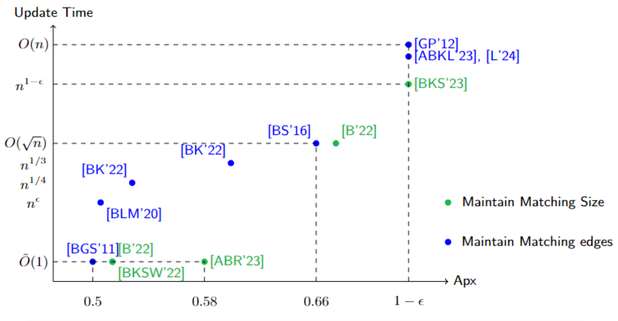
Fully Dynamic Matching and Ordered Ruzsa-Szemerédi Graphs

Soheil Behnezhad, Alma Ghafari
Northeastern University

Abstract

We study the **fully dynamic maximum matching** problem. The goal is to efficiently maintain an approximate maximum matching of a graph that is subject to **edge insertions and deletions**. We present an algorithms that maintain the edges of a $(1-\varepsilon)$-approximate maximum matching for an arbitrarily small constant $\varepsilon > 0$. The update time of our algorithm is parametrized by the density of **Ordered Ruzsa-Szemerédi** (ORS) graphs, a generalization of the Ruzsa-Szemerédi graphs. While determining the density of ORS (or RS) remains a hard problem in combinatorics, we prove that if the existing constructions of ORS graphs are optimal, then our algorithm runs in $n^{0.5+\varepsilon}$ time for any fixed $\varepsilon > 0$ which would be significantly faster than existing near-linear in n time algorithms.

Previous Work



Open Problem:

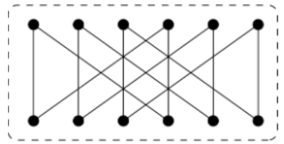
Is it possible to maintain a $(1-\varepsilon)$-approximate maximum matching in a fully dynamic graph, for any fixed $\varepsilon > 0$, in $n^{1-\Omega(1)}$ update time?

Our Contributions

A New Dynamic Matching Algorithm:

We present a new algorithm whose update time depends on the density of Ordered Ruzsa Szemeredi **(ORS)** graphs, a generalization of the well-known RuzsaSzemeredi **(RS)** graphs.

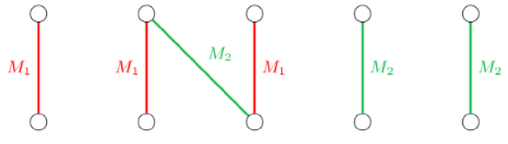
Definition: An n-vertex graph $G = (V, E)$ is an $RS_n(r,t)$ graph if its edge-set E can be decomposed into t edge-disjoint induced matchings each of size r. We use $RS_n(r)$ to denote the maximum t for which $RS_n(r,t)$ graphs exists.



Instead of each matching being an induced matching in the whole graph, the edges of an ORS graph should be decomposed into an ordered list of matchings such that each matching is induced only with respect to the previous matchings in the ordering. Every $RS_n(r,t)$ graph is an $ORS_n(r,t)$ graph but the reverse is not necessarily true.

Definition:

An n-vertex graph $G = (V, E)$ is an $ORS_n(r,t)$ graph if its edge-set E can be decomposed into an ordered list of t edge-disjoint matchings $M_1, \dots, M_t$ each of size r such that for every $i \in [t]$, matching M_i is an induced matching in $M_1 \cup \cdots \cup M_i$. We use $ORS_n(r)$ to denote the maximum t for which $ORS_n(r,t)$ graphs exists.



Result 1:

Let $\varepsilon > 0$ be fixed. There is a randomized fully dynamic algorithm that maintains the edges of a $(1-\varepsilon)$- approximate maximum matching in $\tilde{O}\left(\sqrt{n^{1+o(1)} \cdot ORS_n(f(\varepsilon)\cdot n)}\right)$ amortized update-time. The algorithm works against adaptive adversaries.

