Online Diverse Bipartite b-Matchings

April Niu
Student ID: 260763893

Rohit Vasishta
Student ID: 261152324

Abstract

The bipartite vertex matching problem is one which has been extensively studied across fields of Computer Science, Mathematics and Economics. Variations of the problem attempt to study the relative tradeoffs that may occur while balancing different objectives than the standard "utility" function, in multiple experimental setups. Matching algorithms have been known to result in biased outputs, further reinforcing stereotypes or producing unfair opportunities and outcomes for socially disadvantaged groups (Alasadi et al., 2019). In an attempt to strengthen diversity and equity, we study the Online Bipartite Diverse b-Matching problem, where a set of "users" arrive sequentially, and are matched to a set of "services" (ads, movies, music, etc). We highlight and suggest many aspects of modelling diversity, specifically, the choice of a metric to measure diversity, a constrained optimization program with the above objective, and a set of experimental validation on real-world datasets.

1 Introduction and Motivation

The vertex b-Matching problem has been extensively studied both theoretically and with experimental validation. Our primary motivation for this problem arises from the classical Adwords problem (Mehta, 2013), where in a bipartite graph $G = (L \cup R, E)$, vertices in L have budgets B_L, and edges $(u, v) \in E$ have bids b_{uv}. An arriving vertex $v \in R$ gets matched to a neighbor $u \in L$ having positive budget, costing $u \in L$ the associated weight b_{uv}. In the classical setting, the goal is to maximize the cardinality of the matching within budge constraint. This problem arises in scenarios like sponsored advertisements, recommendation systems, or in the case of matching reviewers to publish-worthy papers (Ahmed et al., 2017). Whilst efficiency is likely the goal, algorithms that solely rely on maximizing the same may result in outputs that are not representative. For example, search engines could have algorithms that are inherently biased towards matching high-paying jobs more to men than women, a behavior that may arise due to existing biases in training data and/or historical patterns (Arceo-Gomez et al., 2023). One way to break out from such bias is to introduce diversity in the underlying algorithms.

Ergo, build matching algorithms that ensure a high level of diversity, specifically b-matching problems. Our main contribution is proposing a constrained version of an online algorithm that maximizes diversity. We provide an empirical result for the algorithm applied to the MovieLens dataset.

2 Related Work

Our primary references (Dickerson et al., 2019; Ahmed et al., 2017), in which they consider the Online Bipartite Diverse Matching problem. We borrow aspects of their model, and use their code [1] as the base for our experiments. They define two sub-modular functions to measure 'diversity', and provide two provably good algorithms to find online diverse. Their work, however, is restricted to the online Bipartite Matching case. We deviate slightly and consider the case of Bipartite b-Matchings, where each service can be matched to multiple users. Further, we also introduce constrains for the number of times a service (and user) can be matched.

(Kalyanasundaram and Pruhs, 2000) provide a deterministic algorithm to ensure online b-matchings in this scenario with a metric measuring the guarantee of providing a service to a consumer. (Bandyapadhyay et al., 2023) look into the problem of finding a socially fair matching in bipartite graphs, and provide a randomized polynomial-time algorithm to find such a matching in edge-weighted complete bipartite graphs.

Of particular relevance to our project is the

[1] https://bitbucket.org/karthikabinav/onlinesubmodularbipartitematching

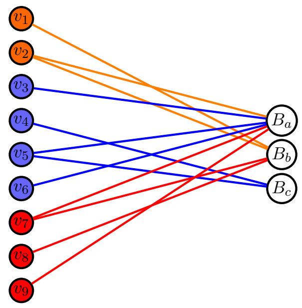

Figure 1: An instance of the diverse matching problem. Users B_a, B_b, B_c are matched to movies of types Red, Blue and Orange. In this instance, user B_a has the most 'diverse' matching, as B_b is not matched to Red movies, and B_c is matched only to Blue movies. It is easy to verify by hand that the Shannon index for B_a is the highest.

work by (Ahmed et al., 2017), which focuses on b-matchings with diversity as an objective, where the problem is shown to be NP-Hard and an offline greedy algorithm is provided.

3 Diversity in Matching

Consider a graph $G = (L \cup R, E)$ where the set of nodes L is divided into groups: $L = \bigcup_{i \in I} A_i$. Then a diverse b-matching is a matching that matches a vertex in R to a set of vertices in L from different groups. The diversity of a matching is captured by submodular functions, which are widely used in economics to model diminishing returns (Lehmann et al., 2001). These functions are also used as objective functions to optimize diversity in the literature (Ahmed et al., 2017; Dickerson et al., 2019) hence we follow the same in our work.

