The Lazy Engineer and Recommendation Subgraphs[*]

Arda Antikacioglu
Department of Mathematics
Carnegie Mellon University
aantikac@andrew.cmu.edu

R. Ravi
Tepper School of Business
Carnegie Mellon University
ravi@cmu.edu

Srinath Sridar
Bloomreach Inc.
srinath@bloomreach.com

s

1. ALGORITHMS FOR RECOMMENDATION SUBGRAPHS

In this section, we analyze the lazy algorithm of choosing any set of c recommendations, the more interesting greedy algorithm which has better theoretical performance guarantees, and a slower matching based solution that constructs a recommendation subgraph out of overlaying several matchings, in that order.

1.1 Fixed Degree Model

While the algorithm was mentioned earlier, we present it more formally below.

```
Data: A bipartite graph $G = (L, R, E)$
Result: A (c,a)-recommendation subgraph $H$
for $u$ in $L$ do
    neighbors ← a random sample of $c$ vertices in $N(u)$;
    for $v$ in neighbors do
        $H \leftarrow H \cup \{(u, v)\}$;
    end
end
return $H$;
```

Algorithm 1: The sampling algorithm

Given a bipartite graph G, the algorithm is obviously linear time since we do $|L|$ iterations of the loop, and a constant amount of work in each iteration. The space complexity is similarly linear, since the only thing we store is H. The following theorem gives a lower bound on the expected solution.
To analyze the this algorithm, we will use random graphs and give a bound on its expected performance when the underlying supergraph $G = (L, R, E)$ is chosen probabilistically as follows. Each vertex $v \in L$ uniformly and independently samples a set of d neighbors from R. This model is similar to, but is distinct from the more commonly known Erdös-Renyi model of random graphs. In particular, while the degree of each vertex in L is fixed under this model, concentration bounds can show that the degrees of the vertices in L would have similarly been concentrated around d under appropriate parameter settings. We now prove the following theorem about the performance of the Sampling Algorithm.

[*]This work was supported in part by an internship at BloomReach Inc.

Theorem 1. *Let S be the random variable denoting the number of vertices $v \in R$ such that $\deg_H(v) \geq a$ in the fixed-degree model. Then*

$$\mathrm{E}[S] \geq r\left(1 - e^{-ck+\frac{a-1}{r}}\frac{(ck)^a - 1}{ck-1}\right)$$

Proof. Let X_{uv} be the indicator variable of the event that the edge uv ($u \in L$, $v \in R$) is in the subgraph that we picked and set $X_v = \sum_{u\in L} X_{uv}$ so that X_v represents the degree of the vertex v in our subgraph. Because our algorithm uniformly subsamples a uniformly random selection of edges, we can assume that H was generated the same way as G but sampled c instead of d edges for each vertex $u \in L$. So X_{uv} is a Bernoulli random variable. Using the bound $\binom{n}{i} \leq n^i$ on binomial coefficients we get,

$$\begin{aligned}\Pr[X_v < a] &= \sum_{i=0}^{a-1}\binom{cl}{i}\left(1-\frac{1}{r}\right)^{cl-i}\left(\frac{1}{r}\right)^i\\ &\leq \sum_{i=0}^{a-1}\left(\frac{cl}{r}\right)^i\left(1-\frac{1}{r}\right)^{cl-i}\\ &\leq \left(1-\frac{1}{r}\right)^{cl-(a-1)}\sum_{i=0}^{a-1}(ck)^i\\ &\leq \left(1-\frac{1}{r}\right)^{cl-(a-1)}\frac{(ck)^a-1}{ck-1}\\ &\leq e^{-ck+\frac{a-1}{r}}\frac{(ck)^a-1}{ck-1}\end{aligned}$$

Letting $Y_v = [X_v \geq a]$, we now see that

$$\mathrm{E}[S] = \mathrm{E}\left[\sum_{v\in R} Y_v\right] \geq r\left(1 - e^{-ck+\frac{a-1}{r}}\frac{(ck)^a-1}{ck-1}\right)$$

□

We can combine this lower bound with a trivial lower bound to obtain an approximation ratio that holds in expectation.

