi streaming model.

Def: Let $G = (L \cup R, E)$ be a bipartite graph. Let M be a matching. Let $e = (u,v) \in M$ be an edge, s.t $u \in L$, $v \in R$. A length 3 augmenting path for e is a quadruple (w_l, u, v, w_r) s.t w_l, w_r are free and $(w_l, u), (v, w_r) \in E$.



$w_\ell, w_r \to$ wing tips.
$(w_\ell, u) \to$ left wing
$(v, w_r) \to$ right wing

Simultaneously augmentable length 3 augmenting paths:
set of vertex-disjoint length 3 augmenting paths

Algorithm: To find a set of simultaneously augmentable length 3 augmenting paths.

Input: $G = (L \cup R, E)$, $M \to$ matching for G; parameter $\delta \in (0,1)$

1: In one pass, find a maximal set of disjoint left wings:
given edge $e = (u,v)$, check if:
(a) either $v \in L$ and u is free.
(b) or $u \in L$ and v is free.
Add this edge to our set if u, v haven't been added yet (as part of any left wing).
$\to$ clearly maximal.

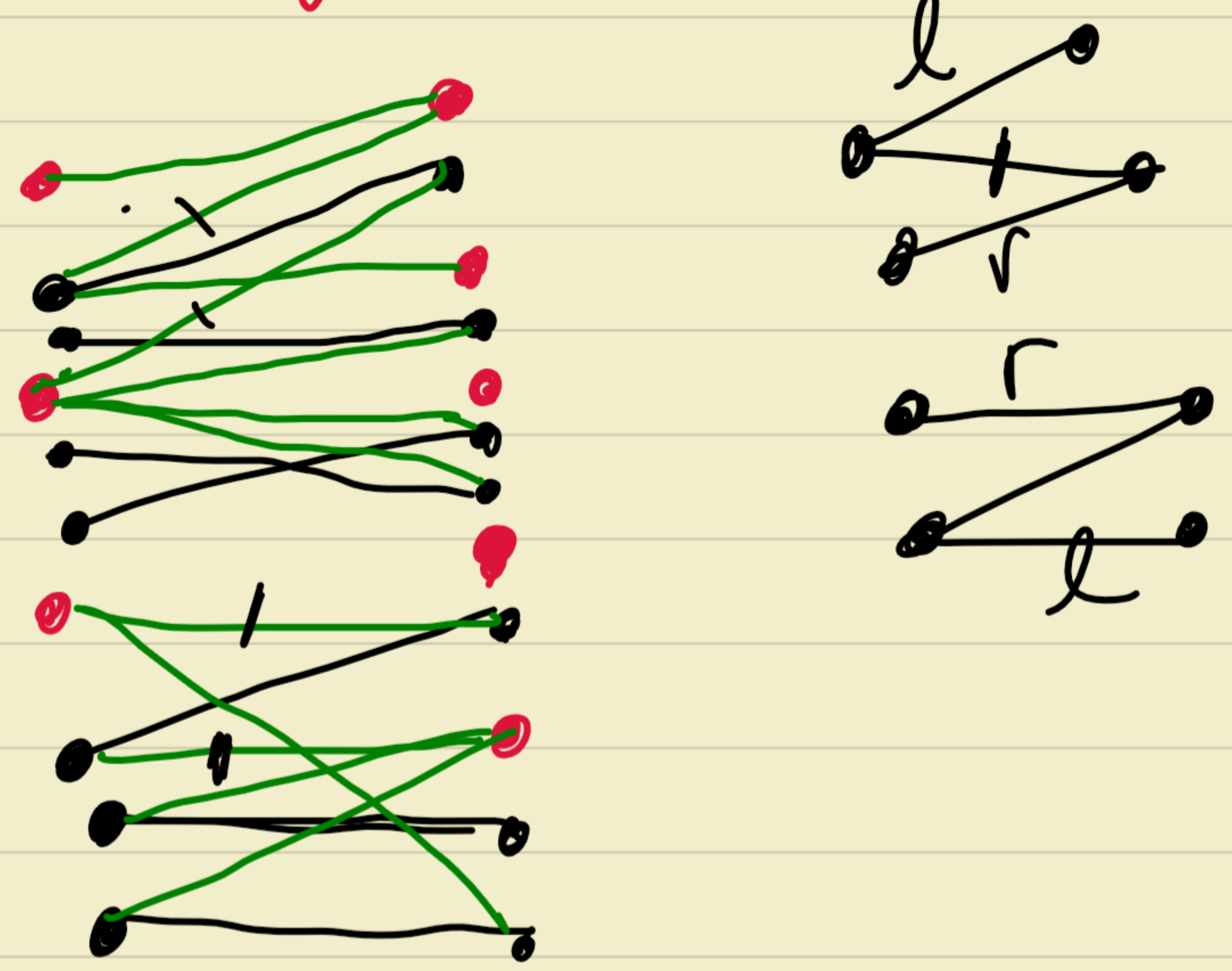
2: If no. of left wings found $\leq \delta M$, terminate