Definition 3.1. Let Ω be a set. Then a set-valued function $f : 2^\Omega \to \mathbb{R}$ is *submodular* if $\forall A \subseteq B \subseteq \Omega$; $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$.

Specifically, in the family of submodular functions we choose the weighted coverage function as the objective function.

Definition 3.2. Consider a collection of finite sets $\{A_i \subseteq U : i \in I\}$. For $J \subseteq I$, let $A_J = \bigcup_{j \in J} A_j$, then define $f(J) = w(A_J)$, as the *weighted coverage function* where $w : U \to \mathbb{R}^+$ is a non-negative weight function.

In the case of coverage function, the function value is maximized when the input set covers as many groups as possible i.e when it is most diverse.

We choose to measure diversity through the Shannon entropy function, a standard metric used in diversity literature (Ahmed et al., 2017; Dickerson et al., 2019). Intuitively, entropy of a state, (or, in this paper, the set of matched vertices), captures how well mixed the state is.

Definition 3.3. The *Shannon Entropy* of a vertex is given by $-\sum_{j \in J}(p_j \log p_j)$, where p_j is the proportion of selected edges in group J.

Entropy of a vertex is maximized when the p_j's are the same across different groups. *The Entropy of a Matching* is then defined as the sum of entropy for all vertices in one side of the bipartition. Thus, higher entropy implies higher diversity of a matching.

To measure the loss in economic efficiency due to diversity, we use the metric *Price of Diversity* (PoD) of a Matching, which is defined as the ratio

$$PoD = \frac{\text{Utility using diversity matching}}{\text{Optimal utility derived}}.$$

Our model assumes a bi-partition of a graph, with a set of services (ex. Movies) on the left and a collection of users arriving one after another on the right.

4 Algorithms

In this section, we present two algorithms that perform online bipartite b-matching. The first has diversity as an objective whereas the second algorithm optimizes social welfare (utility). Both algorithms are greedy as even in the offline case the b-matching problem is NP-Hard. (Ahmed et al., 2017) In the context of online matching, we assume a set of vertices L of size m to be fixed while the vertices in R of size n arrive online. We also have the extra assumptions and constraints that each vertex can be matched at most a certain constant number of times. The constraints are denoted by vector B_l and B_r which represent the constraints on the left side and right side of the matching respectively.

Let M be an $m \times n$ matrix representing a matching between L and R. $M_{uv} = 1$ if there is an edge between vertex $u \in L$ and $v \in R$, and 0 otherwise. Let I_x denote a column vector of size x with all 1's. The following mathematical program models the online bipartite b-matching problem:

$$\begin{aligned} \text{maximize}_M \; & f(M) && (1) \\ \text{subject to } & MI_n \leq B_l && (2) \\ & M^T I_m \leq B_r && (3) \\ & M_{ij} \in \{0, 1\} && (4) \end{aligned}$$

where each column of M is revealed in an online fashion. In the case of maximizing diversity, the function f is replaced by the coverage function as opposed to the social welfare (utility) function in the standard case.

In general, we keep the constraints and replace the objective function base on needs. This way we still output a matching that satisfies certian requirements from the stakeholders but with different desirable properties, in this case, diversity.

4.1 Diversity-Greedy

The Diversity-Greedy algorithm takes one side of the matching L and the constraints B_l andB_r as offline input. We also assume that the the algorithm has oracle access to the coverage function f, in the sense that the function f and be evaluated at any input in constant time.

The main idea of the algorithms is as follows. At each round of the algorithm, some vertex $v \in R$ arrives online. The neighbours of v, denoted as $N(v) \subseteq L$ are revealed. Let M_v be the current matching for v. The algorithm then compute the value $f(M_v \cup \{e\})$ for each edge $e \in \{(u, v) : u \in N(v)\}$. It chooses the edge e that maximizes the quantity $f(M_v \cup \{e\}) - f(M_v)$. We call this algorithm OBDBM (online bipartite b-matching).