Better Upper Bounds for ORS:

Let $0 < c < 1/5$ be a constant. Since $ORS_n(cn) \le RS_n(cn)$ as every **RS** graph is also an **ORS** graph with the same parameters, it is natural to first look into the more well-studied case of **RS** graphs. Despite all the attention to **RS** graphs such as in **property testing** and **streaming algorithm**, The value of $RS_n(cn)$ remains widely unknown.

$n^{\Omega_c\left(\frac{1}{\log\log n}\right)} = n^{o(1)} \le RS_n(cn)$	[Fischer, Lehman, Newman, Raskhodnikova Rubinfeld, and Samorodnitsky 02]
$RS_n(cn) \le n/\log^{O\left(log\left(\frac{1}{c}\right)\right)} n$	[Fox11]

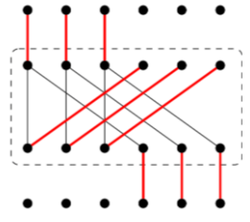
Result 2:

For any $c > 0$, it holds that $ORS_n(cn) \le n/\log^{\text{Poly}\left(\frac{1}{c}\right)} n$.

A Hard Example

Let $G = (V, E)$ be an ORS graph with matchings $M_1, \dots, M_t$.

- Insert graph G.
- For $i = t$ to 1:
  1. Connect the Complement nodes of M_i to outside.
  2. Delete the added edges.
  3. Delete the edges of M_i.



Tools

- We can assume without loss of generality that maximum matching size is $\Omega(n)$. Using this assumption we find the matching once, do nothing for the next εn updates. [AssadiKhannaLi 16] [Kiss 22]
- Finding $(1-\varepsilon)$-approximation on G reduces to finding $O(1)$-approximation on an adaptively chosen subgraph of G. [McGregor 05] [AhnGuha11] [AssadiLiuTarjan 21]

Algorithm

In each step we are given a subgraph U.

Lemma:

Given U with **average degree** d there is an algorithm that using **random sampling** takes $O\left(\frac{n^2}{d}\right)$ time to find an approximate matching.

We guarantee that if our algorithm takes $\Omega(n^2)$ time to solve an instance $G[U]$, then the maximum matching in $G[U]$ must be an induced matching of the graph G, and **charge heavy computations to ORS**.

Problem 1: It might be that this matching is not induced yet it is sparse enough that is hard to find.

Lemma:

Let $M_1, \dots, M_t$ be the matchings that we charge and let d_i be the average degree of the i-th matching and suppose that these matchings are edge-disjoint. We show in our update-time analysis that $\sum_{i=1}^{t} \frac{1}{d_i}$ can be at most $ORS_n(f(\varepsilon)\cdot n)$, and therefore the total time spent by our algorithm can be upper bounded by $O\left(n^2 ORS_n(f(\varepsilon)\cdot n)\right)$.

Problem 2: We need to ensure that the matchings that we charge are **edge-disjoint**.

We maintain a set S and add all edges of any matching that we charge to this set.

Before solving $G[U]$, we first go over the edges in S and see whether they can be used to find a large matching in $G[U]$.

- If they do, we do not run the random sampling algorithm
- If not, the matching that we find must be edge-disjoint.

We reset S after $\tilde{O}\left(n\sqrt{n^{1+o(1)} \cdot ORS_n(f(\varepsilon)\cdot n)}\right)$ updates to avoid it being dense. Since we output a matching after εn updates, the amortized running time is

$$\frac{s}{\varepsilon n} + \frac{O\left(n^2 ORS_n(f(\varepsilon)\cdot n)\right)}{s} = \tilde{O}\left(\sqrt{n^{1+o(1)} \cdot ORS_n(f(\varepsilon)\cdot n)}\right).$$