3: In a second pass, for the edges in M with left wings, find a maximal set of disjoint right wings: similar as above.

4. So now we've found a bunch of disjoint length 3 augmenting paths.

We now identify those vertices which:
- $\to$ Are endpoints of a matched edge which got a left wing.
- $\to$ Are the wing tips of a matched edge which got both wings.
- $\to$ Are endpoints of a matched edge that is no longer 3-augmentable.

In subsequent passes, we ignore edges incident on any one of these vertices.



5 Repeat!

Remark: In step 4; points (b) and (c) are clear. If we have found a length-3 augmenting path, there is no point in considering those vertices again. Similarly, if there is some matched edge which is no longer 3 augmentable, we just ignore the endpoints of that edge. The only non-trivial point is point (a): why do we ignore those matched edges which got a left wing?

Explanation: we claim that an edge which only got a left wing can never get a right wing in any subsequent pass. This is because if an edge with a left wing didn't get a right wing, all possible right wing tips are already taken by a length 3 augmenting path! . so, it makes sense to ignore that edge.

Algorithm 3: (Unweighted Bipartite Matching):

Input: $G = (L \cup R, E)$ and a parameter $0 < \varepsilon < \frac{1}{3}$.

(1) In one pass, find a bipartition of G and a maximal matching M.

(2) For $k = \lceil \log(6\varepsilon) / (\log(8/9)) \rceil$ Do:

(a) Run Alg 2. with G, M and $\delta = \dfrac{\varepsilon}{2 - 3\varepsilon}$

Augmentation Step.

(b) For each edge $(u,v) \in M$ for which an augmenting path (w_ℓ, u, v, w_r) is found by Alg 2, remove (u,v) from M and add (w_ℓ, u) and (v, w_r) to M.