Algorithm 1 Diversity-Greedy (OBDBM)

Offline Input: The set of vertices R, matching constraints B_l, B_r.
for each $v \in L$ arrive online **do**
    Let M_v be current matching for vertex v
    Consider all neighbours of v in L: $\{u : u \in N(v)\}$ such that their constraint B_{l_u} is not violated.
    Include the edge $e = (u, v)$ in M_v if e maximizes the function $f(M_v \cup \{e\}) - f(M_v)$.
    Stop when the constraint B_{r_v} is satisfied or there is no more edges.
end for
Output: Matching M

4.2 Utility-Greedy

The utility greedy algorithm works essentially the same way as the diversity greedy algorithm. Instead of optimizing a submodular function f, the utility greedy algorithm optimize social welfare function. Such a social welfare function can be, for example, the number of clicks in the Adword problem, or simply the cardinality of the matching.

We note that an extensive amount of research has been conducted in online b-matching problem with various assumptions and constraints. In most cases, a greedy approach yields an optimal competitive ratio (Albers and Schubert, 2021; Aggarwal et al.; Khan et al., 2016). We omit in-depth discussion on this subject as it is not within this project's scope. We adopt the most naive greedy algorithm here.

The purpose of including the utility-greedy algorithm is to compare the diversity of the matchings output from OBDBM, and also as a benchmark algorithm for giving a matching with optimal utility when computing the Price of Diversity for the OBDBM algorithm.

5 Experiment

In this section, we describe the experimental results of the OBDBM algorithm applied to the MovieLens datasets (Harper and Konstan, 2015). The results of the OBDBM is compared against with the utility-greedy algorithm (OUM). In this example, we consider a set of users arrive online, and the goal is to match them with a set of movies with diverse genres. The code is publicly available[2].

5.1 Experiment Setup

There are 3952 movies, 6040 users and 100209 ratings of the movies by the users in this dataset. We sample a reduced dataset for the experiment as follows: users who have given the most ratings are chosen, and a set of 100 movies are sampled randomly. Following the same technique in (Ahmed et al., 2017; Dickerson et al., 2019), we compute the complete predicted rating for these users using a standard collaborative recommender system (Bradley, 2016).

The graph $G = (L \cup R, E)$ is created from the reduced dataset as follows. There is an edge between user u and movie m if u hasn't rated m. The weight on the edge is given by computing the average predicted rating for the genres of the movie from that user. This weight is also used in the weighted coverage function. Concretely, for each user u, the weighted coverage function f_u on a set of genres H is defined in the following way. Let G be the set of all genres. Let $A_g = \{m : m \in g\}$ be the set of movies belong to genre $g \in G$ from the entire reduced dataset. Then the weight function w_u for user u on genre f takes input A_g and outputs the average predicted rating for u on genre

[2] https://github.com/AprilNiu/onlinebipartitebmatching

i. Mathematically, f_u is defined by:

$$f_u(H) = w_u(\bigcup_{g \in H} A_g) = \sum_{g \in H} w_u(A_g).$$

That is, f_u takes input a set of genres, outputs the sum of the average predicted rating on the set of genres. It is maximized when all the movie genres are covered.

The utility of a matching is the sum of the predicted ratings for all users. In the case of maximizing utility, we simply pick the top movies that gives the highest predicted rating at each round.

In this experiment, a set of 10 users arrive online at each round. Every user is recommended upto 5 movies, i.e., $B_{r_u} = 5 \; \forall u$. Each movie can be recommended at most 20 times, i.e., $B_{l_m} = 20 \; \forall m$. We run the experiment for different number of rounds. A user can arrive multiple times, but they are treated as a new user.

5.2 Experiment Results

In this section, we give a summary on the experiment results.

Diverse-Greedy recommends a wider variety of movies. A fraction of the output is shown in Table A.1 in the Appendix. For all users, there is greater diversity (variety of genres) in matching by OBDBM when compared to OUM. The implication and understanding is simple – within the constraints given, OBDBM ensures a higher diversity in user-movie matching when compared to Utility-Greedy. The diversity is quantified using entropy introduced in Definition 3.3. Figure 2 shows that OBDBM almost always out-performs OUM in Entropy. It

