Notes on Finding Densest Subgraphs via Convex Programming*

Mauro Sozio[†1]

[1]Telecom ParisTech, Paris, France

February 21, 2017

1 Background and Preliminaries

1.1 Densest Subgraph

Definition 1.1 *(Densest Subgraph Problem) Given an undirected graph $G = (V, E)$, we wish to find a subgraph $H = (V_H, E_H)$ with maximum density, where the density $\rho(H)$ of H is defined as $\rho(H) = \frac{|E_H|}{|V_H|}$.*

A set $S \subseteq V$ of nodes is called *densest subset* if it induces a subgraph with maximum density. We denote with $E(S)$ the set of edges uv such that both u and v are in S.

1.2 LP Duality

Linear programs are problems that can be expressed as:

$$\begin{array}{lrl} (\mathsf{LP}) & \max & c^T x, \\ \text{s.t.} & Ax & \leq b, \\ & x & \geq 0. \end{array}$$

The dual of a linear program can be formulated as:

$$\begin{array}{lrl} (\mathsf{DP}) & \min & b^T y, \\ \text{s.t.} & A^T y & \geq c, \\ & y & \geq 0. \end{array}$$

The following two theorems play an important role in linear programming and approximation algorithms.

*These notes have been adapted from [3]

†sozio@telecom-paristech.fr

Theorem 1.2 *(Weak-duality) Let $\bar{x}$ and $\bar{y}$ be two feasible solutions for* LP *and* DP*, respectively. Then*

$$c^T \bar{x} \leq b^T \bar{y}.$$

Theorem 1.3 *(Strong-duality) Let $\bar{x}$ and $\bar{y}$ be two optimal solutions for* LP *and* DP *(if they exist), respectively. Then*

$$c^T \bar{x} = b^T \bar{y}.$$

More information about the role of linear programming in approximation algorithms can be found in [4].

2 Algorithms

The main idea of our algorithms is to compute (or approximate) a maximal density vector r^G, from which the densest subgraph can be recovered by sorting the nodes in non-increasing order on the coordinates of r^G.

Auxiliary Vector. We give a useful view on the density vector. We can imagine that each edge $e \in E$ distributes its weight 1 among the nodes contained in e. This can be captured by an auxiliary vector α in $\mathcal{D}(G) := \{\alpha \in \mathbb{R}_+^{\sum_{e \in E} |e|} : \forall e \in E, \sum_{u \in e} \alpha_u^e = 1\}$, and in particular, α_u^e is the weight received by node u from edge e. Indeed, our algorithms maintain such an auxiliary vector and enforce the following invariant relating the density vector $r \in \mathbb{R}^V$ with the auxiliary vector in $\alpha \in \mathcal{D}(G)$.

Invariant Pair. We say that $(r \in \mathbb{R}_+^V, \alpha \in \mathcal{D}(G))$ is an invariant pair, if for all $u \in V$, $r(u) = \sum_{e \in E: u \in e} \alpha_u^e$.

We give our subroutines and their intuition in this section. Their analysis will be given in Section 3.

2.1 Frank-Wolfe Based Algorithm

The intuition is that in a densest subset S in graph G, it is possible for each edge $e \in E(S)$ to distribute its weight among its own nodes such that the total weight received by each node in S is exactly the density $\rho(S) = \frac{|E(S)|}{|S|}$. Moreover, in calculating the density of S, edges that are cut by S do not contribute their weights to any node in S.

This suggests a general framework for an iterative algorithm, which maintains the invariant between the density vector $r \in \mathbb{R}_+^V$ and the auxiliary vector $\alpha \in \mathcal{D}(G)$. In each iteration of Algorithm 1, each edge $e \in E$ attempts to distribute its weight 1 towards a node $x \in e$ whose current density $r(x)$ is minimum among the nodes in e. The algorithm is highly parallelizable over work performed per edge and per node.

In Section 3, we will show that Algorithm 1 is in fact a variant of the Frank-Wolfe algorithm described in [2], which converges to an optimum solution of a carefully defined constrained convex program. We will then show how to obtain a densest subgraph from such a solution.

Algorithm 1 Frank-Wolfe Based Algorithm

function FRANK-WOLFE($G = (V, E, w)$, $T \in \mathbb{Z}_+$)
  for each $e = uv$ in E in parallel **do**
    $\alpha_u^{e\,(0)}, \alpha_v^{e\,(0)} \leftarrow \frac{1}{2}$
  for each $u \in V$ in parallel **do**
    $r^{(0)}(u) \leftarrow \sum_{e \in E: u \in e} \alpha_u^{e\,(0)}$
  for each iteration $t = 1, \ldots, T$ **do**
    $\gamma_t \leftarrow \frac{2}{t+2}$
    for each e in E in parallel **do**
      $x \leftarrow \arg\min_{v \in e} r^{(t-1)}(v)$
      for each $u \in e$ **do**
        $\widehat{\alpha}_u^e \leftarrow 1$, if $u = x$ and 0 otherwise.
    $\alpha^{(t)} \leftarrow (1 - \gamma_t) \cdot \alpha^{(t-1)} + \gamma_t \cdot \widehat{\alpha}$
    for each $u \in V$ in parallel **do**
      $r^{(t)}(u) \leftarrow \sum_{e \in E: u \in e} \alpha_u^{e\,(t)}$
  return $(\alpha^{(t)}, r^{(t)})$