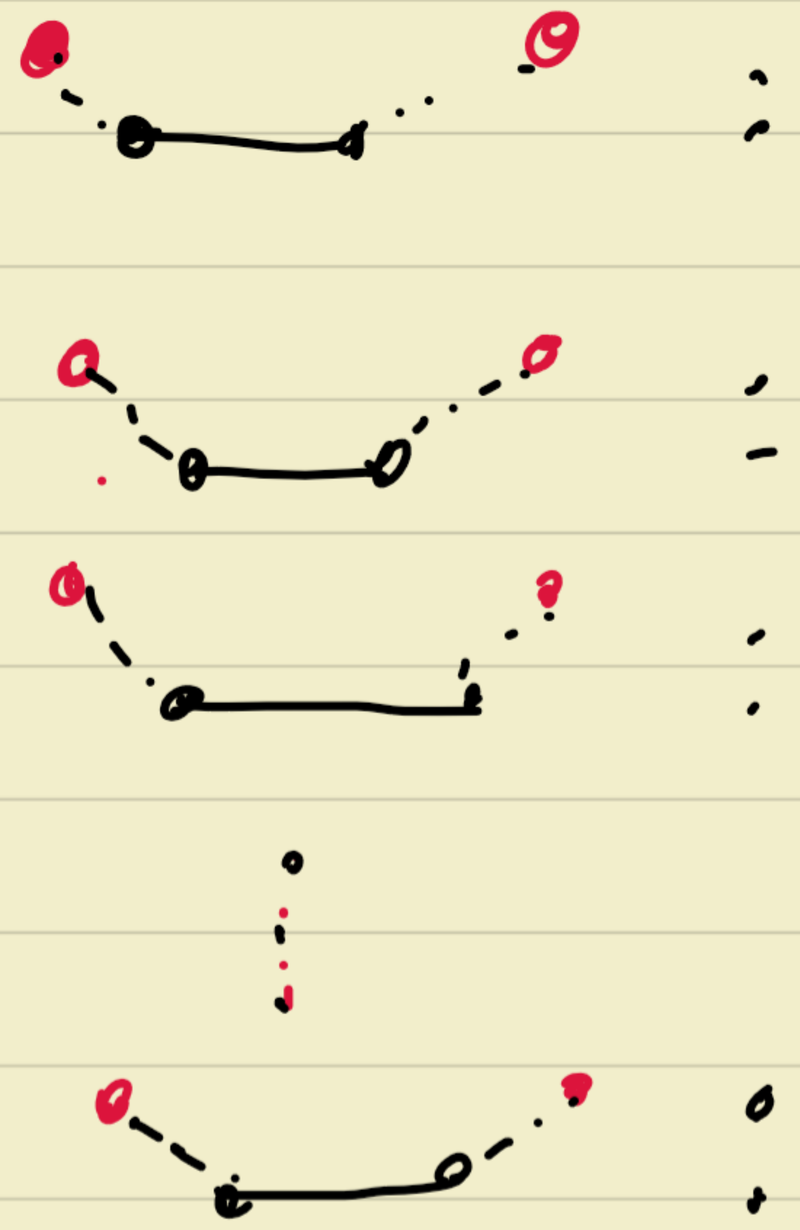
Analysis of the Algorithm:

Lemma 1: (Maximal set of simultaneously augmentable length 3 augmenting paths and the size of a maximum such set).

The size of a maximal set of simultaneously augmentable length 3 augmenting paths is atleast $\frac{1}{3}$ of the size of a maximum set of simultaneously augmentable length 3 augmenting paths.

Proof: Let AP_{max} be some maximum set of simultaneous length-3 augmenting paths. Let AP be a maximal set of such paths. We show that $AP_{max} \leq 3AP$.

Make a list: enumerate all 3-length augmenting paths in AP_{max}, and list all 3-length a.ps in AP which share atleast one vertex with it.



} total no. of lists $\leq 3AP$.

Claim: each element of AP appears in atmost 3 lists.
$\Rightarrow$ Proves the claim.

<u>Lemma 2</u>: (maximal vs maximum matching). Let X be a maximum-sized set of simultaneously augmentable length-3 augmenting paths for a maximal matching M. Let $\alpha = |X|/|M|$ and let OPT be a maximum matching. Then:

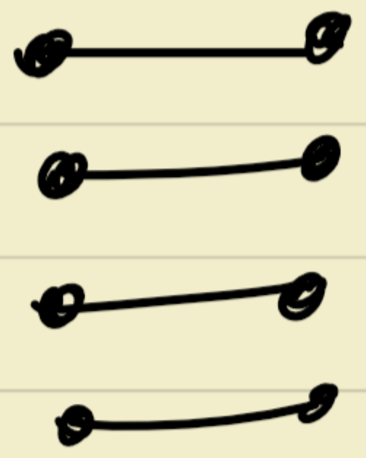
$$|M|(1+\alpha) = |M| + |X| \geq \frac{2}{3}|OPT|.$$

<u>Proof</u>: Consider the symmetric difference $OPT \triangledown M$
(Recall: $OPT \triangledown M = (OPT - M) \cup (M - OPT)$).

