Group-Aware Weighted Bipartite B-Matching

Cheng Chen
University of Victoria, Canada
cchenv@uvic.ca

Sean Chester
NTNU, Norway
sean.chester@idi.ntnu.no

Venkatesh Srinivasan
University of Victoria, Canada
srinivas@uvic.ca

Kui Wu
University of Victoria, Canada
wkui@uvic.ca

Alex Thomo
University of Victoria, Canada
thomo@uvic.ca

ABSTRACT

The weighted bipartite B-matching (WBM) problem models a host of data management applications, ranging from recommender systems to Internet advertising and e-commerce. Many of these applications, however, demand versatile assignment constraints, which WBM is weak at modelling.

In this paper, we investigate powerful generalisations of WBM. We first show that a recent proposal for conflict-aware WBM by Chen et al. is hard to approximate by reducing their problem from Maximum Weight Independent Set. We then propose two related problems, collectively called group-aware WBM. For the first problem, which *constrains the degree* of groups of vertices, we show that a linear programming formulation produces a Totally Unimodular (TU) matrix and is thus polynomial-time solvable. Nonetheless, we also give a simple greedy algorithm subject to a 2-extendible system that scales to higher workloads. For the second problem, which instead *limits the budget* of groups of vertices, we prove its NP-hardness but again give a greedy algorithm with an approximation guarantee. Our experimental evaluation reveals that the greedy algorithms vastly outperform their theoretical guarantees and scale to bipartite graphs with more than eleven million edges.

CCS Concepts

•Mathematics of computing → Graph algorithms; Approximation algorithms; •Information systems → *Web searching and information discovery;*

Keywords

Bipartite Graphs, Matchings, NP-hardness, Linear Programming, Submodular Systems

1. INTRODUCTION

The weighted bipartite B-matching problem (WBM) is a classic optimisation problem that is ubiquitous in data management and e-commerce applications. Figure 1 illustrates the problem: the input is an edge-weighted, undirected, bipartite graph $G = ((U, V), E, W)$ and a maximum degree constraint (in red) for each vertex. The WBM problem seeks to match vertices in U to vertices in V so that each vertex is matched with no more vertices than its degree constraint allows. Equivalently stated, we want to compute a subgraph H of G with the maximum sum of edge weights where no degree constraint is violated. The solution to the given example is subgraph H with edges given in blue and score 1.6.

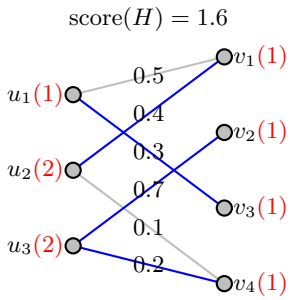


Figure 1: The WBM problem. The input graph has score 2.2, the sum of all its edge weights. The blue edges of the solution H yield the highest score, 1.6, of all subgraphs satisfying the red degree constraints.

When U represents a set of users and V represents a set of items (e.g. webpages, movies, books, etc), WBM expressively models a broad range of data management problems.

For example, if the items are webpages, and (a) the edge weights are expected advertising revenue, (b) users in U have degree constraint 3, and (c) webpages in V have degree constraint $|U|$, then WBM solves the problem of selecting three internet advertisements for every user.

Or, if the items are movies, and (a) the edge weights are predicted ratings, (b) users in U have degree constraint 10, and (c) movies in V have degree constraint $|U|$, then WBM makes the 10 best movie recommendations for every user.

A problem with WBM is that it cannot model diversity constraints on the items matched to users. For example, we cannot specify that no more than 5 history movies should be matched to every user. Diversity is a critical consideration for most applications of WBM, from recommender systems [14] to web document ranking [32] to online shopping [36]. In particular, WBM cannot capture diversity across topics and genres [38], groups of redundant customers, such as households, nor temporal ranges [22].

Recently, Chen et al. [10] introduced a generalisation of

CIKM '16 October 24–28, 2016, Indianapolis, IN, USA

DOI: 10.475/123_4

WBM to increase the expressiveness of its diversity requirement: *conflict-aware* WBM (CA-WBM), adds conflict edges among vertices of V[1] and requires that the solution H contains fewer than $\tau(u)$ conflicts for any vertex $u \in U$. Conflict edges can be added between similar products (e.g., highly similar movies) to ensure that products are diversified for any given user. (CA-WBM is illustrated in Figure 2b.)

This generalisation vastly expands the expressiveness of WBM, but it has two notable shortcomings. Firstly, while WBM can be solved efficiently in polynomial time with the Hungarian Algorithm [21], CA-WBM is instead NP-hard [10]. In fact, our first contribution in this paper is a new hardness result (Section 3) for CA-WBM, reducing from the Maximum Weight Independent Set problem, which proves that it is hard *even to approximate CA-WBM*. Secondly, adding conflict edges between pairs of vertices is often unnecessarily general, because in many applications the conflicts are transitive within *groups* (e.g., households, genres, topics, temporal ranges), i.e., the conflicts arrange in cliques.

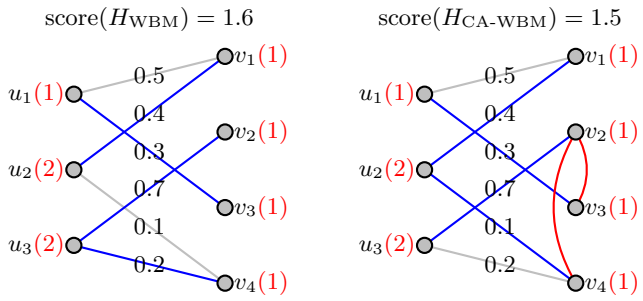
Motivated by the intractability, even inapproximability, of CA-WBM, but the need for more expressive models than WBM, and the organisation of conflicts into disjoint groups, we introduce a novel generalisation of WBM: *group-aware* WBM (GA-WBM). We study two variants of the problem: the first generalises the WBM constraints by constraining the degree for each group (Section 4); the other generalises the WBM optimisation function by imposing a ceiling on the budget/payoff for each group (Section 5). The former captures scenarios when one wishes to limit the overall *number* of products from one group matched to a user, e.g., no more than three news articles on a single topic; the latter captures scenarios when one wishes to limit the overall *weight* of products matched to a user, e.g., no more than $100 is to be spent on online advertising for keywords of a category.

For the degree constraint variant, we present an exact, linear programming algorithm, proving that the problem variant is in **P**, and a scalable, greedy algorithm with approximation guarantees. For the budgetted variant, we prove NP-hardness, but again give a greedy algorithm with a constant-factor approximation guarantee, precisely what we prove cannot be done for CA-WBM. As such, *group-aware weighted bipartite B-matching* can model a broader range of problems than WBM while still admitting efficient algorithms with good performance guarantees.