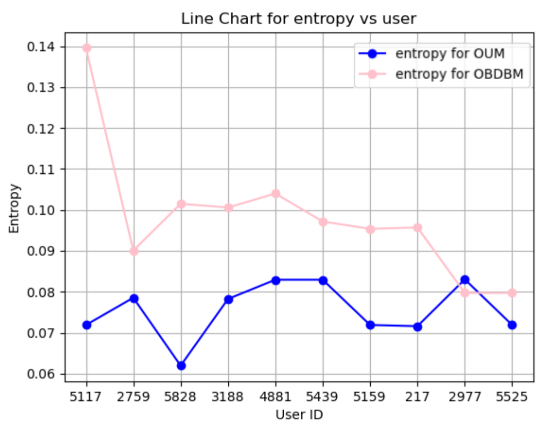

Figure 2: A comparative study of user entropy

then follows that one can expect a drop in utility

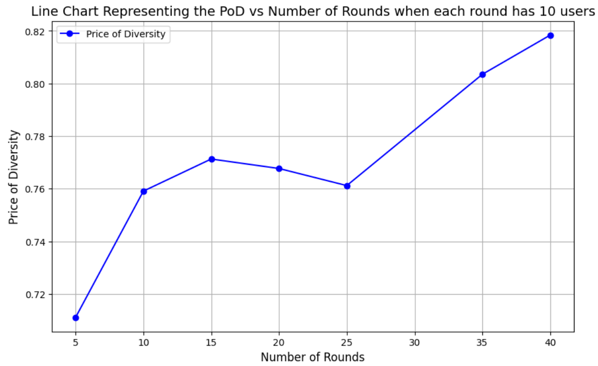

Figure 3: Price of Diversity vs Number of Rounds

with the usage of the Diversity algorithm. Our following observations verify this suspicion.

Price of Diversity. As observed in 5.2, we notice a Price of Diversity is strictly less than 1 across all cases of varying the number of rounds. This confirms our initial suspicion that optimizing for diversity will bear non-negative costs on utility. However, the Price of Diversity increases as the number of rounds increases. The reason why PoD is low when the number of round is small can be explained by the fact that the OBDBM algorithm tries to incorporate as many genres as possible. However, the objective value eventually plateaus as it covers more genres, hence an increase in utility is ultimately observed. This figure suggests that the loss in utility might not be as significant as in the worst case as the number of rounds increases.

6 Conclusion and Future Research

We studied the main aspects of modelling diversity into the problem of online b-matching, and presented greedy algorithms that maximize diversity (OBDBM) and utility(OUM). While OBDBM has higher measures of diversity when compared to OUM, we do not have an established theoretical guarantee for the effectiveness of our diversity-maximizing algorithm, nor a bound on the Price of Diversity. Computing the above will require thought over adversarial distributions of arrivals of vertices, and experimentation with larger datasets. We also want to remark that the definition of diversity (or utility for that matter) is debatable when considering different contexts. Therefore, one should carefully consider what objective function to use when apply these algorithms, as a decision over these definitions has the ability to have large-scale consequences.

References

Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. *Online Vertex-Weighted Bipartite Matching and Single-bid Budgeted Allocations*, pages 1253–1264.

Faez Ahmed, John P. Dickerson, and Mark Fuge. 2017. Diverse weighted bipartite b-matching. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 35–41.

Jamal Alasadi, Ahmed Al Hilli, and Vivek K. Singh. 2019. Toward fairness in face matching algorithms. In *Proceedings of the 1st International Workshop on Fairness, Accountability, and Transparency in MultiMedia*, FAT/MM '19, page 19–25, New York, NY, USA. Association for Computing Machinery.

Susanne Albers and Sebastian Schubert. 2021. Optimal Algorithms for Online b-Matching with Variable Vertex Capacities. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021)*, volume 207 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:18.

Eva O. Arceo-Gomez, Raymundo M. Campos-Vazquez, Raquel Y. Badillo, and Sergio Lopez-Araiza. 2023. Gender stereotypes in job advertisements: What do they imply for the gender salary gap? - journal of labor research. *SpringerLink*.

Sayan Bandyapadhyay, Fedor Fomin, Tanmay Inamdar, Fahad Panolan, and Kirill Simonov. 2023. Socially fair matching: Exact and approximation algorithms. In *Algorithms and Data Structures*, pages 79–92, Cham. Springer Nature Switzerland.

P. Allen Bradley. 2016. Movielens collab- orative filtering.

John P Dickerson, Karthik A Sankararaman, Aravind Srinivasan, and Pan Xu. 2019. Balancing relevance and diversity in online bipartite matching via submodularity. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, AAAI '19.

F. Maxwell Harper and Joseph A. Konstan. 2015. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4).

Bala Kalyanasundaram and Kirk R. Pruhs. 2000. An optimal deterministic algorithm for online b-matching. *Theoretical Computer Science*, 233(1):319–325.