3 Analysis

We analyze the subroutines given in Section 2 and prove their correctness. Our invariant pair (r, α), where $r \in \mathbb{R}_+^V$ and $\alpha \in \mathcal{D}(G) := \{\alpha \in \mathbb{R}_+^{\sum_{e \in E} |e|} : \forall e \in E, \sum_{u \in e} \alpha_u^e = w_e\}$, is inspired from the Charikar's LP relaxation for densest subgraphs [1].

$$
\begin{aligned}
\mathsf{LP}(G) \qquad \max & \sum_{e \in E} w_e x_e \\
\text{s.t.} \quad & x_e \le y_u, \quad \forall u \in e \\
& \sum_{u \in V} y_u = 1, \\
& x_e, y_u \ge 0, \quad \forall u \in V, e \in E
\end{aligned}
$$

From which we obtain the dual $\mathsf{DP}(G)$ as follows:

$$
\begin{array}{lll}
\multicolumn{3}{l}{\mathsf{DP}(G)} \\
\min & \rho & \\
\text{s.t.} & \rho \geq \displaystyle\sum_{e:u\in e} \alpha_u^e, & \forall u \in V \\
& \displaystyle\sum_{u\in e} \alpha_u^e \geq 1, & \forall e \in E \\
& \alpha_u^e \geq 0, & \forall u \in e \in E
\end{array}
$$

Observe that the dual $\mathsf{DP}(G)$ can be interpreted as follows. For each edge $e = uv$, α_u^e specifies the fraction of the edge e assigned to u, while α_v^e specifies the fraction of the edge e assigned to v. The quantity $\sum_{e:u\in e} \alpha_u^e$ denotes then the fractional degree of u. Therefore, $\mathsf{DP}(G)$ can be seen as the problem of finding a fractional assignment of edges to nodes so as to minimize the maximum fractional degree of the graph.

Then, LP duality gives the following fact.

Fact 3.1 *Suppose (r, α) is an invariant pair for the graph $G = (V, E, w)$. Then, the maximum coordinate of r gives an upper bound on the maximum density of a subset in V.*

Convex Program. The intuition of our subroutines is that each edge $e \in E$ tries to distribute its weight among the nodes it contains such that the total weights received by the nodes are as even as possible. This suggests that we consider the objective function $Q_G(\alpha) := \sum_{u\in V} r(u)^2$, where (r, α) is an invariant pair, i.e., $r(u) = \sum_{e\in E:u\in e} \alpha_u^e$. We then consider the following convex program $\mathsf{CP}(G) := \min\{Q_G(\alpha) : \alpha \in \mathcal{D}(G)\}$.

We analyze the correctness and the convergence rate of our algorithms in Section 2 by showing the following.

1. If $\alpha \in \mathcal{D}(G)$ is an optimum solution for $\mathsf{CP}(G)$, then we can derive the densest subgraph from the density vector $r \in \mathbb{R}_+^V$ induced from the invariant.
2. Our iterative Algorithm 1 is essentially the variant of Frank-Wolfe algorithm described in [2] applied to $\mathsf{CP}(G)$. Hence, theoretical convergence rates can readily be deduced.

3.1 Properties of an Optimal Solution of $\mathsf{CP}(G)$

The following definition is needed for deriving a densest subgraph from the density vector.

Definition 3.2 (Stable Subset) *A non-empty subset $B \subseteq V$ is stable with respect to the invariant pair $(r \in \mathbb{R}_+^V, \alpha \in \mathcal{D}(G))$, if the following conditions hold.*

(a) For all $u \in B$ and $v \notin B$, $r(u) > r(v)$.
(b) For all $e \in E$ such that e intersects both B and $V \setminus B$, $\alpha_u^e = 0, \forall u \in e \cap B$.

Lemma 3.3 (Stable Subset) *Let α be an optimum solution for $\mathsf{CP}(G)$ and let (r, α) be the corresponding invariant pair. Let $\rho = \max_{u \in V} r(u)$. Then, the set of nodes u with $r(u) = \rho$ is a stable subset.*

Proof: It suffices to prove that if there exists $u, v \in e \in E$ such that $r(u) > r(v)$, then $\alpha_u^e = 0$. Otherwise, there exists $\epsilon > 0$ such that we could decrease α_u^e by ϵ and increase α_v^e by ϵ to strictly decrease the objective function. ■

Lemma 3.4 (Uniform Stable Subset is Densest) *Let α be an optimum solution for $\mathsf{CP}(G)$ and let (r, α) be the corresponding invariant pair. Let $\rho = \max_{u \in V} r(u)$. The stable subset consisting of all nodes u such that $r(u) = \rho$ is a densest subgraph in G and it has density ρ.*

Proof: Since α is feasible, we have that $\sum_{e=uv \in E(S)} \alpha_u^e + \alpha_v^e = |E(S)|$. From the fact that S is stable, it follows that $\sum_{e=uv \in E(S)} \alpha_u^e + \alpha_v^e = \rho \cdot |S|$, which implies that $\rho(S) = \frac{|E(S)|}{|S|} = \rho$. Since the subset S corresponds to a feasible primal solution in $\mathsf{LP}(G)$ with objective value $\rho(S) = \rho$, it follows that ρ is the optimal value for both $\mathsf{LP}(G)$ and $\mathsf{DP}(G)$. Hence, S is a densest subset in G with density ρ.