Consider the graph induced by this symmetric difference. Let C be a connected component of this graph. We claim that:

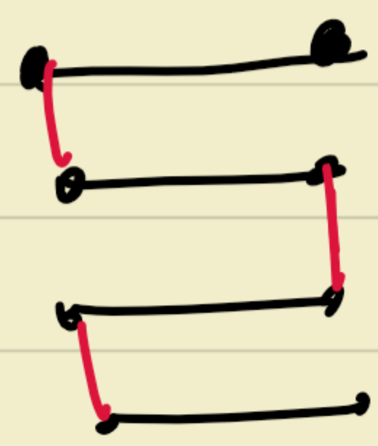
$$OPT_C \leq M_C + 1. \qquad (\star)$$

We prove this as follows: first, draw all edges of M_C:



} Now, all other edges are edges from OPT_C (not in M_C). Also, recall that no two edges of OPT_C intersect.

This means that <u>atmost two</u> edges of OPT_C are incident to any edge in M_C. So, for the connected component to be indeed connected, we must have the following structure:



} This has $M_C - 1$ edges in OPT_C. Atmost 2 more OPT_C edges are present in this CC. This proves $(\star)$

Infact, it follows that inequality (*) holds for the original graph as well, and not just the symmetric difference (this is easy to see).
From now on, we will refer to the original graph.
If C is any C.C of the original graph:

Either $M_C = 1$ and $OPT_C = 2$ or
$3|M_C| \geq 2|OPT|_C$ (using *).

Now, no. of C.Cs in which $M_C = 1$ and $OPT_C = 2$ is $\leq |X|$. (by the definition of X). Let S_1 be the set of these C.Cs, and let S_2 be the set of all other C.Cs.
So:

$$
\begin{aligned}
2|OPT| &= 2\sum_{C \in S_1} |OPT|_C + 2\sum_{C \in S_2} |OPT|_C \\
&\leq 4|S_1| + \sum_{C \in S_2} 3|M_C| \\
&\leq 6|S_1| + 3\sum_{C \in S_2} |M_C| \\
&= 3|S_1| + 3|S_1| + 3\sum_{C \in S_2} |M_C| \\
&\leq 3|X| + 3\sum_{C \in S_1} |M_C| + 3\sum_{C \in S_2} |M_C| = 3|X| + 3|M|
\end{aligned}
$$

QED.

<u>Lemma 3</u>: (Lower bound on size of set returned by Algorithm 2):
Algorithm 2 finds $(\alpha|M| - 2\delta|M|)/3$ simultaneously augmentable length-3 augmenting paths in $3/\delta$ passes.

<u>Proof</u>: Call each iteration of Alg 2 a phase. Note that there are atmost $\frac{1}{\delta}$ phases, as atleast δM edges of M are removed in each phase. Also, one phase includes 3 passes of the stream. So, there are atmost $\frac{3}{\delta}$ passes.

Now, define the following.

$L(M) :=$ left endpoints of edges in M.

$\text{Free}_L(M) :=$ set of all $v \in R$ which are free and $\exists\, u \in L(M)$ s.t. (u,v) is an edge.

In the last phase, we found $\leq \delta M$ disjoint left wings. Note that the set forms a <u>maximal matching</u> b/w the remaining vertices in $L(M)$ and $\text{Free}_L(M)$; so it follows that the <u>maximum</u> no. of disjoint left wings we could have found in the last step is $\leq 2\delta M$ (maximal matching $\rightarrow \frac{1}{2}$-approx. to maximum matching)