In all, this paper makes the following four contributions:

- We prove that the CA-WBM problem [10] is hard to approximate by reducing from Maximum Weight Independent Set (Section 3);
- We introduce group-aware WBM subject to *degree constraints* (GA-WBM-D), together with a polynomial-time, exact linear programming algorithm and a scalable, 2-approximate greedy algorithm (Section 4);
- We introduce group-aware WBM subject to *budget ceilings* (GA-WBM-B) and prove it is NP-hard. We give a greedy algorithm using a k-extendible system to guarantee 3-approximate solutions (Section 5); and
- We conduct an extensive experimental evaluation on e-commerce data showing that the linear programs and the greedy algorithms return excellent results on small inputs, and the greedy algorithms scale to bipartite graphs with over eleven million edges (Section 6).

[1]We assume (WLOG) that all conflicts and partitions occur on vertex set V, but they could occur on U, instead, to model diversification across user groups (e.g., households).



(a) H_{WBM}: the WBM solution shown in Figure 1

(b) $H_{\text{CA-WBM}}$: the top score avoiding conflict edge (v_2, v_4)

Figure 2: The CA-WBM problem contrasted with WBM. Two conflicts, $(v_2, v_3), (v_2, v_4)$, are introduced, e.g., because the products are too similar. If u_3 has a conflict threshold $\tau(u_3) = 0$, then it cannot match both v_2 and v_4, leading to a lower score, but potentially more diverse, solution.

2. DEFINITIONS AND PRELIMINARIES

Throughout this paper, we assume that any graph $G = ((U, V), E, W)$ is edge-weighted, undirected, and bipartite. In other words, U and V are disjoint sets of vertices and the edge set $E \subseteq U \times V$ contains only edges between vertices in U and vertices in V. The edge-labelling function $W : E \to \mathbb{R}_+$ assigns a positive, real-valued weight to every edge. The problems in this paper define a scoring function and constraints and find a subgraph $H = ((U, V), E', W)$ that maximises the score while satisfying the constraints.

2.1 Bipartite B-matchings

We review here two known B-matching problems. WBM maximises the sum of edge weights under the condition that each vertex must satisfy a degree bound assigned by a vertex-labelling function, B:

Weighted Bipartite B-Matching (WBM)
Given G and vertex-labelling function $B : U \bigcup V \to \mathbb{N}$, find a subgraph $H = ((U, V), E', W)$ maximising $\sum_{e \in E'} W(e)$ with every vertex $u \in U \bigcup V$ adjacent to at most $B(u)$ edges.

Figure 1 (WBM) is repeated in Figure 2a for ease of comparison. The matching excludes the high-weight (u_1, v_1) edge, because u_1 and v_1 are permitted only one edge and $\{(u_1, v_3), (u_2, v_1)\}$ produces a higher overall score.

CA-WBM imposes a set of conflict pairs C, requiring that each $u \in U$ is adjacent to at most $\tau(u)$ of those pairs:

Conflict-Aware
Weighted Bipartite B-Matching (CA-WBM) [10]
Given G, vertex-labelling functions $B : U \bigcup V \to \mathbb{N}$, $\tau : U \to \mathbb{N}$, and a set of unordered pairs $C \subseteq V \times V$, find the subgraph $H = ((U, V), E', W)$ maximising $\sum_{e \in E'} W(e)$ with every vertex $u \in U \bigcup V$ adjacent to at most $B(u)$ edges and every vertex $u \in U$ adjacent to at most $\tau(u)$ pairs of vertices $v, v' \in V$ that appear as an unordered pair $(v, v') \in C$.

Figure 2b illustrates CA-WBM. The conflict pairs (v_2, v_3) and (v_2, v_4) are marked by red arcs and the threshold function is: $\tau(u) = 0, \forall u \in U$. Relative to WBM, the conflicts restrict the space of feasible solutions; e.g., the subgraph in Figure 2a would not solve the CA-WBM instance, because u_3 is matched to both vertices of the conflict pair (v_2, v_4).

2.2 k-Extendible systems

We use the notion of k-extendible systems to prove approximation guarantees for our greedy algorithms in Sections 4.3 and 5.3. Moreover, we will deal with objective functions that are linear/monotone submodular and need to be maximised. We review here what it means for a function to be monotone submodular and what is a k-extendible system constraint. Submodular set functions are used to capture a natural diminishing returns property:

DEFINITION 1 (MONOTONE SUBMODULAR FUNCTION). *Let U be a finite set of elements. A positive set function $f : 2^U \to \mathbb{R}_+$ is monotone if for any two sets $A \subseteq B \subseteq U$, we have that $f(A) \leq f(B)$. Set function f is submodular if for every $A, B \subseteq U$ with $A \subseteq B$ and every $x \in U \setminus B$, we have that $f(A \cup \{x\}) - f(A) \geq f(B \cup \{x\}) - f(B)$.*

A k-extendible system constraint models a matroid-like property satisfied by the set of feasible solutions to an objective function.

DEFINITION 2 (k-EXTENDIBLE SYSTEM [29]). *Let U be a finite set and $\mathcal{F}$, $\mathcal{F} \subseteq 2^U$, be a collection of subsets of U. Set system $(U, \mathcal{F})$ is called a* k-extendible system *if it satisfies the following properties:*

1. Downward-closure*: If $A \subseteq B$ and $B \in \mathcal{F}$, then $A \in \mathcal{F}$.*
2. Exchange*: Let $A, B \in \mathcal{F}$ with $A \subseteq B$, and let $x \in U \setminus B$ be such that $A \cup \{x\} \in \mathcal{F}$. Then there exists $Y \subseteq B \setminus A$, $|Y| \leq k$, such that $(B \setminus Y) \cup \{x\} \in \mathcal{F}$. In other words, let us start with any choice of two sets A and B such that B is an extension of A. Suppose that there is an element x such that the set A with x added to it also belongs to $\mathcal{F}$. Then we will be able to find a subset Y inside B of size at most k such that if we remove the elements of Y from B and add the element x to the resulting set, it will also belong to the collection $\mathcal{F}$.*

Fisher *et. al* [31] showed that if the set of feasible solutions forms a k-extendible system and the objective function is positive, monotone submodular, a natural greedy algorithm, that incrementally adds an element that most improves the current solution, is guaranteed to produce a $(k+1)$-approximate solution.

THEOREM 1 (FISHER, NEMHAUSER, WOLSEY [31]). *Let $(U, \mathcal{F})$ be a k-extendible system for some k. Let $W : 2^U \to \mathbb{R}_+$ be positive, monotone submodular function. The greedy algorithm gives a $(k+1)$-approximation algorithm for the optimization problem that asks to determine $\max_{F \in \mathcal{F}} W(F)$.*

Mestre [29] showed a stronger result that if the objective function is indeed *linear*, the greedy algorithm yields a k-approximate solutions.