Theorem 2. *The above sampling algorithm gives a $\left(1-\frac{1}{e}\right)$-factor approximation to the $(c,1)$-graph recommendation problem in expectation.*

Proof. The size of the optimal solution is bounded above by both the number of edges in the graph and the number of vertices in R. The former of these is $cl = ckr$ and the latter is r, which shows that the optimal solution size $OPT \leq$

$r \max(ck, 1)$. Therefore, by simple case analysis the approximation ratio in expectation is at least $(1 - \exp(-ck))/\min(ck, 1) \geq 1 - \frac{1}{e}$ $\square$

For the $(c, 1)$-recommendation subgraph problem the approximation obtained by this sampling approach can be much better for certain values of ck. In particular, if $ck > 1$, then the approximation ratio is $1 - \exp(-ck)$, which approaches 1 as $ck \to \infty$. In particular, if $ck = 3$, then the solution will be at least 95% as good as the optimal solution even with our trivial bounds. Similarly, when $ck < 1$, the approximation ratio is $(1 - \exp(-ck))/ck$ which also approaches 1 as $ck \to 0$. In particular, if $ck = 0.1$ then the solution will be at 95% as good as the optimal solution. The case when $ck = 1$ represents the worst case outcome for this model where we only guarantee 63% optimality. Figure 1 shows the approximation ratio as a function of ck.

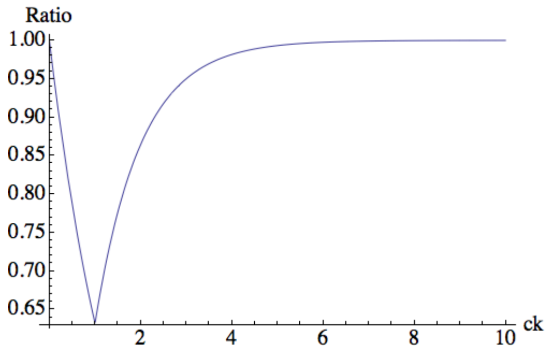

Figure 1: Approx ratio as a function of ck

a	1	2	3	4	5
ck	3.00	4.74	7.05	10.01	13.48

Figure 2: The required ck to obtain 95% optimality for (c, a)-recommendation subgraph

For the general (c, a)-recommendation subgraph problem, if $ck > a$, then the problem is easy on average. This is in comparison to the trivial estimate of cl. For a fixed a, a random solution gets better as ck increases because the decrease in e^{-ck} more than compensates for the polynomial in ck next to it. However, in the more realistic case $ck < a$, we need to use the trivial estimate of ckr/a, and the analysis for $a = 1$ does not extend here. The table in Figure 2 shows how large ck needs to be for the solution to be 95% optimal for different values of a.

We close out this section by showing that the main result that holds in expectation also hold with high probability for $a = 1$, using the following variant of Chernoff bounds.

Theorem 3. *[1] Let $X_1, \ldots, X_n$ be non-positively correlated variables. If $X = \sum_{i=1}^{n} X_i$, then for any $\delta \geq 0$*

$$\Pr[X \geq (1 + \delta)\mathrm{E}[X]] \leq \left(\frac{e^{\delta}}{(1 + \delta)^{1+\delta}}\right)^{E[X]}$$

Theorem 4. *Let S be the random variable denoting the number of vertices $v \in R$ such that $\deg_H(v) \geq 1$. Then $S \leq r(1 - 2\exp(-ck))$ with probability at most $(e/4)^{r(1-\exp(-ck))}$.*

Proof. We can write S as $\sum_{v \in R} 1 - X_v$ where X_v is the indicator variable that denotes that X_v is matched. Note that the variables $1 - X_v$ for each $v \in R$ are non-positively correlated. In particular, if $N(v)$ and $N(v')$ are disjoint, then $1 - X_v$ and $1 - X_{v'}$ are independent. Otherwise, v not claiming any edges can only increase the probability that v' has an edge from any vertex $u \in N(v) \cap N(v')$. Also note that the expected size of S is $r(1 - \exp(-ck))$ by Theorem 1. Therefore, we can apply Theorem 3 with $\delta = 1$ to obtain the result. $\square$

For realistic scenarios where r is very large, this gives very good bounds.