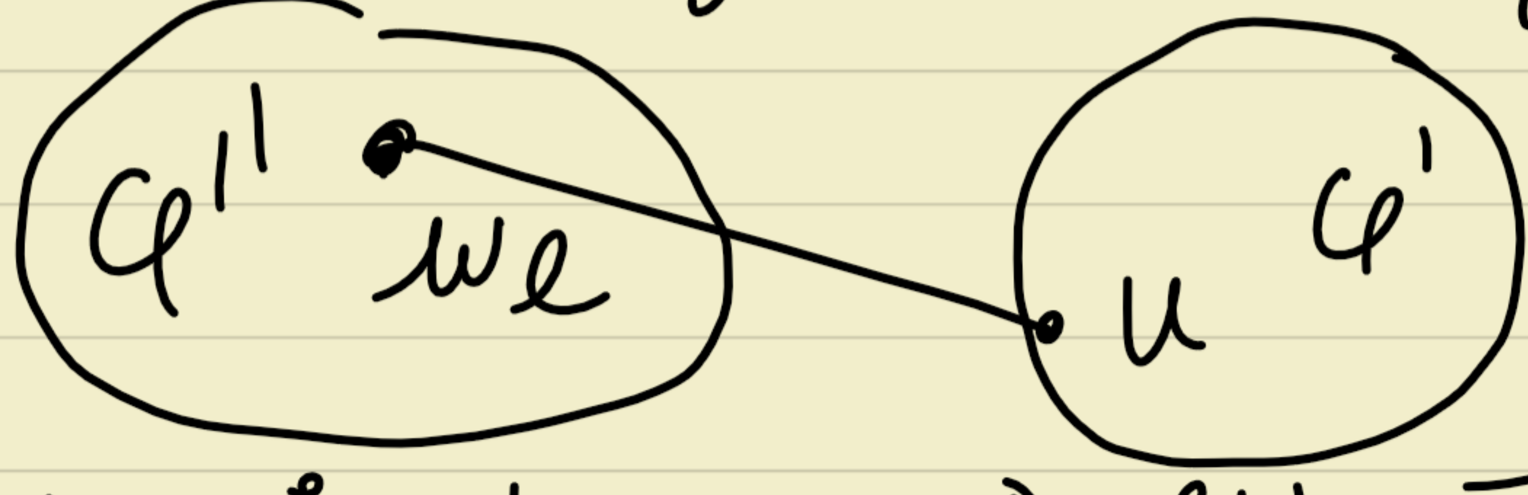
So, there are $\leq 2\delta M$ simultaneously augmentable length-5 augmenting paths in the remaining graph. Call this graph G'.

Now: Let $G'' = G / G'$. We first make this claim: any length-3 augmenting path in X lies completely either in G'' or G' (note that G/G' is <u>deleting the vertices of G' from G</u>). We now prove this: Suppose this is not true. Let (w_ℓ, u, v, w_r) be a length-3 augmenting path in X which has a vertex in G' and G''.

<u>Case 1:</u>

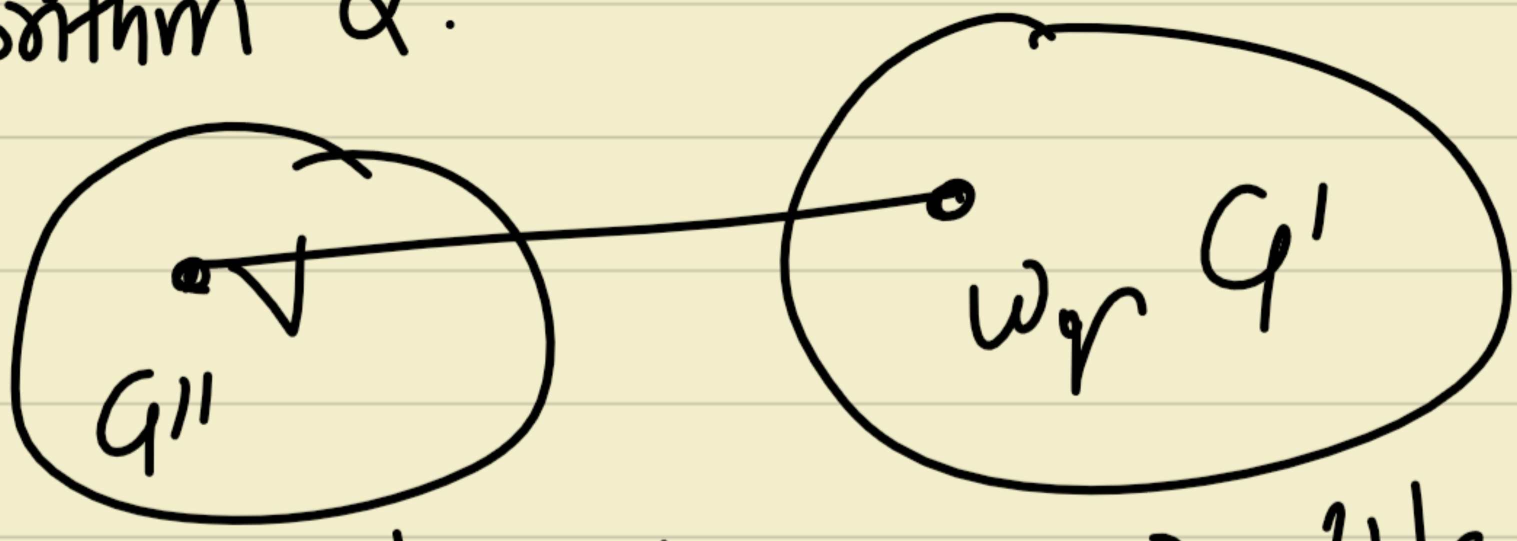
this case is not possible; the only u lands up in G'' is if u is matched to a vertex in G'' by M (look at algorithm 2, analyze how it behaves).