THEOREM 2 (MESTRE [29]). *Let $(U, \mathcal{F})$ be a k-extendible system for some k. Let $W : U \to \mathbb{R}_+$ be a weight function on U. The greedy algorithm gives a k-approximation algorithm for the optimization problem that asks to determine $\max_{F \in \mathcal{F}} W(F)$ where $W(F) = \sum_{s \in F} W(s)$ for any $F \in \mathcal{F}$.*

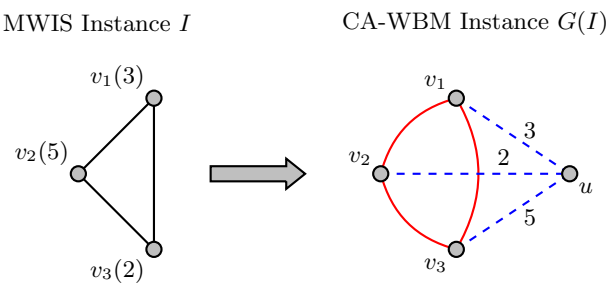


Figure 3: The reduction from MWIS to CA-WBM. Vertex weights are in parentheses and edge weights are above the edge. Edges in the original graph become (red) conflicts, while all vertices are connected to a new vertex u with a dashed blue edge. Degree constraints (not shown) are 1 for v_1, v_2, v_3 and 3 for u (i.e., the degrees of the vertices).

3. STRONGER HARDNESS FOR CA-WBM

CA-WBM is already known to be NP-Hard [10], based on a reduction from the NP-hard Revenue Maximisation in Interval Scheduling problem [3, 7, 19]. Here we give a simpler reduction from the Maximum Weight Independent Set (MWIS) problem (defined below), which, due to Håstad [17], has the stronger implication of being hard to approximate.

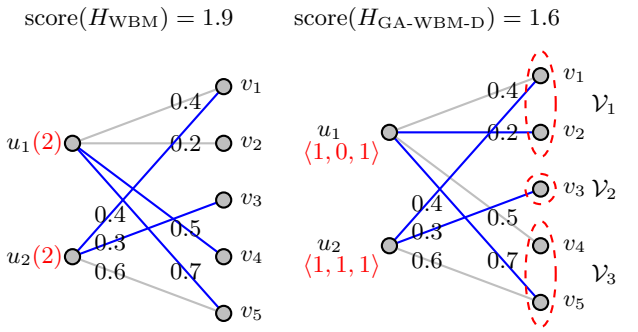
Maximum Weight Independent Set (MWIS)
Given a (not generally bipartite) graph $G = (V, L, R)$, with vertex set $V = \{v_1, \ldots, v_n\}$, edge set $L = \{l_1, \ldots, l_m\}$, and vertex-labelling function $R : V \to \mathbb{R}_+$, find the independent set $V' \subseteq V$ that maximises $\sum_{v \in V'} R(v)$.

THEOREM 3. *Unless P=NP, there is no approximation algorithm for CA-WBM with an approximation ratio $n^{1-\epsilon}$, for any fixed $\epsilon > 0$, where $n = |U \cup V|$.*

PROOF. We give a polynomial-time reduction from the NP-hard Maximum Weight Independent Set (MWIS) problem. Let I be an instance of MWIS. We construct a graph $G(I) = ((U, V), E, W)$, which is an instance of CA-WBM, as follows:

- $U = \{u\}$ and V is the same as in G,
- $W(v_k, u) = R(v_k)$ for $1 \leq k \leq n$,
- $C = L$,
- $B(v_k) = 1$ for $1 \leq k \leq n$ and $B(u) = n$, and
- $\tau = 0$.

We construct a graph with all vertices V of I isolated, and connect them to a new vertex u. Any adjacent vertices $(u, v) \in L$ are added to the set of conflicts C. Denote E as the set of edges connecting u and vertices of V, i.e., $E = \{u\} \times V$ and assign the weight of vertex v_i to edge (u, v_i). Figure 3 illustrates an example of such a reduction.



(a) Solution H_{WBM}, no group constraints

(b) Solution $H_{\text{GA-WBM-D}}$ satisfying group *degree* constraints

Figure 4: Contrasting WBM (a) and GA-WBM-D (b). All right-hand vertices $v \in V$ have degree constraint 1. In WBM, the only modellable diversity constraint is that u_1, u_2 cannot both choose v_1 nor v_5. In GA-WBM-D, we explicitly model three equivalence classes of similar vertices in V. We constrain u_1 and u_2 to matching at most $\langle 1, 0, 1\rangle$ and $\langle 1, 1, 1\rangle$ vertices, respectively, from $\langle \mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3\rangle$, producing a more diversified assignment.

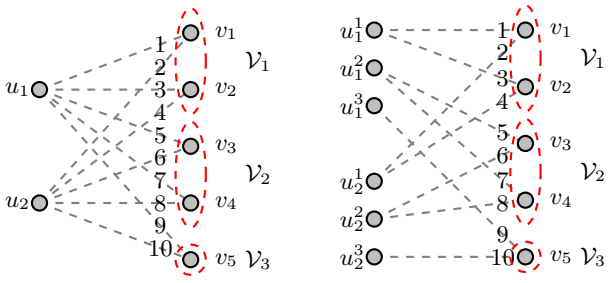
For an optimal solution O of the CA-WBM instance $G(I)$, any two matched edges in O must not have the same end point in V, since $\tau = 0$. This implies that O corresponds to an independent set in V for I, an instance of MWIS. In addition, since $W(v_k, u) = R(v_k)$ for $1 \leq k \leq n$, the total weight of edges in O also corresponds to the total weight of vertices in the independent set in V. Hence, a maximum weight subgraph of an instance of CA-WBM satisfying the degree constraints and conflict constraints exactly corresponds to a maximum weight independent set for an instance of MWIS. Furthermore, we observe that the reduction above is a polynomial-time reduction, gap-preserving reduction. Therefore, using the hardness of approximation result for MWIS [17], we conclude that CA-WBM is NP-hard and is not approximable with $n^{1-\epsilon}$ for any $\epsilon > 0$. □

4. GA-WBM + DEGREE CONSTRAINTS

In this section, we introduce our first GA-WBM variant (Section 4.1), wherein each group can be adjacent to a limited number of edges. We present an exact, efficient linear programming algorithm (Section 4.2), demonstrating that GA-WBM-D $\in \mathbf{P}$ (Theorem 4), and a scalable, greedy algorithm (Section 4.3) that is 2-approximate (Theorem 5).