Arif Khan, Alex Pothen, Md. Mostofa Ali Patwary, Nadathur Rajagopalan Satish, Narayanan Sundaram, Fredrik Manne, Mahantesh Halappanavar, and Pradeep Dubey. 2016. Efficient approximation algorithms for weighted \$b\$-matching. *SIAM Journal on Scientific Computing*, 38(5):S593–S619.

Benny Lehmann, Daniel Lehmann, and Noam Nisan. 2001. Combinatorial auctions with decreasing marginal utilities. EC '01, page 18–28, New York, NY, USA. Association for Computing Machinery.

Aranyak Mehta. 2013. Online matching and ad allocation. *Foundations and Trends® in Theoretical Computer Science*, 8(4):265–368.

A Appendix

A.1 Output of our experiment: What genres users get matched to

User ID	Matched Genres by OB-DBM	Matched Genres by OUM
2923	Comedy, Sci-Fi, Animation, Romance, Action, Adventure, Horror, Drama, Musical, Thriller, War, Children's, Crime	Comedy, Drama, Animation, Action, Adventure
5258	Comedy, Sci-Fi, Animation, Romance, Action, Adventure, Horror, Drama, Musical, Thriller, War, Children's, Crime	Comedy, Drama, Animation, War, Action, Thriller
3897	Comedy, Sci-Fi, Animation, Romance, Action, Adventure, Horror, Drama, Musical, Thriller, War, Children's, Crime	Comedy, Drama, Animation, War, Action
318	Comedy, Sci-Fi, Animation, Romance, Action, Adventure, Horror, Drama, Musical, Thriller, War, Children's, Crime	Comedy, Drama, Animation, Action, Thriller
2893	Comedy, Sci-Fi, Animation, Romance, Action, Adventure, Horror, Drama, Musical, Thriller, War, Children's, Crime	Comedy, Drama, Animation, War, Action, Thriller
5533	Comedy,Sci-Fi,Animation,Romance, Action, Adventure, Horror, Drama, Musical, Thriller, War, Children's, Crime	Comedy, Drama, Animation, Film-Noir
3552	Comedy, Sci-Fi, Animation, Romance, Action, Horror, Adventure, Drama, Musical, Thriller, War, Children's,Crime	Comedy, Drama, Animation, War, Action
574	Comedy, Sci-Fi, Animation, Romance, Action, Adventure, Horror, Drama, Musical, Thriller, War, Children's, Crime	Comedy, Drama, Sci-Fi, War, Action, Thriller
4464	Comedy, Sci-Fi, Animation, Romance, Action, Adventure, Horror, Drama, Musical, Thriller, War, Children's, Crime	Comedy, Drama, Animation, War, Action, Thriller
356	Comedy, Sci-Fi, Animation, Romance, Action, Adventure, Horror, Drama, Musical, Thriller, War, Children's, Crime	Comedy, Drama, Animation,

Table 1: Genres of Movies Matched to Users

A.2 Work Division

April

1. Identified the literature gap.
2. Proposed the mathematical program to model the online diversity b-matching.
3. Debugged the code base we borrowed.
4. Implemented the experiments.
5. Conducted presentations to lab-mates and discussed improvements.
6. Co-authored the final report.

Rohit

1. Conducted the literature review.
2. Critiqued and worked on the maximization program suggested by April.
3. Chose the relevant sub-modular functions.
4. Conducted presentations to lab-mates and discussed improvements.
5. Conducted experimental design and interpretation.
6. Co-authored the final report

A.3 Key Learnings

As students in theoretical Computer Science, we were not comfortable coding things up. It is our first time implementing an algorithm designed by ourselves and applying it to a real-world dataset. The codebase we borrowed had multiple bugs and missing files, and did not have helpful comments, and presented us with a lot of challenges. However, it was a lot of fun to debug and play around with different parameters, and observe the effect they have on the outputs. We were also new to the area of online algorithm design and modelling. We realized how small assumptions on the available offline information (for example, distributions of arrivals, modelling of different types, assumptions on budget constraints) can make huge difference when designing these algorithms. Even though we went to great lengths to discuss and receive opinions from PhD students in our lab and our advisor, we felt that we needed more thought and conversations with the people working in this field to produce strong theoretical results. We needed some experimental confirmation before we set out on this task. This is why prioritized empirical results in the end. However, we see great potential and applicability of our project, and are thus determined to produce stronger results with the hope that they can be used for societal benefit.