These lecture notes have not undergone rigorous peer-review. Please email quanquan.liu@yale.edu if you see any errors.

1 Introduction

Today we'll discuss solving the densest subgraph problem with the primal/dual method using the multiplicative weight update (MWU) method. Last time, we discussed how to solve packing and covering LPs using MWU assuming an oracle on that solves a convex combination of the weights in the weights in the LP. Today, we'll instantiate this method for the densest subgraph problem. We'll be discussing the MWU algorithm given in [BGM14].

Let us first start with the primal and dual formulation of MWU given in [Cha00].

(PRIMAL) Maximum Density Subgraph

$$
\begin{aligned}
\text{maximize} \quad & \sum_{e \in E} x_e \\
\text{subject to} \quad & \sum_{v \in V} y_v = 1 \\
& x_e \leq y_u, x_e \leq y_v \qquad \forall e = \{u, v\} \in E \\
& y_v, x_e \geq 0 \qquad \forall e \in E, v \in V
\end{aligned}
$$

(DUAL) Lowest Out-Degree Orientation

$$
\begin{aligned}
\text{minimize} \quad & B \\
\text{subject to} \quad & \alpha_{eu} + \alpha_{ev} \geq 1 \qquad \forall e = \{u, v\} \in E \\
& \sum_{e \ni u} \alpha_{eu} \leq B \qquad \forall u \in V \text{ where } e \in E \\
& \alpha_{eu}, \alpha_{ev} \geq 0 \qquad \forall e = \{u, v\} \in E
\end{aligned}
$$

Figure 1: Fig. 1 of [SV20]: Linear programs for densest subgraph (primal) and fractional lowest out-degree orientation (dual).

The intuition for the primal and dual are as follows. First, for the primal, if $x_e = 1$, then y_u, y_v both have to be 1. Charikar showed that the optimum solution to the primal is at least the value of the densest subgraph in the input graph. This means that whenever an edge x_e is in the densest subgraph, then both of its endpoints must also be in the densest subgraph. Now, we interpret the objective. First, the densest subgraph objective maximizes $\frac{\sum_{e \in E} x_e}{\sum_{v \in V} y_v}$ when we don't have any constraints on the values for y_v. Then, we can scale the y_v values such that $\sum_{v \in V} y_v = 1$ and the LP ensures that the x_e values scale accordingly. Scaling the y_v values as such ensures a simpler objective in the primal.

For the dual, we suppose we distribute weight of 1 on every edge. Then, the constraint $\alpha_{eu} + \alpha_{ev} \geq 1$ ensures that the weight of 1 on the edge is distributed among its endpoints. Then, we seek to minimize the maximum weight distributed onto any vertex. Hence, this problem is called the minimum fractional edge orientation problem since we can consider distributing weights as fractionally orienting the edges and the problem solves for the minimum such orientation.

Let OPT be the optimum solution to the primal. Then, by primal-dual, the following lemma is true.

Lemma 1.1. *The dual with B set to z is feasible if and only if $z \geq$ OPT.*

2 MWU for Densest Subgraphs

We now apply MWU to the *dual*. Recall that our application of MWU first seeks to find a convex region on some of the constraints. Note that the number of "experts" of our MWU is m. We'll define our convex region K as follows:

$$K = \left\{ \alpha_{eu} \geq 0 \mid z = \text{OPT}, \sum_{e \ni u} \alpha_{eu} \leq z \right\}.$$

Now, we check for feasibility of the constraints $\alpha_{eu} + \alpha_{ev} \geq 1$. But first, let's consider the width of the problem before we set our convex region in stone. Naively, each α_{ev} can be as large as OPT. This means that $\alpha_{ev} + \alpha_{eu} \leq 2\text{OPT}$, i.e. our loss per day is as much as 2OPT! This means that our width $\rho = \Theta(\text{OPT})$. Since OPT can be as large as n, we have that $\rho = \Theta(n)$. This is not good for distributed, parallel, any scalable applications!

