Advanced Topics in Theoretical Computer Science

University of BRISTOL

Lecture 11

Matchings: Unweighted and Weighted

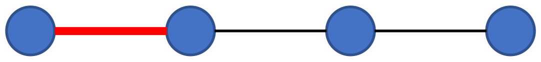
Matchings in Graphs

Definition: Let $G = (V, E)$ be a graph

- A *matching* $M \subseteq E$ is a subset of vertex-disjoint edges, i.e., for every $v \in V$: $|\{ab \in M : a = v \text{ or } b = v\}| \leq 1$.

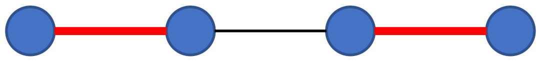
- A matching $M \subseteq E$ is *maximal* if it cannot be enlarged by adding an edge outside M to it, i.e., $M \cup \{e\}$ is not a matching, for every $e \in E \setminus M$.

- A matching $M^* \subseteq E$ is *maximum* if it is of largest size.



Matching

Maximal matching



Maximum matching

Not a matching:

Computing Matchings in the Streaming Model

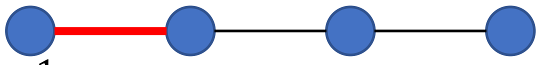
Goal: Semi-streaming algorithm for computing large matchings

How large a Matching can we compute?

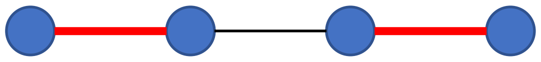
- Computing a maximum matching requires space $\Omega(n^2)$!
- Instead, we will compute approximations:

Definition. Let M^* be a maximum matching and let M be an arbitrary matching in G. Then, for $0 \leq c \leq 1$, M is a *c-approximate matching* if:

$$|M| \geq c \cdot |M^*|$$



$\frac{1}{2}$-approximate Matching



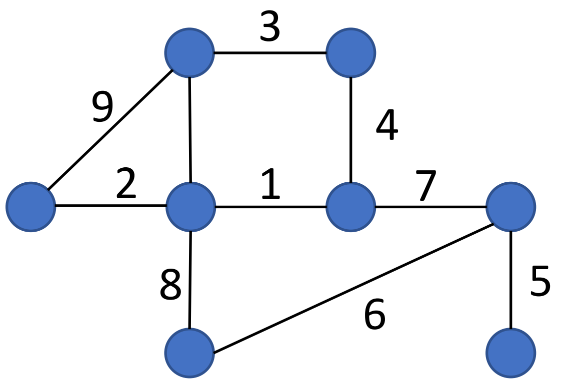
1-approximate Matching

The Greedy Matching Algorithm

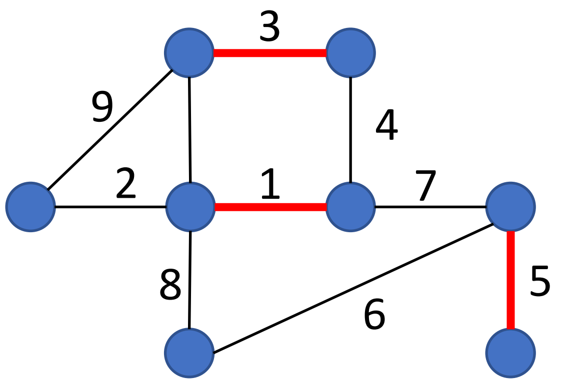
Greedy Matching Algorithm:

- Start with an empty matching $M \leftarrow \emptyset$
- Process all edges (any order): Upon arrival of edge uv, insert edge into M if both endpoints have not yet been matched, i.e., for every $ab \in M$:

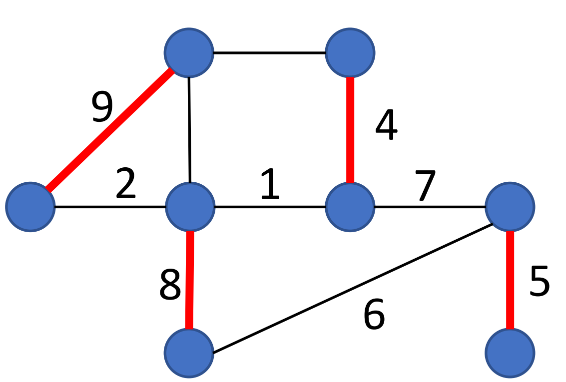
$$\{a, b\} \cap \{u, v\} = \emptyset.$$



Arrival order



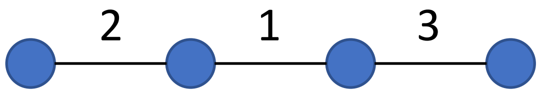
M produced by Greedy



Maximum Matching M^*

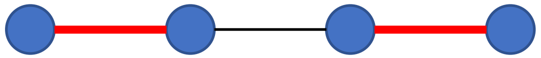
How good is Greedy?