■

3.2 Frank-Wolfe Algorithm

We summarize the Frank-Wolfe algorithm described in [2], which solves a convex program $\min Q(\alpha)$ subject to $\alpha \in \mathcal{D}$, where Q is twice differentiable and $\mathcal{D}$ is a compact convex set in some Euclidean space. The algorithm is an iterative method similar to gradient descent, where we denote ∇Q as the gradient of Q.

Algorithm 2 Frank-Wolfe Algorithm on function Q and feasible set $\mathcal{D}$

```
1: function FRANK-WOLFE(Q,D)
2:     Set initial α^(0) ∈ D arbitrarily.
3:     for each iteration t = 1, ..., T do
4:         γ_t ← 2/(t+2)
5:         α̂ ← arg min_{α∈D} ⟨α, ∇Q(α^(t−1))⟩
6:         α^(t) ← (1 − γ_t) · α^(t−1) + γ_t · α̂
7:     return α^(t)
```

The following lemma shows that our iterative Algorithm 1 is actually a realization of Algorithm 2 applied to our objective function Q_G and feasible set $\mathcal{D}(G)$.

Lemma 3.5 *Lines (8) to (11) of Algorithm 1 implement line (5) in Algorithm 2.*

Proof: Line (5) in Algorithm 2 solves the following problem. Given $\alpha \in \mathcal{D}(G)$, find $\widehat{\alpha} \in \mathcal{D}$ to minimize $\langle \widehat{\alpha}, \nabla Q(\alpha) \rangle = 2 \sum_{e \in E} \sum_{u \in e} \widehat{\alpha}_u^e \cdot r(u)$, where $r \in \mathbb{R}_+^V$ is induced by α via the invariant.

We can consider each edge $e \in E$ independently because the feasible set $\mathcal{D}(G)$ places a constraint $\sum_{u \in e} \widehat{\alpha}_u^e = 1$ on each edge $e \in E$. Hence, to minimize $\sum_{u \in e} \widehat{\alpha}_u^e \cdot r(u)$, edge e should distribute its weight 1 entirely to a vertex $x \in e$ having minimum $r(x)$. This is achieved precisely by lines (8) to (11) of Algorithm 1. ■

3.3 Rate of Convergence

After establishing that Algorithm 1 is a realization of Algorithm 2 in Lemma 3.5, we can use previous convergence analysis of the Frank-Wolfe algorithm [2].

The convergence rate of Algorithm 2 can be described by a constant $C_Q := \frac{1}{2}\mathsf{Diam}(\mathcal{D})^2 \sup_{\alpha \in \mathcal{D}} \|\nabla^2 Q(\alpha)\|_2$, where $\nabla^2 Q$ is the Hessian and $\|\cdot\|_2$ is the spectral norm of a matrix.

Theorem 3.6 (Convergence Rate of Frank-Wolfe [2]) *Suppose $\alpha^* \in \mathcal{D}$ is an optimal solution. Then, for all $t \geq 1$, $Q(\alpha^{(t)}) - Q(\alpha^*) \leq \frac{2C_Q}{t+2}$.*

Lemmas 3.7 and 3.8 are the results of some straightforward computation that we omit here.

Lemma 3.7 (Bounding C_Q) *For a graph $G = (V, E, w)$ with maximum node degree Δ (where the degree of a node is the number of edges that contain it), we have the corresponding $C_{Q_G} \leq 2\Delta \sum_{e \in E} w_e^2$.*

Lemma 3.8 (Error in r implies Error in Q) *Suppose $\alpha \in \mathcal{D}$ induces r such that $\epsilon := \|r - r^*\|_2$, where r^* is induced by an optimal α^* for $\mathsf{CP}(G)$. Then, $Q(\alpha) - Q(\alpha^*) \geq \epsilon^2$.*

The following corollary concludes the proof of the rate of convergence of Algorithm 1.

Corollary 3.9 (Convergence of Parallel Algorithm.) *Suppose Δ is the maximum degree of a node in G. In Algorithm 1, for $t > \frac{4\Delta|E|}{\epsilon^2}$, we have $\|r^{(t)} - r^*\|_2 \leq \epsilon$, where r^* is induced by an optimal α^* for $\mathsf{CP}(G)$.*

References

[1] M. Charikar. Greedy approximation algorithms for finding dense components in a graph. In *Approximation Algorithms for Combinatorial Optimization*, pages 84–95. Springer, 2000.

[2] M. Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *ICML*, pages 427–435, 2013.

[3] M. S. M. Danisch, T-H. Hubert Chan. Large scale density-friendly graph decomposition via convex programming. In *WWW*, 2017.

[4] V. V. Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.