4.1 Problem Formulation

In many applications, items cluster into groups of mutual similarity, and users should be matched to a limited number of items from the same group. For example, books can be grouped by genre, topic, or author, and diverse matchings for a user should span the groups. GA-WBM partitions the items into equivalence classes (groups) and imposes constraints on how much each user can be matched to each class. In comparison to CA-WBM [10], partitioning into equivalence classes can be viewed as all conflicts being transitive (i.e., the existence of conflicts (a, b) and (b, c) implies (a, c) is a conflict). For genres, topics, and authors of books, transitivity clearly holds.



(a) Original graph

(b) Split graph

Figure 5: Setup of the linear program for GA-WBM-D. The original vertices of U (a) are split, one for each equivalence class (b). The red ellipses indicate equivalence classes and the integers on the edges show their sequence numbers.

In the degree-constraints version of GA-WBM, we limit the *number of edges* that match each user u to each equivalence class $\mathcal{V}_i$. The problem is illustrated and contrasted to WBM in Figure 4: here, V is partitioned into 3 groups, $\mathcal{V} = \langle \mathcal{V}_1 = \{v_1, v_2\}, \mathcal{V}_2 = \{v_3\}, \mathcal{V}_3 = \{v_4, v_5\}\rangle$. Each vertex of U has now a sequence of degree constraints, one for each $\mathcal{V}_i$. Thus GA-WBM can model that u_1 should match to at most one of v_4, v_5, a diversity constraint that is inexpressible in WBM. We obtain an instance of WBM if we create only one equivalence class, in which case the degree constraint sequence reduces to a single degree constraint for each $u \in U$.

Formally, we represent the degree constraints as a mapping from user-class pairs to integers. The number of edges from a user u to vertices in an equivalence class $\mathcal{V}_i$ must not exceed the limit specified in the mapping.

Group-Aware Weighted Bipartite B-Matching Subject to Degree Constraints (GA-WBM-D)
Given G, a vertex-labelling function $B : V \to \mathbb{N}$, a partitioning of V into k equivalence classes $\mathcal{V} = \langle \mathcal{V}_1 \ldots, \mathcal{V}_k\rangle$, and a U-degree-constraint mapping $D : (U \times \mathcal{V}) \to \mathbb{N}$, find the subgraph $H = ((U, V), E', W)$ maximising $\sum_{e \in E'} W(e)$ with every vertex $v \in V$ adjacent to at most $B(v)$ edges and for all vertex-class pairs $(u \in U, \mathcal{V}_i \in \mathcal{V})$:

$$|\{e = (u, v_j) \in E' : v_j \in \mathcal{V}_i\}| \leq D(u, \mathcal{V}_i).$$

4.2 A Linear Program for GA-WBM-D

Here we present a linear programming formulation to solve GA-WBM-D, given inputs $G = ((U, V), E, W), \mathcal{V}, B, D$, with $m = |U|$ and $n = |V|$. We denote by $X = [x_{ij}]^T$ the mn-dimensional column vector of 0-1 variables, by $x_{ij} = 1$ that item i is matched to user j and by $x_{ij} = 0$ otherwise. Then GA-WBM-D *finds the set of matches such that the total profit is maximized under the degree constraints*, i.e.,

$$\begin{aligned} \max_X \quad & WX \\ s.t. \quad & \mathbb{A}X(i) \leq Q(i), \forall i, 1 \leq i \leq km + n \\ & x_{ij} \in \{0, 1\}, \forall i, j, 1 \leq i \leq m, 1 \leq j \leq n, \end{aligned} \tag{1}$$

where matrix $\mathbb{A}$ is an $(km + n) \times mn$ matrix and Q is a vector of values of B and D. That is, the degree constraints are given by $\mathbb{A}X(i) \leq Q(i)$, where $\mathbb{A}X(i)$ denotes the i-th element in (vector) $\mathbb{A}X$ and $Q(i)$ the i-th element in Q.

Below, we show that GA-WBM-D is in **P** by proving that $\mathbb{A}$ is totally unimodular (TU). A matrix A is said to be totally unimodular if the determinant of each square submatrix of A is 0, -1 or 1. As [35, 33] show, if $\mathbb{A}$ is TU, the polyhedron $P = \{X : \mathbb{A}X \leq Q\}$ is integral and an integral optimal solution can be found in polynomial time using an LP algorithm.

LEMMA 1. *In the LP formulation of GA-WBM-D, the coefficient matrix $\mathbb{A}$ is totally unimodular (TU).*

PROOF. We need to prove that the matrix $\mathbb{A}$ in Problem 1 is totally unimodular. Note that $\mathbb{A}$ captures two types of degree constraints; 1) a degree constraint of each item, and 2) a sub-degree constraint for each user with regard to each item group. There are n constraints of type 1 and km constraints of type 2 for a total of $km + n$ constraints. In other words, A is a $(km + n) \times mn$ matrix.

In order to prove the lemma, we would like to make use to well-known result that the incidence matrix of a bipartite graph is totally unimodular [4, 37]. Our main observation is that considering each users' sub-degree constraints for k conflict groups is equivalent to replacing each user vertex with k copies, with each only connecting to a single item group. Figure 5 shows such a transformation. The degree constraint of each copy is the original vertex's sub-degree constraint for the corresponding item group. This transformation only increases the number of user vertices and the transformed graph is still a bipartite graph. We call the resulting graph as the split graph.

From the transformation in Figure 5, we can see that $\mathbb{A}$ is the incidence matrix of the split graph in which original user vertices have been replaced by the copies and is therefore totally unimodular.

Therefore, we conclude that in the LP formulation of GA-WBM-D, the coefficient matrix $\mathbb{A}$ is totally unimodular and hence solving the LP always yields integral solutions. □

From [35, 33], we immediately have the following theorem:

THEOREM 4. *There exists an algorithm that solves GA-WBM-D in polynomial time.*

4.3 A Greedy Algorithm for GA-WBM-D

Although Program 1 in Section 4.2 can be solved in polynomial time and is exact, it scales poorly (see Section 6.3). Therefore, we also introduce GREEDY-D (Algorithm 1), a simple but provably approximate greedy algorithm. The idea is to start with an empty edge set E' (Line 1) and repeatedly add the next highest-weighted edge e (Lines 2–3) if $E' \bigcup \{e\}$ does not violate the degree constraints (Line 4). After trying all $e \in E$, we return $H = ((U, V), E', W)$, a new graph constructed with the greedily built edge set E'.