What is the Approximation Factor of Greedy?



Arrival order


Greedy


Maximum matching

- Example above shows that approximation factor at best $\frac{1}{2}$, i.e., $\leq \frac{1}{2}$

- We will show that example above is the worst case and Greedy always produces at least a $\frac{1}{2}$-approximate matching!

Greedy Produces a Maximal Matching

Lemma. Let M be the matching produced by Greedy. Then M is maximal.

Proof.

- Suppose that M is not maximal. Then there exists an edge $e \in E \setminus M$ such that $M \cup \{e\}$ is a matching.
- However, when Greedy processed e it would have added e to M, a contradiction.

$\square$

Lemma. Let M be a maximal matching and let M^* be a maximum matching. Then:

$$|M| \geq \frac{1}{2}|M^*|.$$

Proof.

- If an edge of M^* is not included in M then it is blocked by an edge in M
- An edge in M blocks at most 2 edges from M^*

$\square$

Greedy Constitutes a Semi-streaming Algorithm

Semi-streaming:

- Greedy constitutes a semi-streaming (i.e., space $O(n \text{ poly} \log n)$) algorithm
- It has an approximation factor of $\frac{1}{2}$

Is there a semi-streaming algorithm with approximation guarantee > $\frac{1}{2}$?

Maybe… open problem (since 2004)

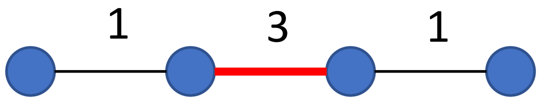
Is there a streaming algorithm with space $O(n^{1.9999})$ with approximation guarantee > $\frac{1}{2}$?

Maybe… open problem

Weighted Matching

Weighted Matching:

- Let $G = (V, E, w)$ be a weighted graph where $w: E \rightarrow \mathbb{N}$ is an edge weight function
- The weight of a matching M is the sum of the weights of its edges
- A *maximum weight matching* is a matching of largest weight
- We assume that the weight $w(e)$ of edge e appears together with e in the stream



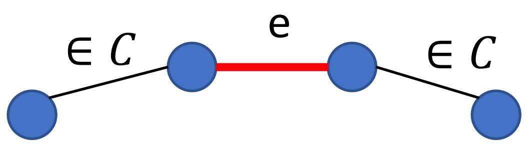
Maximum weight matching
(not a maximum matching!)

Goal: Semi-streaming algorithm for approximating a maximum weight matching

Weighted Matching

Streaming Algorithm for Weighted Matching by Eviction: [Feigenbaum et al., 2005]

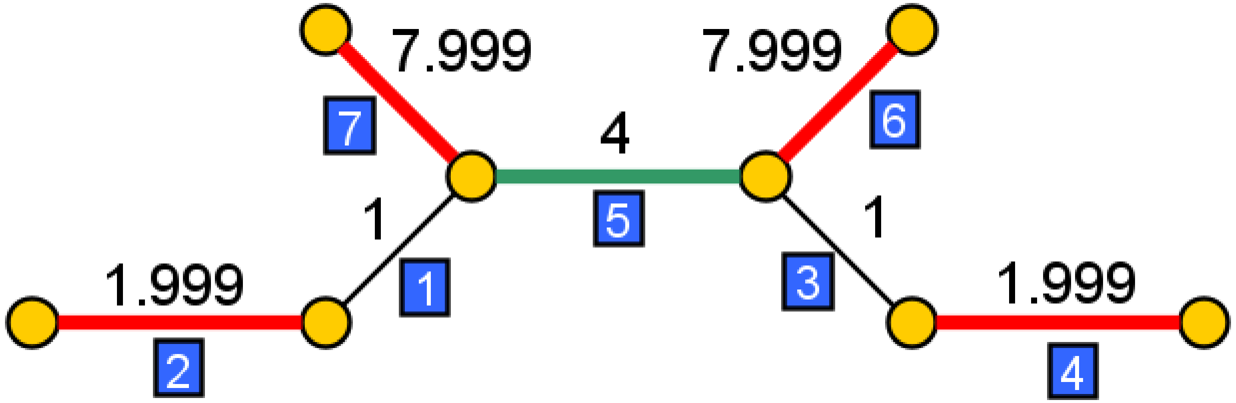
1. M ← ∅
2. While(stream not yet empty)
   a) Let $e = a_1 a_2$ be next edge in stream (we assume that $w(e)$ arrives with e)
   b) Let C be the set of at most two edges of M incident to a_1 or a_2
   c) **if** ($w(e) \geq 2\, w(C)$) **then**
   $M \leftarrow (M \setminus C) \cup \{e\}$
3. **return** M