Luckily, [BGM14] performs a *width reduction* to modify the original polyhedral K such that we have bounded width. Suppose we add an additional constraint to ensure that $\alpha_{ev} \leq q$ for some small constant $q \geq 1$.

$$K = \left\{ \alpha_{eu} \geq 0 \mid z = \text{OPT}, \sum_{e \ni u} \alpha_{eu} \leq z, \alpha_{eu} \leq q \right\}.$$

Then, we can show that the dual is feasible for z if and only if the dual is feasible with this new constraint. We let $\text{DUAL}(z)$ denote the feasibility program when B is set to z and $\text{DUAL}(z, q)$ to be the feasibility program when B is set to z and we add the constraints $\alpha_{ev} \leq q$. This is true since we do not set α_{eu} to be larger than necessary to satisfy the constraint $\alpha_{eu} + \alpha_{ev} \geq 1$. Hence, we do not every set any α_{eu} to be greater than 1.

Now, what is the weight of the new system? The width is now at most $2q$! Since $q = O(1)$; we have now reduced the weight to $O\left(\frac{\log m}{\varepsilon^2}\right)$ for constant ε.

We've reduced the width so let's proceed with our new K. We now define our oracle. Recall that we want our oracle to solve a convex combination of our constraints $\alpha_{eu} + \alpha_{ev} \geq 1$. The easiest convex combination is to sum up all of these constraints. The left hand size of the sum simplifies to $\sum_v \sum_{e \text{ incident to } v} \alpha_{ev}$. We now multiply both sides by $\mathbf{p}$. Hence, we obtain the following constraint

$$\sum_v \sum_{e \text{ incident to } v} p_e \alpha_{ev} \geq ||\mathbf{p}||_1.$$

If the above constraint is not satisfied then $\text{DUAL}(z, q)$ is infeasible. Hence, our oracle $\text{ORACLE}(\mathbf{p})$ just needs to return $\alpha \in K$ that computes:

$$C(\mathbf{p}, z, q) = \max_{\alpha \in K} \sum_v \sum_{e \text{ incident to } v} p_e \alpha_{ev}$$

We output *infeasible* if the above value is less than $||\mathbf{p}||_1$.

Otherwise, we return $\frac{1}{T} \sum_{t \in [T]} \alpha^t$ as the feasible solution. We now prove the existence of a linear time oracle.

Lemma 2.1. $\text{ORACLE}(\mathbf{p})$ *can be computed in* $O(m)$ *time.*

Proof. For any $\mathbf{p}$, for each v, the optimum solution $C(\mathbf{p}, z, q)$ sets α_{ev} as follows. Let $r = \lfloor \frac{z}{q} \rfloor$ and let $s = z - r \cdot q$. Then, set $\alpha_{ev} = q$ for the r largest p_e incident to v and $\alpha_{ev} = s$ for the $(r+1)$-st largest p_e. We can find and set these values for each vertex in $O(m)$ time. □

Thus, in $O\left(\frac{m \log m}{\varepsilon^2}\right)$ time for constant q, our MWU algorithm returns infeasible for the Dual or finds α where $\alpha_{eu} + \alpha_{ev} \geq 1 - \varepsilon$ for all constraints.

Finally, we use primal-dual to find a solution to the primal. Suppose we pass $\mathbf{x}$ for $\mathbf{p}$ in $C(\mathbf{p}, z, q)$. We show the following theorem about the $\mathbf{x}$ returned by ORACLE($\mathbf{x}$).

Theorem 2.2. *Let $\tilde{D}$ be the smallest value found via binary searching $(1+\varepsilon)^i$ values for $i \in [\log(n)]$ where our MWU procedure does not return infeasible. Then, for any $\varepsilon \in (0, 1/3)$ and any constant $q \geq 1$, we satisfy:*

1. *the MWU does not return infeasible and returns an α where $\tilde{D}$ is guaranteed to be in $\mathrm{OPT}(1-\varepsilon) \leq \tilde{D} \leq D(1+\varepsilon)$,*
2. *the MWU returns an $\mathbf{x}$ where $\sum_{e \in E} x_e \geq (1 - 3\varepsilon) \cdot C(\mathbf{x}, \tilde{D}, q)$.*