Although we will later evaluate the efficiency, scalability, and accuracy of GREEDY-D empirically (Section 6.3), we prove here a theoretical *guarantee* on its performance:

THEOREM 5. *Algorithm 1 is a 2-approximation algorithm.*

PROOF. Recall from Section 2.2 that we use the concept of a *k-extendible system* to performance guarantees for greedy algorithms. To apply this result to our problem, we will check that the set of all feasible solutions to GA-WBM-D forms a 2-extendible system. For the GA-WBM-D problem, let $U = E$ and $\mathcal{F}$ be the set of all subgraphs of G satisfying the degree constraints on the items and the sub-degree constraints on the users. Then it is easy to see that $(U, \mathcal{F})$ is downward closed. That is, removing an edge from a feasible solution H will always result in a feasible solution as this will not cause any violation of constraints.

For the exchange property, consider the case when a new edge $e = (u, v)$ is added to a feasible solution H. Assume that $v \in \mathcal{V}_i$. We observe that adding e could result in a violation of the degree constraint at v, and sub-degree constraint at u. However, this can be rectified by removing two other edges, one incident on u and other incident on v. Therefore, we obtain a 2-extendible system. □

Algorithm 1: GREEDY-D

Input: $G = ((U, V), E, W), \mathcal{V}, B, D$
Output: A subgraph $H = ((U, V), E', W)$ satisfying constraints B, D with a greedily-maximised score, $\sum_{e \in E'} W(e)$

```
1 E' = ∅
2 Sort E by descending W(e)
3 for e ∈ E do
4     if H = ((U,V), E' ⋃ {e}, W) does not violate B, D then
5         E' = E' ⋃ {e}
6 return A new graph H = ((U,V), E', W)
```

5. GA-WBM + BUDGET CEILINGS

This section presents a variant of GA-WBM that introduces diversity through the optimisation function rather than the constraints: it introduces a ceiling on the sum of edge weights each $u \in U$ can amass for each group, $\mathcal{V}_i$. We formally define the problem and prove it is hard (Section 5.1). We then give an ILP formulation along with a tractable relaxation (Section 5.2) and, again, a greedy algorithm with a constant-factor approximation guarantee (Section 5.3).

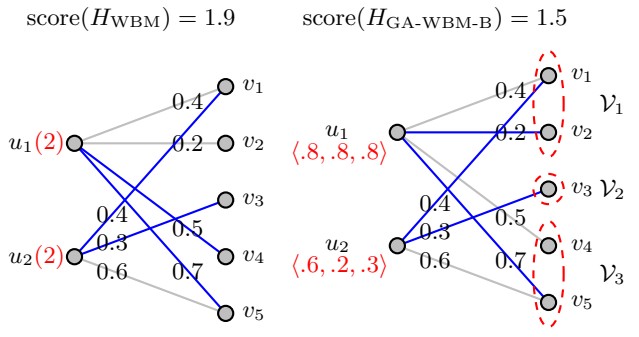
5.1 Problem Formulation

Section 4.1 formally modelled diversity in V by partitioning V into groups and constraining each user $u \in U$ to a limited number of edges per group. With GA-WBM-B, we cap the score each user can derive from each group by mapping user-group pairs onto a real-valued budget ceiling. Figure 6 illustrates how capping budgets can produce the same diverse matchings as the group degree constraints in Figure 4.

Group-Aware Weighted Bipartite B-Matching Subject to Budget Ceilings (GA-WBM-B)
Given G, $B : U \bigcup V \to \mathbb{N}$, a partitioning of V into $\mathcal{V} = \langle \mathcal{V}_1 \ldots, \mathcal{V}_k \rangle$, and a mapping $C : (U \times \mathcal{V}) \to \mathbb{R}_+$, find the subgraph $H = ((U, V), E', W)$ with every vertex $u \in U \bigcup V$ adjacent to $\leq B(u)$ edges that maximises the score function:

$$\sum_{u, \mathcal{V}_i} \min \left\{ C(u, \mathcal{V}_i), \sum_{(u, v_j) \in E', v_j \in \mathcal{V}_i} W(u, v_j) \right\}.$$

This formulation is more natural for some applications. For example, sponsored search auctions hosted by search engines (e.g., Google, Bing, and Yahoo) include budget specification as a feature. To achieve better coverage, advertisers



(a) Solution H_{WBM}, repeated from Figure 4a

(b) Solution $H_{\text{GA-WBM-B}}$ satisfying *budget ceilings* for user-group pairs

Figure 6: Contrasting WBM (a) and GA-WBM-B (b) (c.f., Figure 4b). $B(v) = 1\ \forall v \in V$ and $B(u) = 3\ \forall u \in U$. The $\mathcal{V}_3$ budget for u_1 is nearly saturated by the edge (u_1, v_5); so, a higher score can be obtained by matching (u_1, v_2) rather than (u_1, v_4). The $\mathcal{V}_2$ budget for u_2 is over-satured by (u_2, v_3), but no alternative produces a higher score; so, the score only obtains the ceiling of 0.2 from the pair $u_2, \mathcal{V}_2$.

who bid on keywords can specify budgets for different categories of keywords. The constraints are identical to WBM, but a subgraph will no longer profit from matching a specific user and item group once the ceiling has been hit; so, maximising the objective function requires diversifying across groups. We obtain an instance of WBM if the ceilings are sufficiently high, e.g., $\sum_{e \in E} W(e)$ for all user-group pairs.

GA-WBM-B generalizes the *maximum budgeted allocation* (MBA) problem [2, 16, 23, 34]. Given n items and m users, where each user i with budget C_i is willing to pay w_{ij} on item j, MBA finds an allocation which maximises the total revenue. In MBA, each user has an overall budget on all items and there is no constraint on the number of allocated items. Each item, however, can be allocated only once. Therefore, MBA can be regarded as a special case of GA-WBM-B in which k is 1, $B(u) = n$ for $u \in U$ and $B(v) = 1$ for $v \in V$. Since MBA is known to be NP-hard and it is a special case of GA-WBM-B, GA-WBM-B is also NP-hard. Therefore we have Theorem 6:

THEOREM 6. *GA-WBM-B is NP-hard.*

5.2 Integer LP for GA-WBM-B

If one considers the sum of edge weights (e.g., users' payments) in E' to be the total revenue, then GA-WBM-B finds the allocation which maximises the total revenue while satisfying degree constraints. Due to the existence of budget ceilings, users will have to receive items from diverse groups. GA-WBM-B can be formulated as a linear program as follows by adapting the formulation in (1):

$$\begin{aligned} \max_{X} \quad & \sum_{i \in U} \sum_{p=1}^{k} \min \left\{ C(i, \mathcal{V}_p), \sum_{j \in \mathcal{V}_p} w_{ij} x_{ij} \right\} \\ s.t. \quad & \mathbb{A}X(i) \leq B(i), \forall i, 1 \leq i \leq m+n \\ & x_{ij} \in \{0,1\}, \forall i, j, 1 \leq i \leq m, 1 \leq j \leq n, \end{aligned} \tag{2}$$

where $X = [x_{ij}]^T$.