Analysis:

- Since M is a matching and thus $|M| \leq \frac{n}{2}$, we use space at most $O(n \log n)$ (disregarding the space for storing the weights)
- Approximation factor?

Example



- Maximum matching: $\boldsymbol{M}^*$ has weight almost 20
- Matching $\boldsymbol{M}$ computed by the algorithm has weight 4

$\rightarrow$ Algorithm computes roughly a $\frac{1}{5}$-approximation on this instance

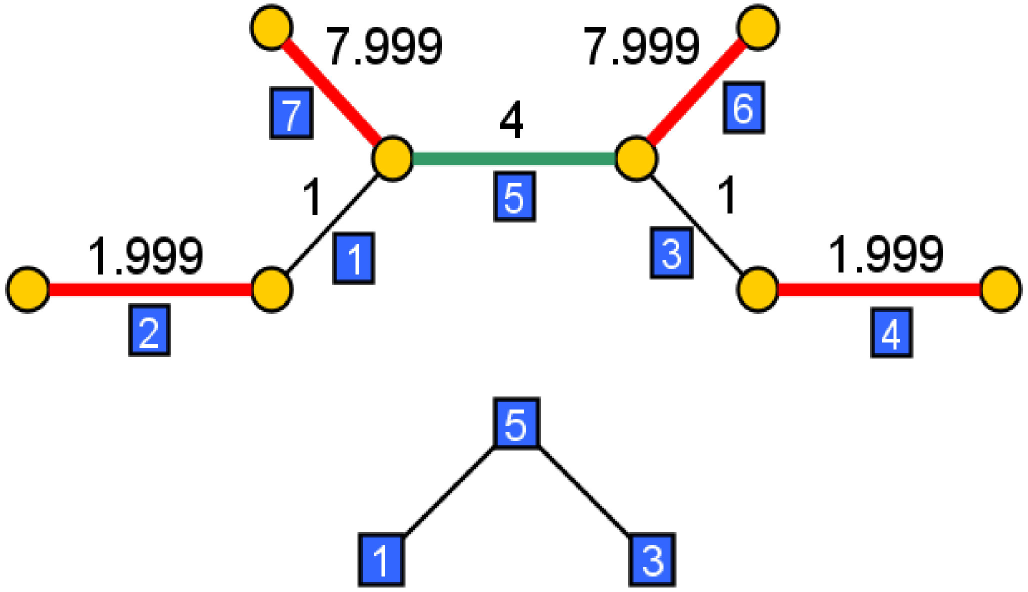
Weighted Matching: Analysis

We say that…:

- Edge e is *born* when it is inserted into M
- Edge e is *killed* by edge f if e is removed from M upon processing f
- Edge e *survives* if it was born and it has never been killed

Analysis:

- Observe that the final matching M is the set of survivors
- For each survivor $e \in M$, build its *killing tree*: e is at the root and each node's descendants are the edges killed by that node. $T(e)$: set of nodes in e's killing tree without e.