<u>Case 2:</u>



this is also not possible. The only way w_ℓ (a wing tip) is in G'' is if w_ℓ already is part of an augmenting path found by algorithm 2.

Case 3:



↳ also not possible, because w_r is already a part of an augmenting path found by Alg 2; so, it will land up in G''.

From the above claim, and using the fact that G' has $\leq 2\delta M$ simultaneously augmentable length-3 paths, we see that the Maximum set of simultaneously augmentable length 3 augmenting paths in G'' must have length atleast

$$\alpha|M| - 2\delta|M| = X - 2\delta|M|$$

(otherwise size of maximum set of sim. aug. length 3-aug paths in G would be $< |X|$, a contradiction).

Also, note that the set of length-3 augmenting paths found by Alg 2 is a maximal set; by lemma 1, the size of this set is $(\alpha|M| - 2\delta|M|)/3$, completing the pf.

Main Theorem: For any $0 < \varepsilon < \frac{1}{3}$ and a bipartite graph, Algorithm 3 finds a $\frac{2}{3} - \varepsilon$-approximation of a maximum matching in $O(\log 1/\varepsilon)/\varepsilon$ passes. The algorithm processes each edge in $O(1)$ time in each pass except the first pass, in which the bipartition is found. (Recall that we are using DSU to maintain connected components). The storage space required by the algorithm is $O(n \log n)$.

⟶

Pf. Let us first analyze the space complexity. To find the bipartition, we only need to maintain:

(1) The connected component to which a vertex belongs.

(2) The sign of each vertex, indicating to which partition the vertex belongs.

Clearly, (2) requires $O(n)$ space; (1) requires $O(n\log n)$ space (since max # of CCs $= n$).

Also, we need to store the edges of the maximal matching found; a matching on n vertices clearly cannot have size $> O(n)$; so again, storing the matching takes $O(n\log n)$ space.

Finally, each phase of Algorithm 2 needs $O(n\log n)$ space; we are just storing disjoint-vertex edges (like left wings, right wings). So overall, space required: $O(n\log n)$.

Now we prove the correctness of the algorithm. In the ith phase, suppose M_i is the matching found by our algorithm. Let X_i be a maximum set of length-3 simultaneously augmentable augmenting paths for M_i. As before:

$$\alpha_i = |X_i|/|M_i|$$

Also, let OPT be a maximum matching of G.

First, suppose $\alpha_i \le \dfrac{3\varepsilon}{(2-3\varepsilon)}$ for some phase i.

By lemma 2, we know that:

$$|M_i|(1+\alpha_i) \ge \frac{2}{3}|OPT|$$

$$\Rightarrow \quad |M_i| \ge \frac{2}{3}\cdot\frac{1}{(1+\alpha_i)}|OPT|$$

$$\ge \frac{2}{3}\left(\frac{1}{1+\frac{3\varepsilon}{2-3\varepsilon}}\right)OPT = \left(\frac{2}{3}-\varepsilon\right)\cdot|OPT|$$

and so M_i is already a $\frac{2}{3}-\varepsilon$ approximation to OPT.

So, we assume that $\alpha_i > \frac{3\varepsilon}{2-3\varepsilon}$ for all phases i.

Let $\delta = \frac{\varepsilon}{2-3\varepsilon}$. Our above assumption is the same as saying : $\delta \leq \frac{\alpha_i}{3}$ for all phases i.