Proof. First, suppose that $\tilde{D} < D^* \cdot (1-\varepsilon)$ and the MWU does not return infeasible. Since MWU guarantees that the α values satisfy $\alpha_{eu} + \alpha_{ev} \geq 1 - \varepsilon$. This means that scaling up $\tilde{D}$ by $\frac{1}{1-\varepsilon}$ results in a feasible solution for the original dual constraints. Thus, we obtained a value $D^*_{new} = \frac{\tilde{D}}{1-\varepsilon} < D^*$ which contradicts the optimality of D^*; hence the MWU must return infeasible in this case. Furthermore, $\tilde{D} \leq (1+\varepsilon) \cdot D^*$ by our binary search procedure and for all values $\tilde{D} \in [D^*, D^*(1+\varepsilon)]$, the MWU will return a feasible solution. Hence, if the MWU procedure does not return infeasible, then, we are guaranteed Item 1.

Then, for $\tilde{D} \leq (1+\varepsilon) \cdot D^*$, it must be the case for

$$\lambda^* = \{\lambda \mid \alpha_{eu} + \alpha_{ev} \geq \lambda \forall e = (u, v), \alpha \in K\},$$

it must hold that $\lambda^* \leq 1 + \varepsilon$. Otherwise, if $\lambda^* > 1 + \varepsilon$, we can scale α by $> 1/(1+\varepsilon)$ and obtain a feasible solution to the dual with $\frac{\tilde{D}}{\lambda^*} < D^*$ which is not possible. Hence, it must hold by the guarantees of MWU that

$$\begin{aligned}
(1+\varepsilon) \cdot \sum_{e \in E} x_e &\geq (1-\varepsilon) \cdot C(\mathbf{x}, \tilde{D}, q) \\
\sum_{e \in E} x_e &\geq (1-\varepsilon) \cdot \frac{C(\mathbf{x}, \tilde{D}, q)}{1+\varepsilon} \\
\sum_{e \in E} x_e &\geq (1-\varepsilon)^2 \cdot C(\mathbf{x}, \tilde{D}, q) \\
\sum_{e \in E} x_e &\geq (1-3\varepsilon) \cdot C(\mathbf{x}, \tilde{D}, q).
\end{aligned}$$

□

Now, we need to find the solution from the primal. However, one major obstacle to finding the solution to the primal is as follows. We used MWU to solve the modified dual problem. Unfortunately, DUAL($\tilde{D}, q$) has a different primal than the original primal. In particular, the primal feasibility problem corresponding to DUAL($\tilde{D}, q$) is the following:

$$\frac{\sum_{e \in E} x_e}{\sum_{e \in E, v \in V} \left(\tilde{D} y_v + q z_{ev}\right)} \geq 1$$

$$\text{where } x_e \leq \min\left(y_u + z_{eu}, y_v + z_{ev}\right) \forall e = (u, v) \in E.$$

The primal corresponding with $\text{DUAL}(\tilde{D})$ does not have the $\mathbf{z}$ variables. Although the exact optimal primal solution will set $\mathbf{z} = 0$, this is *not* the case for ε-approximate solutions since ε-approximate solutions can set large $\mathbf{z}$ so that we do not know how to interpret the corresponding $\mathbf{x}, \mathbf{y}$ variables in the context of the original primal.

Now, we show the procedure used in [BGM14] that rounds an ε-approximate solution to $\text{DUAL}(\tilde{D}, q)$ into an ε-approximate solution to $\text{DUAL}(\tilde{D})$. Recall from our oracle that $C(\mathbf{x}, \tilde{D}, q)$ is computed by first sorting, for every vertex v, the values x_e of the edges incident to v in non-increasing order, and assigning q to the first $r = \lfloor \tilde{D}/q \rfloor$ edges in this order and $s = \tilde{D} - q \cdot r$ to the $(r+1)$-st edge in this order. Let the sorted order be $x_1(v) \geq x_2(v) \geq \cdots \geq x_k(v)$. Thus, we can also write our $C(\mathbf{x}, \tilde{D}, q)$ as follows:

$$C(\mathbf{x}, \tilde{D}, q) = \sum_{v \in V} \left(s \cdot x_{r+1}(v) + \sum_{i=1}^{r} q x_i(v) \right). \tag{2.1}$$

Using this formulation, we now prove the following two steps of our rounding algorithm.