1.2 The Greedy Algorithm

The results in the previous section concentrated on producing nearly optimal solutions in expectation. In this section, we will show that it is possible to obtain good solutions regardless of the model that generated the recommendation subgraph.

We analyze the following natural greedy algorithm for constructing a (c, a)-recommendation subgraph H iteratively.

```
Data: A bipartite graph $G = (L, R, E)$
Result: A (c,a)-recommendation subgraph $H$
for $v$ in $R$ do
    freelinks ← $\{u \in N(v) | useddegree[u] < c\}$;
    if length(freelinks) ≥ $a$ then
        restrict freelinks to $a$ elements;
        for $u$ in freelinks do
            $H \leftarrow H \cup \{(u, v)\}$;
            useddegree[$u$] ← useddegree[$u$]+1;
        end
    end
end
return $H$;
```
Algorithm 2: The greedy Algorithm

The algorithm loops through each vertex in R, and does a constant amount of work in each iteration. Therefore, the runtime is linear. Furthermore, the only data structure we use is an array which keeps track of $\deg_H(u)$ for each $u \in R$, so we only use linear memory as well. Finally, we prove prove the following tight approximation property of this algorithm.

Theorem 5. *The greedy algorithm gives at last a $1/(a+1)$-approximation to the (c, a)-graph recommendation problem.*

Proof. Let $R_{GREEDY}, R_{OPT} \subseteq R$ be the set of vertices that have degree $\geq a$ in the greedy and optimal solutions respectively. Note that any $v \in R_{OPT}$ along with neighbors $\{u_1, \ldots u_a\}$ forms a set of candidate edges that can be used by the greedy algorithm. Each selection of the greedy algorithm might result in some candidates becoming infeasible, but it can continue as long as the candidate pool is not depleted. Each time the greedy algorithm selects some vertex $v \in R$ with edges to $\{u_1, \ldots, u_a\}$, we remove v from the candidate pool. If any u_i had degree c in the optimal solution, we would also need to remove an arbitrary vertex $v_i \in R$ adjacent to u_i from the optimal solution. Therefore, at each step of the greedy algorithm, we may remove at most $a+1$ vertices from the candidate pool as illustrated in Figure **??**. Since our candidate pool has size OPT, the greedy algorithm can not stop before it has added $OPT/(a+1)$ vertices to the solution. $\square$

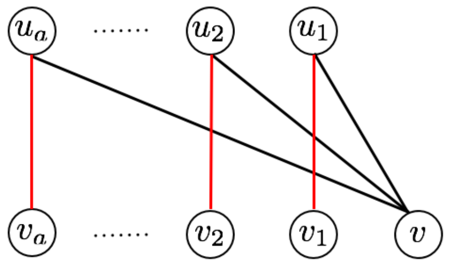

Figure 3: This diagram shows one step of the greedy algorithm. When v selects edges to $u_1,\dots,u_a$, it potentially removes $v_1,\dots,v_a$ from the pool of candidates that are avaiable. The potentially invalidated edges are shown in red.

This approximation guarantee is as good as we can expect, since for $a=1$ we recover the familiar 1/2-approximation of the greedy algorithm for matchings. As in matchings, randomizing the order in which the vertices are processed still leaves a constant factor gap in the quality of the solution [4]. Despite this result, the greedy algorithm fares much better when we give it the same expectation treatment we have given the sampling algorithm. Switching to the Erdös-Renyi model instead of the fixed degree model used in the previous section, we now prove the near optimality of the greedy algorithm for the (c,a)-recommendation subgraph problem.