Now, by Lemma 3: # of simultaneously augmentable length-3 augmenting paths found by Algorithm 2 at ith stage: is atleast

$$\frac{\left(\alpha_i|M_i| - 2\delta|M_i|\right)}{3} \geq \frac{\alpha_i|M_i| - \frac{2\alpha_i|M_i|}{3}}{3}$$

$$= \frac{\alpha_i|M_i|}{9} \qquad (*)$$

Now, more notation: at any stage i, let $s_i = \frac{|M_i|}{|OPT|}$, i.e s_i is the ratio b/w the matching M_i and the maximum matching OPT.

Because M_0 is a maximal matching, we know that:

$$s_0 \geq \frac{1}{2}$$

At any stage, by Lemma 2:

$$|M_i| + \alpha_i|M_i| \geq \frac{2}{3}|OPT|$$

$$\Rightarrow \quad s_i + \alpha_i s_i \geq \frac{2}{3} \quad \text{at all stages.}$$

Now, observe that:

$$|M_{i+1}| \geq |M_i| + \#\text{ of simultaneously augmentable length 3 augmenting paths}$$

because augmenting an edge increases the size of M_i by 1.

By lemma 3:

$$|M_{i+1}| \geq |M_i| + \frac{\left(\alpha_i|M_i| - 2\delta|M_i|\right)}{3}$$

$$= |M_i|\left(1 + \frac{\alpha_i - 2\delta}{3}\right) \geq |M_i|\left(1 + \frac{\alpha_i}{9}\right)$$

$\Rightarrow$ look at $(*)$ proven above

This is the same as saying:

$$s_{i+1} \geq s_i + \frac{\alpha_i s_i}{9}$$

Using the above inequality and using the fact that $s_i + \alpha_i s_i \geq \frac{2}{3}$ (shown above), we see that:

$$s_{i+1} \geq s_i + \frac{\alpha_i s_i}{9} = \frac{9s_i + \alpha_i s_i}{9}$$

$$= \frac{8s_i}{9} + \frac{(s_i + \alpha_i s_i)}{9}$$

$$\geq \frac{8s_i}{9} + \frac{2}{27}.$$

Unfold this recurrence (using $s_0 \geq \frac{1}{2}$), and obtain:

$$s_i \geq \left(\frac{8}{9}\right)^i \cdot \frac{1}{2} + \frac{2}{27}\left(1 + \frac{8}{9} + \cdots + \left(\frac{8}{9}\right)^{i-1}\right)$$

$$= \left(\frac{8}{9}\right)^i \cdot \frac{1}{2} + \frac{2}{27}\left(\frac{1 - \left(\frac{8}{9}\right)^i}{1 - \frac{8}{9}}\right)$$

$$= \left(\frac{8}{9}\right)^i \cdot \frac{1}{2} + \frac{2}{3}\left(1 - \left(\frac{8}{9}\right)^i\right)$$

$$= \frac{2}{3} - \frac{1}{6} \cdot \left(\frac{8}{9}\right)^i$$

Recall that the algorithm runs for $k = \left\lceil \frac{\log 6\varepsilon}{\log(8/9)} \right\rceil$ stages.

$$\Rightarrow s_k \geq \frac{2}{3} - \varepsilon$$

$$\Rightarrow \frac{|M_k|}{|OPT|} \geq \frac{2}{3} - \varepsilon \quad \text{(Proven)}.$$

Now, each stage of Algo 3 requires Algo 2; Algo 2 takes $\frac{3}{\delta} = \frac{(2-3\varepsilon)\cdot 3}{\varepsilon} = \frac{6-9\varepsilon}{\varepsilon}$ passes. So,

$$\# \text{ of passes} = k \cdot \frac{6-9\varepsilon}{\varepsilon} = \left\lceil \frac{\log 6\varepsilon}{\log(8/9)} \right\rceil \cdot \frac{6-9\varepsilon}{\varepsilon} = O\left(\frac{\log 1/\varepsilon}{\varepsilon}\right)$$

Important Note: At each phase, M_i is a maximal matching. We need this to apply Lemma 2 to M_i.
But this is true! Try to prove this.