Discretization Discretization exactly does what it sounds, it discretizes the returned x_e values so that we end up with a small number of distinct values of x_e. Let $X = \max_{e \in E} x_e$. Then, we scale up or down the x_e values such that $X = 1$. Now, we consider all edges e with $x_e \leq \varepsilon/m^2$. The sum of x_e of all of these edges contributes at most ε/m to $\sum_{e \in E} x_e$. We set the value of these x_e's to $x_e = 0$. Using Eq. (2.1), the largest $r+1$ values of x_e are multiplied with either r or s. Since we rescaled the x_e values such that $X = 1$, $C(\mathbf{x}, \tilde{D}, q) \geq 2$ with the rescaled $\mathbf{x}$. Since, we originally have that $\sum_{e \in E} x_e \geq (1 - 3\varepsilon) C(\mathbf{x}, \tilde{D}, q)$, our new rescaled vector with all values of $x_e \leq \varepsilon/m^2$ set to 0 satisfies:

$$\sum_{e \in E} x_e \geq (1 - 3\varepsilon) C(\mathbf{x}, \tilde{D}, q) - \varepsilon/m$$
$$\sum_{e \in E} x_e \geq (1 - 4\varepsilon) C(\mathbf{x}, \tilde{D}, q).$$

Now, we round each rescaled x_e down to the nearest power of $(1+\varepsilon)$, which does not change the value of x_e by more than a factor of $(1+\varepsilon)$. This means that the resulting $\mathbf{x}$ satisfies

$$\sum_{e \in E} x_e \geq \frac{(1 - 4\varepsilon) C(\mathbf{x}, \tilde{D}, q)}{1+\varepsilon}$$
$$\sum_{e \in E} x_e \geq (1 - 6\varepsilon) C(\mathbf{x}, \tilde{D}, q).$$

Now that we have rounded every value of x_e to the nearest power of $(1+\varepsilon)$ and we are guaranteed that $x_e \geq \frac{\varepsilon}{m^2}$, there are now only $O\left(\frac{\log m}{\varepsilon}\right)$ distinct values of x_e.

Line Sweep Using our values of x_e, we will now find an approximate densest subgraph (subset of vertices whose induced subgraph is an approximate densest subgraph). Recall that intuitively the x_e values tells us which edges are in the densest subgraph. Hence, approximately optimal x_e values should tell us the approximate densest subgraph.

Fix a $\gamma \geq 0$. Let $I(x_e) = 1$ if $x_e \geq \gamma$. Let $G(\gamma)$ be the subgraph induced by the set of edges $x_e \geq \gamma$. Let $E(\gamma)$ be the set of such induced edges and $V(\gamma)$ denote the set of the endpoints of these edges. Let

$d_v(\gamma)$ be the degree of v in $G(\gamma)$. Then, $d_v(\gamma) = \sum_{e \in N(v)} I(x_e)$ and $|E(\gamma)| = \sum_{e \in E} I(x_e)$. Finally, let

$$H_v(\gamma) = \sum_{v \in V} \left(\frac{s}{q} \cdot I(x_{r+1}(v)) + \sum_{i=1}^{r} I(x_i(v)) \right). \tag{2.2}$$

We now show the following lemma.

Lemma 2.3. *There exists a γ where $G(\gamma) \neq$ and $|E(\gamma)| \geq q(1-6\varepsilon)\sum_{v \in V} H_v(\gamma)$. Furthermore, this value of γ can be computed in $O\left(\frac{m \log m}{\varepsilon}\right)$ time.*

Proof. First, observe that

$$\sum_{e \in E} x_e = \int_0^1 |E(\gamma)| d\gamma$$

and

$$C(\mathbf{x}, \tilde{D}, q) = q \int_0^1 \sum_{v \in V} H_v(\gamma) d\gamma,$$

by definition of our variables. Since $\sum_{e \in E} x_e \geq (1-6\varepsilon)C(\mathbf{x}, \tilde{D}, q)$, there must exist a γ where $|E(\gamma)| \geq q(1-6\varepsilon)\sum_{v \in V} H_v(\gamma)$. Since there are only $O\left(\frac{\log m}{\varepsilon}\right)$ distinct values of γ, we can search through all possible values of γ in $O(m)$ per value of γ and compute the associated values to determine whether $|E(\gamma)| \geq q(1-6\varepsilon)\sum_{v \in V} H_v(\gamma)$. □

We now prove our final theorem.