Theorem 6. *Let $G=(L,R,E)$ be a graph drawn from the $G_{l,r,p}$. If S is the size of the (c,a)-recommendation subgraph produced by the greedy algorithm, then:*

$$\mathrm{E}[S] \geq r - \frac{a(lp)^{a-1}}{(1-p)^a}\sum_{i=0}^{r-1}(1-p)^{l-\frac{ia}{c}}$$

Proof. Note that if edges are generated uniformly, we can consider the graph as being revealed to us one vertex at a time as the greedy algorithm runs. In particular, consider the event X_i that the greedy algorithm matches the $(i+1)^{th}$ vertex it inspects. While, X_{i+1} is dependent on $X_1,\dots,X_i$, the worst condition for X_{i+1} is when all the previous i vertices were from the same vertices in L, which are now not available for matching the $(i+1)^{th}$ vertex. The maximum number of such invalidated vertices is at most $\lceil i/c \rceil$. Therefore, the probability that fewer than a of the at least $l-\lceil i/c \rceil$ available vertices have an edge to this vertex is at most $\Pr[Y\sim Bin(l-\frac{i}{c},p):Y<a]$. We can bound this probability by bounding each term in its binomial expansion by $(1-p)^{l-\frac{ia}{c}-a+1}(lp)^{a-1}$ to obtain the following.

$$\Pr[Y\sim Bin(l-\frac{ia}{c},p):Y<a] \leq a(1-p)^{l-\frac{ia}{c}-a+1}(lp)^{a-1}$$

Summing over all the X_i using the linearity of expectation and this upper bound, we obtain

$$\begin{aligned}\mathrm{E}[S] &\geq r - \sum_{i=0}^{r-1}\mathrm{E}[\neg X_i]\\ &\geq r - \sum_{i=0}^{r-1}\Pr[Y\sim Bin(l-\frac{ia}{c},p):Y<a]\\ &\geq r - a(lp)^{a-1}\sum_{i=0}^{r-1}(1-p)^{l-\frac{ia}{c}-a+1}\end{aligned}$$

$\square$

Asymptotically, this result explains why the greedy algorithm does much better in expectation than $1/(a+1)$ guarantee we can prove in the worst case. In particular, suppose a and c are fixed and that l/r is taken to be a constant as both l and r tend to ∞. In the realm where sublinear error is possible (i.e. when $lc/a > r$) each term in the sum above becomes $\Theta(l^{-\epsilon})$ for some $\epsilon > 0$ if we set $p=\Theta(\log(l)/l)$. Consequently, the error term reduces to $\Theta(l^{1-\epsilon}\log^a(l))$ which is sublinear on the number of vertices.

1.3 Existence of Perfect Recommendation Subgraphs

We define a *perfect* (c,a)-recommendation subgraph on G to be a subgraph H such that $deg_H(u)\leq c$ for all $u\in L$ and $deg_H(v)=a$ for $\min(r,cl/a)$ of the vertices in R. In this section we will prove a sufficient condition for perfect (c,a)-recommendation subgraphs to exist in a bipartite graph G under the Erdös-Renyi model [2] where edges are sampled uniformly and independently with probability p. We use the algorithm we propose to prove this condition as a candidate to compare against random choice and greedy in our tests. Our result relies on the following characterization of perfect matchings.

Theorem 7. *[3] Let G be a bipartite graph drawn from $G_{n,n,p}$. If $p\geq\frac{\log n-\log\log n}{n}$, then as $\lim_{n\to\infty}$ probability that G has a perfect matching approaches 1.*

We will prove that a perfect (c,a)-recommendation subgraph exists in random graphs with high probability by building it up from a matchings each of which must exist with high probability if p is sufficiently high. In particular, we show that p only needs to be $\Omega(\frac{\log n}{n})$ for this to succeed.

Theorem 8. *Let G be a random graph drawn from $G_{l,r,p}$ with $p\geq a\frac{\log l-\log\log l}{l}$ and $kc\geq a$, then the probability that G has a perfect (c,a)-recommendation subgraph tends to 1 as $l,r\to\infty$.*

Proof. Given the size and the degree constraints of L, at most lc/a vertices in R can have degree a in a (c,a)-recommendation subgraph. We therefore restrict R to an arbitrary subset R' of size lc/a. Next, we pick an enumeration of the vertices in $R'=\{v_0,\dots,v_{lc/a-1}\}$ and add each of these vertices into a subsets as follows. Define $R_i=\{v_{(i-1)l/a},\dots,v_{(i-1)l/a+l-1}\}$ for each $1\leq i\leq c$ where the arithmetic in the indices is done modulo lc/a. Note both L and all of the R_i's have size l.