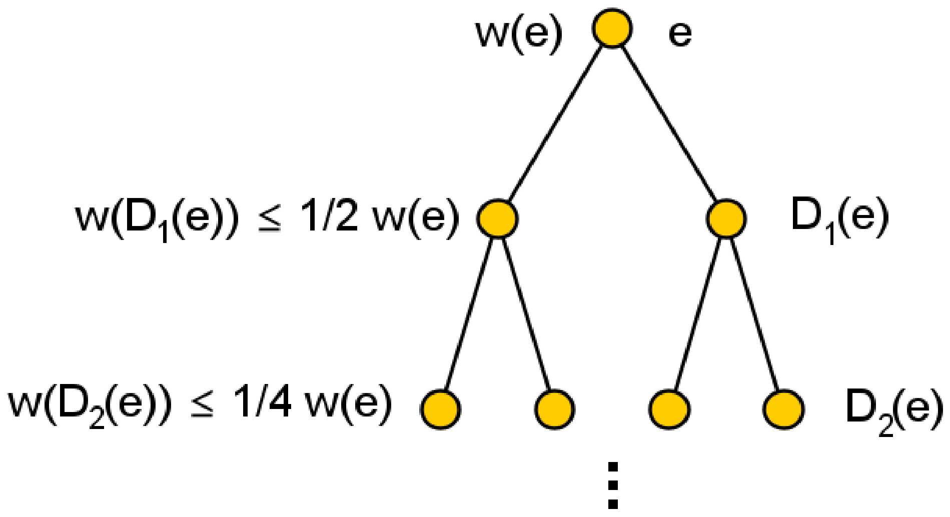


Weighted Matching: Analysis (2)

Lemma. $w\big(T(e)\big) < w(e)$.

Proof.

- By construction of algorithm, we have
$$\sum_{f \text{ child of } e} w(f) \leq \tfrac{1}{2} w(e)$$
- Let $D_i(e)$ be the level i descendants of e in e's killing tree. Then:
$$w\big(D_i(e)\big) \leq \frac{w(e)}{2^i}$$
- Summing up over all levels, we obtain:



$$w\big(T(e)\big) \leq \sum_{i \geq 1} w\big(D_i(e)\big) \leq \sum_{i \geq 1} \frac{1}{2^i} w(e) < w(e)$$

□

Corollary. The total weight of all edges killed is at most $w(M)$, i.e., $w\big(T(M)\big) < w(M)$.

Weighted Matching: Analysis (3)

Relating $\mathbf{w}(\mathrm{M})$ to $\boldsymbol{w}(\boldsymbol{M}^*)$: Charging scheme

- We will maintain the weight of $\boldsymbol{M}^*$ using a *charging scheme*
- For each edge $e = uv$, we define slots $< e, u >$ and $< e, v >$ to maintain charge
- Our scheme will maintain the following invariants:

[CS1] For each vertex $v \in V$, at most one slot of the form $< e, v >$ holds a charge
[CS2] For each slot $< e, v >$, the charge allocated to it is at most $2 \cdot w(e)$

Creating Charge: Charge is only created when an edge $z = uv \in M^*$ arrives in the stream

1. If z is born then $\frac{w(z)}{2}$ is allocated to both $< z, u >$ and $< z, v >$
2. If z is not born because exactly one edge e touches z at vertex u (say), then a charge of $w(z)$ is allocated to $< e, u >$
3. If z is not born because exactly two edges e, f touch z at vertices u, v (resp.), then:

$$< e, u > = \frac{w(z)w(e)}{w(e) + w(f)} \ and < f, v > = \frac{w(z)w(f)}{w(e) + w(f)}.$$

Weighted Matching: Analysis (4)

Charge Transfer:

When an edge $e = uv \ \in E \ \setminus M^*$ arrives in the stream, we may reallocate charge:

- If e is not born, nothing happens
- If e is born, any charge associated to u is transferred to e (similarly for v)

Observe: Invariants are maintained throughout charging scheme!

Theorem. The approximation factor of the weighted matching algorithm is $\mathbf{1/6}$.

Proof.

- Total Charge: $w(M^*)$
- Each charge is allocated with an edge that is born and is therefore contained in a killing tree
- Surviving edge $e = uv \in M$: total charge $< e, u > \ + < e, v > \leq 4\ w(e)$ (by **[CS2]**)
- Killed edge, i.e., edge $f \ \in T(e)$, for some survivor e: at most one endpoint is charged since f was killed and charge on other endpoint was transferred to killer: total charge $2\ w(f)$ (by **[CS2]**)

$$w(M^*) \leq 2\left(w\big(T(M)\big) + 2w(M)\right) \leq 2\big(w(M) + 2w(M)\big) = 6\ w(M),$$

Using corollary on page 11.

□

Summary

Summary and References:

- The weighted matching algorithm presented here is due to:

Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, Jian Zhang: "On Graph Problems in a Semi-streaming Model." ICALP 2004: 531-543

- We now know how to compute an almost $\frac{1}{2}$-approximation in the semi-streaming model for weighted matching, which is due to

Ami Paz, Gregory Schwartzman: "A $(2 + \epsilon)$-Approximation for Maximum Weight Matching in the Semi-Streaming Model". SODA 2017: 2153-2161