Since obtaining an integer solution from the LP in GA-WBM-B is NP-hard, we use a rounding procedure to further improve efficiency after solving the LP relaxation. Our LP-based algorithm for GA-WBM-B is as follows:

1. Solve the linear program relaxation to obtain an optimal solution S.
2. Sort the first mn elements of S from largest to smallest. We round each non-zero value to 1 provided doing so does not violate the degree constraints or the budget ceilings. Otherwise, we set it to 0.

We refer to this as the *LP relaxation with rounding*.

Algorithm 2: GREEDY-B

Input: $G = ((U, V), E, W), \mathcal{V}, B, C$
Output: A subgraph $H = ((U, V), E', W)$ satisfying constraints $\mathcal{V}, B, C$ with a greedily-maximised score, $\sum_{e \in E'} W(e)$

1 $E' = \emptyset$, $A = \emptyset$
2 $\mathcal{F}$ is a universal set that contains all candidate edge sets satisfying the degree constraints
3 $A = \{e | E' \cup e \in \mathcal{F}\}$
4 **while** $A \neq \emptyset$ **do**
5 $\quad e^* = \arg\max_{e \in A} \Delta_{E'}(e)$
6 $\quad E' = E' \cup \{e^*\}$
7 $\quad A = \{e | E' \cup \{e\} \in \mathcal{F}\}$
8 **return** A new graph $H = ((U, V), E', W)$

5.3 A Greedy Algorithm for GA-WBM-B

Given the hardness of GA-WBM-B, we introduce a greedy algorithm with a theoretical guarantee by again establishing connections to the problem of maximizing a non-negative monotone submodular function subject to a k-extendible system constraint. In Problem 2, the objective function,

$$g(X) = \sum_{i \in U} \sum_{p=1}^{k} \min \left\{ C(i, \mathcal{V}_p), \sum_{j \in \mathcal{V}_p} w_{ij} x_{ij} \right\},$$

is the sum of a series of budget-additive functions, $\min(C(i, \mathcal{V}_p), \sum_{j \in \mathcal{V}_p} w_{ij} x_{ij})$, each of which is a monotone submodular function. Since the sum of submodular functions retains submodularity [15, 20], $g(X)$ is also submodular. w_{ij} is non-negative, thus it is easy to see that $g(X)$ is monotone. In addition, as discussed in Section 4.3, the degree constraints form a 2-extendible system.

Algorithm 2 outlines a natural greedy algorithm, GREEDY-B, for GA-WBM-B. Denote $\delta_l = \Delta_{E'_{l-1}}(e_l) = g(E'_l) - g(E'_{l-1})$ as the value of increment (marginal revenue) in the objective function when the l^{th} edge e_l is added. The algorithm starts with an empty solution edge set E' and in each iteration adds to E' an edge that provides the maximum marginal revenue without violating degree constraints. To ensure the newly added edge is always valid, Algorithm 2 keeps track of a set of all valid edges (A in Algorithm 2) with regard to the current E' (Lines 3 and 7). For any $e \in A$, it can be safely added to E' and the increased edge set still satisfies degree constraints, i.e., $E' \cup \{e\} \in \mathcal{F}$, where $\mathcal{F}$ represents a collection of subsets of E, each of which satisfies degree constraints. Fisher, Nemhauser and Wolsey [31] showed that this greedy algorithm gives a $(k+1)$-approximation algorithm to the problem of maximising a non-negative monotone submodular function subject to a k-extendible system constraint. Therefore, GREEDY-B is 3-approximate for GA-WBM-B.

Algorithm 3: GREEDY-B-LF

Input: $G = ((U, V), E, W), \mathcal{V}, B, C$
Output: A subgraph $H = ((U, V), E', W)$ satisfying constraints $\mathcal{V}, B, C$ with a greedily-maximised score, $\sum_{e \in E'} W(e)$

1 $E' = \emptyset$
2 Set the upper heap ($\mathcal{H}^{\text{upper}}$) as an empty max heap
3 **for** $u \in U$ **do**
4 Set the lower heap ($\mathcal{H}^{\text{lower}}_{(u)}$) as an empty max heap
5 **for** u's neighbouring vertices $v \in V$ **do**
6 $w = W((u, v))$; $\delta = \min(w, C_{u,p})$, $v \in \mathcal{V}_p$
7 $\mathcal{H}^{\text{lower}}_{(u)}$.Append($(\delta, (u, v))$);
8 Heapify($\mathcal{H}^{\text{lower}}_{(u)}$); $\mathcal{H}^{\text{upper}}$.Append($\mathcal{H}^{\text{lower}}_{(u)}[0]$)
9 Heapify($\mathcal{H}^{\text{upper}}$)
10 **while** $\mathcal{H}^{\text{upper}} \neq \emptyset$ **do**
11 $(\delta, e) = \mathcal{H}^{\text{upper}}[0]$, $e = (u, v)$, $v \in \mathcal{V}_p$
12 **if** $\delta \leq 0$ **then break**
13 **if** $E' \cup \{e\}$ violates degree constraints on u or v **then**
14 Heappop($\mathcal{H}^{\text{upper}}$); Heappop($\mathcal{H}^{\text{lower}}_{(u)}$)
15 **if** $\mathcal{H}^{\text{lower}}_{(u)} \neq \emptyset$ **then** Heappush($\mathcal{H}^{\text{upper}}$, $\mathcal{H}^{\text{lower}}_{(u)}[0]$)
16 **else if** $\delta \leq C_{u,p}$ **then**
17 $E' = E' \cup \{e\}$; Update $C_{u,p}$ and degree constraints on u and v
18 Heappop($\mathcal{H}^{\text{upper}}$); Heappop($\mathcal{H}^{\text{lower}}_{(u)}$)
19 **if** $\mathcal{H}^{\text{lower}}_{(u)} \neq \emptyset$ **then** Heappush($\mathcal{H}^{\text{upper}}$, $\mathcal{H}^{\text{lower}}_{(u)}[0]$)
20 **else**
21 $(\delta_{\text{root}}, e_{\text{root}}) = \mathcal{H}^{\text{lower}}_{(u)}[0], e_{\text{root}} = (u_{\text{root}}, v_{\text{root}})$, $v_{\text{root}} \in \mathcal{V}_p$
22 **while** $\delta_{\text{root}} > C_{u,p}$ **do**
23 Update δ_{root} and maintain the heap invariant of $\mathcal{H}^{\text{lower}}_{(u)}$
24 $(\delta_{\text{root}}, e_{\text{root}}) = \mathcal{H}^{\text{lower}}_{(u)}[0], e_{\text{root}} = (u_{\text{root}}, v_{\text{root}})$, $v_{\text{root}} \in \mathcal{V}_p$
25 Heappop($\mathcal{H}^{\text{upper}}$)
26 Heappush($\mathcal{H}^{\text{upper}}$, $\mathcal{H}^{\text{lower}}_{(u)}[0]$)
27 **return** A new graph $H = ((U, V), E', W)$

Algorithm 2 provides an intuitive high-level description of the greedy idea. This simple approach, however, has poor scalability in practice. For example, in each iteration, the reconstruction of A must check every edge that is not in E', which is expensive if the original edge set E is large. In fact, this is often the case in the real-world, where graph datasets can have millions of edges. Therefore, the implementation of the greedy approach requires non-trivial optimization to be scalable to real-world datasets which are often large in size.