Using these new sets we define the graphs G_i on the bipartitions (L,R_i). Since the sets R_i are intersecting, we cannot define the graphs G_i to be induced subgraphs. However, note that each vertex $v\in R'$ falls into exactly a of these subsets. Therefore, we can uniformly randomly assign each edge in G to one of a graphs among $\{G_1, \ldots, G_c\}$ it can fall into, and make each of those graphs a random graph. In fact, while the different G_i are coupled, taken in isolation we can consider any single G_i to be drawn from the distribution $G_{l,l,p/a}$ since G was drawn from $G_{l,r,p}$. Since $p/a \geq (\log l - \log\log l)/l$ by assumption, we conclude by Theorem 7, the probability that a particular G_i has no perfect matching is $o(1)$.

Considering c to be fixed, by a union bound, we then conclude that except for a $o(1)$ probability, each one of the G_i's has a perfect matching. By superimposing all of these perfect matchings, we can see that every vertex in R' has degree a. Since each vertex in L is in exactly c matchings, each vertex in L has degree c. It follows that except for a $o(1)$ probability there exists a (c, a)-recommendation subgraph in G. □

1.4 Approximation Algorithm Using Perfect Matchings

The above result now enables us to design a near linear time algorithm with a $(1 - \epsilon)$ approximation guarantee to the (c, a)-recommendation subgraph problem by leveraging combinatorial properties of matchings. We call this method the Partition Algorithm, and we outline it below

```
Data: A bipartite graph $G = (L, R, E)$
Result: A (c,a)-recommendation subgraph $H$
$R' \leftarrow$ a random sample of $|L|c/a$ vertices from $R$;
Choose $G[L, R_1], \ldots, G[L, R_c]$ as in Theorem 8;
for $i$ in [1..n] do
    $M_i \leftarrow$ A matching of $G[L, R_i]$ with no augmenting
    path of length $2c/\epsilon$;
end
$H \leftarrow M_1 \bigcup \ldots \bigcup M_c$;
return $H$;
```

Algorithm 3: The partition algorithm

Theorem 9. *Let G be drawn from $G_{l,r,p}$ where $p \geq a\frac{\log l - \log\log l}{l}$. Then Algorithm 3 3 finds a $(1-\epsilon)$-approximation in $O(\frac{|E|c^2}{\epsilon})$ time with probability $1 - o(1)$.*

Proof. Using the previous theorem, we know that each of the graphs G_i has a perfect matching with high probability. These perfect matchings can be approximated to a $1 - \epsilon/c$ factor by finding matchings that do not have augmenting paths of length $\geq 2c/\epsilon$ [5]. This can be done for each G_i in $O(|E|c/\epsilon)$ time. Furthermore, the union of unmatched vertices makes up an at most $c(\epsilon/c)$ fraction of R', which proves the claim. □

Depending on the parameters used, this algorithm can either give stellar approximations or mediocre ones. Given the relative difficulty of finding matching in very large scale graphs, we will opt to go for speed more so than accuracy.

2. REFERENCES

[1] Anne Auger and Benjamin Doerr. *Theory of Randomized Search Heuristics: Foundations and Recent Developments.* Series on Theoretical Computer Science. World Scientific Publishing Company, 2011.

[2] Paul Erdös and Albért Rényi. On random graphs, I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

[3] Svante Janson, Tomasz Luczak, and Andrzej Rucinski. *Random Graphs.* Wiley Series in Discrete Mathematics and Optimization. Wiley, 2011.

[4] Richard Karp, Umesh Vazirani, and Vijay Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 352–358. ACM, 1990.

[5] László. Lovász and Michael D. Plummer. *Matching theory.* North-Holland mathematics studies. Akadémiai Kiadó, 1986.