Theorem 2.4. *For any $\varepsilon \in (0, 1/12)$, a subgraph of density at least $(1-\varepsilon)D^*$ can be computed in $O\left(\frac{m \log m}{\varepsilon^2}\right)$ time.*

Proof. By Lemma 2.3, there exists a value γ (which we can find in polynomial time) that satisfies $|E(\gamma)| \geq q(1-6\varepsilon)\sum_{v \in V} H_v(\gamma)$. Fix γ to be such a γ. Let V_1 be the set of vertices where for every $v \in V_1$, we have $x_{r+1}(v) \geq \gamma$. For these vertices $v \in V_1$, we have $H_v(\gamma) = \frac{s}{q} + \sum_{i=1}^{r} 1 = r + \frac{s}{q} = \frac{\tilde{D}}{q}$. Let V_2 denote $V \setminus V_1$ and we have $d_w(\gamma) = H_w(\gamma)$ for $w \in V_2$; this is true since $I(x_{r+1}(w)) = 0$ for these vertices.

Thus, we can show the following

$$\sum_{v \in V} H_v(\gamma) = \sum_{v \in V_1} H_v(\gamma) + \sum_{v \in V_2} H_v(\gamma) = \frac{\tilde{D}}{q} \cdot |V_1| + \sum_{v \in V_2} d_v(\gamma).$$

Suppose we consider the induced subgraph consisting of V_1. Then, by using the property $|E(\gamma)| \geq q(1-6\varepsilon)\sum_{v \in V} H_v(\gamma)$,

$$|E_1| \geq |E(\gamma)| - \sum_{v \in V_2} d_v(\gamma) \geq q(1-6\varepsilon)\left(\frac{\tilde{D}}{q} \cdot |V_1| + \sum_{v \in V_2} d_v(\gamma)\right) - \sum_{v \in V_2} d_v(\gamma).$$

Since $q(1-6\varepsilon)X \geq X$ when $q \geq 2$ and $\varepsilon < 1/12$, it holds that $|E_1| \geq (1-6\varepsilon)\tilde{D}|V_1|$ and so we get a $(1-7\varepsilon)$-approximate densest subgraph by binary searching for $\tilde{D}$.

Now, the final piece is to show that V_1 is non-empty. Suppose first that $\sum_{v \in V_2} d_v(\gamma) = 0$, then $|E_1| \geq |E(\gamma)| > 0$ and V_1 is non-empty. Otherwise, the final inequality is strict, and we have that $|E_1| > 0$ which implies that V_1 is non-empty. Hence the density is at least $\tilde{D}(1-6\varepsilon) \geq (1-7\varepsilon)D^*$. □

Note that the above proof *requires* $q > 1 + 6\varepsilon$; otherwise, an approximate solution is not guaranteed.

[SV20] gives a different interpretation of this algorithm and provides the complete analysis *without using MWU as a blackbox*. Please refer to their paper for this alternative analysis.

References

[BGM14] Bahman Bahmani, Ashish Goel, and Kamesh Munagala. Efficient primal-dual graph algorithms for mapreduce. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 59–78. Springer, 2014.

[Cha00] Moses Charikar. Greedy approximation algorithms for finding dense components in a graph. In *International workshop on approximation algorithms for combinatorial optimization*, pages 84–95. Springer, 2000.

[SV20] Hsin-Hao Su and Hoa T. Vu. Distributed Dense Subgraph Detection and Low Outdegree Orientation. In *34th International Symposium on Distributed Computing*, pages 15:1–15:18, 2020.