GREEDY-B-LF (Algorithm 3) improves the scalability of GREEDY-B by employing similar techniques to those proposed in [30] and [25]. The main idea is to efficiently identify a valid edge with maximum marginal revenue and keep the marginal revenue of an edge updated only when necessary. Specifically, we implement a two-level heap (e.g., Figure 7) to store all valid edges and apply the lazy forward (LF) technique for updating the marginal value when necessary.

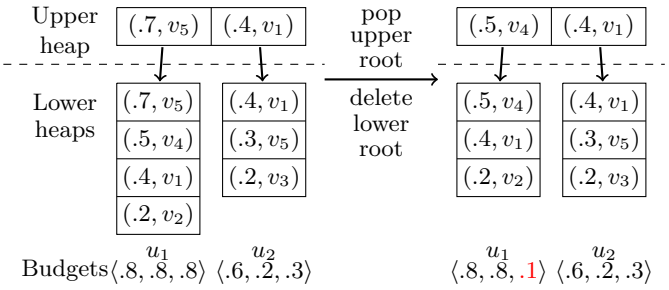


Figure 7: The two-level heap and the lazy forward technique. This example corresponds to Figure 6b.

Using the heap structure improves the efficiency of finding an edge with maximum marginal value. For large graphs, however, maintaining the heap invariant in a single, large heap of marginal revenues for all edges is too expensive. So, we exploit the observation that distinct item vertices in V outnumber those user vertices in U (Table 1), and for any two vertices $u_1, u_2 \in U$, marginal revenue updates to edges of u_1 does not affect edges of u_2. Therefore we use a two-level heap data structure as shown in Figure 7. In Figure 7, an entry of each lower heap of $u \in U$ is a tuple of u's neighbouring vertex $v \in V$ and v's marginal revenue with regard to u's current budget (Lines 4-7 in Algorithm 3). The numbers below u's lower heap show u's current budget for each class. Since each $u \in U$ maintains its own heap, the size of each heap is reduced from $|E|$ to $\approx |E|/|U|$, and the corresponding maintenance cost is much less. This approach also applies to many other scenarios, such as budgeted resource allocation and online advertising, where GA-WBM-B comes in naturally, and where the bipartite graphs often have imbalanced vertex sets. Ergo, we can use the smaller set's entities to distinguish the lower-level heaps, i.e., in our case, each distinct $u \in U$ maintains a heap structure where nodes consist of all neighbouring $v \in V$ and the corresponding marginal values. The upper-level heap is constructed by the roots of the lower-level heap (Line 8) and it will be used for obtaining the edge with maximum marginal revenue (Line 11).

In addition, the lazy forward (LF) technique [30] can significantly reduce unnecessary computation, because it updates a stale marginal value of a node of the lower-level heap only when the node becomes the root of the upper heap (Lines 21-24 in Algorithm 3). Figure 7 shows an example of LF. At the current iteration, after popping the current upper root which is the root of u_1's heap, u_1's budget for $\mathcal{V}_3$ is reduced from 0.8 to 0.1. Since $v_4, v_5 \in \mathcal{V}_3$, the marginal revenue of v_4 to u_1 should be reduced from 0.5 to 0.1. However, LF suppresses the update and hence v_4 of u_1 still has a larger marginal revenue and becomes the new upper root. In the next iteration, the upper root will be considered stale and will be updated.

6. EXPERIMENTAL EVALUATION

In Sections 4–5 we proved worst-case guarantees for the greedy algorithms. Here, we experimentally evaluate how closely the greedy algorithms perform to the worst-case guarantees and how all proposed algorithms scale with input size.

6.1 Methodology

	eBay Canada				eBay US			
	25%	50%	75%	100%	25%	50%	75%	100%
# Sellers ($\|U\|$)	471	942	1,413	1,884	66,751	90,925	109,511	126,101
# Buyers ($\|V\|$)	4,701	9,381	14,062	18,742	1,574,114	2,988,717	4,300,322	5,751,334
# Edges ($\|E\|$)	14,130	28,260	42,390	56,520	2,846,880	5,693,759	8,540,638	11,387,517

Table 1: Statistics of the semi-synthetic eBay datasets.

	25%	50%	75%	100%
LP	2.98	11.50	22.64	41.88
GREEDY-D	0.06	0.13	0.23	0.29

Table 2: GA-WBM-D run times (s) on eBay Canada

	25%	50%	75%	100%
ILP	0.91	2.50	2.66	3.96
LPR	0.55	1.00	1.52	2.10
GREEDY-B-LF	0.06	0.13	0.20	0.28

Table 3: GA-WBM-B run times (s) on eBay Canada

All experiments are run on a 64-bit Ubuntu 14.04 desktop of 3.40 GHz * 8 Intel Core i7 CPU with 12 GB memory.

GA-WBM-D Both algorithms (LP and greedy) are polynomial time, but the size of the LP grows quickly. We use small datasets to evaluate the accuracy of the greedy algorithm relative to the (exact) LP and the scalability of the LP. We use large graphs to test the greedy algorithm's scalability.

GA-WBM-B As GA-WBM-B is NP-hard, our LP relaxation with rounding (LPR) is also approximate. We use small graphs to evaluate the accuracy of LPR and the greedy method and the scalability of the ILP/LPR methods. Large graphs again test the greedy algorithm's scalability.

For LP, ILP and LPR, we use the general mathematical programming solver, Gurobi[2], on Matlab. When we measure the running time of an algorithm, we exclude the time spent on loading data from the disk to the memory.

6.2 Datasets

We use two of our semi-synthetic eBay transaction datasets of [10] and described in Table 1.Each transaction dataset consists of seller vertices (U), buyer vertices (V), and edges (E) representing buyer-seller interactions, such as purchases. While the graph structure of eBay US reflects the true interaction, the eBay Canada dataset has imputed edges: according to [10], each seller is connected to 30 buyers after imputation. For each dataset, there are four subsets of different sizes from the full graph, i.e., using 25%, 50%, 75%, and 100% of the total number of edges (the number of buyer and seller nodes decreases accordingly).

To accommodate the respective problem scenarios, we assign random integer weights from the range $[1, 1000]$ to edges in both datasets. Each $v \in V$ is uniformly, randomly classified into one of 20 groups. For GA-WBM-D, the group-degree constraint of each $u \in U$ (i.e. $D(u, \mathcal{V}_j)$) is uniformly, randomly chosen from $\{0.1, 0.2, 0.3, 0.4, 0.5\} * |\{v : v \in \mathcal{V}_j \ \& \ (u, v) \in E\}|$. For GA-WBM-B, the degree constraint of each $u \in U \bigcup V$ is set to 0.3 of the degree of u. The budget ceiling of each $u \in U$ (i.e., $C(u, \mathcal{V}_j)$) is $\left\lfloor 0.8 \sum_{(u,v)\in E, v\in\mathcal{V}_j} W(v) \right\rfloor$.

6.3 Results and Discussion

[2]http://www.gurobi.com/

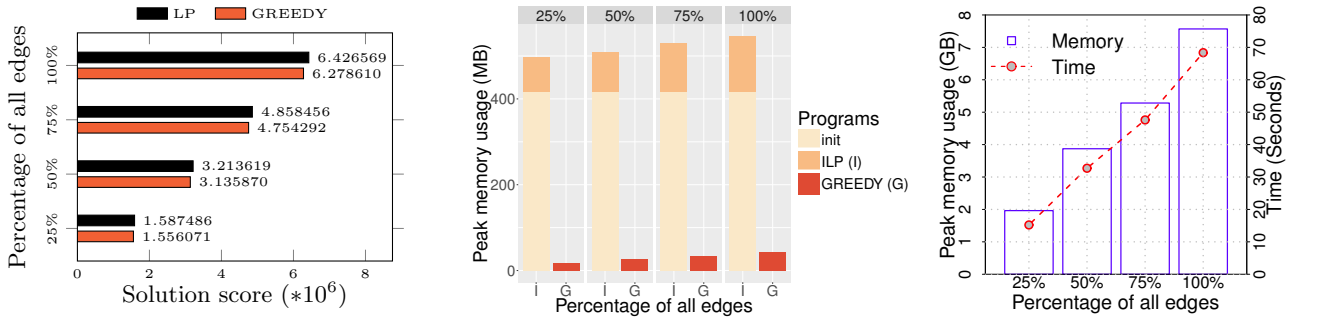
GA-WBM-D Table 2 gives the running times of the LP and the greedy algorithm (GREEDY-D) for the degree-constrained problem on the small eBay Canada dataset. On the 25% sample, GREEDY-D is already $\approx 50\times$ faster than the LP method. As the graph size doubles to 50% and then 100%, the run time of GREEDY-D increases linearly. In contrast, the LP method quadruples run times for every doubling of the graph. At 56,520 edges, GREEDY-D is $\approx 144\times$ faster. The memory consumption for this experiment is shown in Figure 8b. Given the large number of variables in the LP formulation, the solver consumes $\approx 3\times$ more memory than GREEDY-D ($\approx 15\times$ more memory if one includes the constant overhead of loading Matlab, shown by the "init" segment of the LP bar). The memory consumption and quadratic scalability thus limit the graphs that LP can handle.

Figure 8a compares the quality of the solution produced by GREEDY-D to the optimal solution produced by the LP, measured as the score of (i.e., sum of edge weights in) the output subgraph H. GREEDY-D consistently achieves a score that is $\geq 97.5\%$ of the LP score with low variability, a range of 0.0044, in that ratio. This indicates that although GREEDY-D is a 2-approximation, it is able to vastly outperform its theoretical guarantee in practice.

Finally, Figure 8c evaluates GREEDY-D on the large eBay US graph. The input instances—even the 25% sample of this graph which is $50\times$ larger than the 100% sample of eBay Canada used in the previous experiments—are too large for the LP. Observe that the greedy algorithm scales linearly with the dataset in terms of both run time and memory consumption and can handle the full graph, containing 5.8 million vertices and 11 million edges, in roughly one minute.

To summarise the GA-WBM-D results, the LP is reasonably efficient on small graphs and provides an exact solution. For larger graphs where the LP encounters scalability issues both in terms of efficiency and memory consumption, GREEDY-D appears to obtain a $\geq 97.5\%$ quality solution with very fast run times and good, linear scalability.

GA-WBM-B Table 3 gives the run times of the three algorithms for the budget-capped problem on the eBay Canada dataset. Observe that the type of constraints does not influence much the efficiency of our greedy algorithms: GREEDY-B-LF (Algorithm 3) has near identical run times, and hence scalability, as GREEDY-D (c.f., Table 2). The approximate linear program using rounding (LPR) is about $1.9\times$ faster than the ILP, albeit $\approx 8\times$ slower than the greedy algorithm.

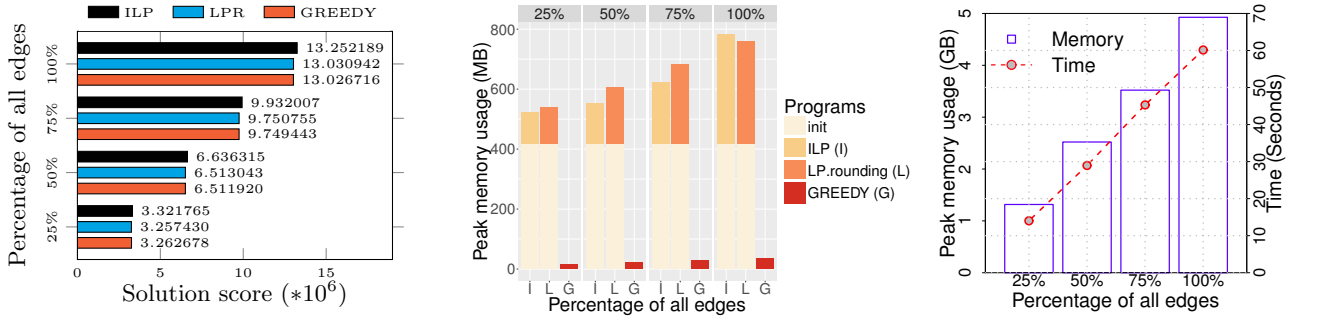


(a) Solution score on 25, 50, 75, 100 % samples of the eBay Canada dataset

(b) Peak memory usage on 25, 50, 75, 100 % samples of the eBay Canada dataset

(c) Scalability of GREEDY-D on 25, 50, 75, 100 % samples of eBay US

Figure 8: Experiment plots for the LP and GREEDY-D algorithms on the degree-constrained problem (GA-WBM-D)



(a) Solution score on 25, 50, 75, 100 % samples of the eBay Canada dataset

(b) Peak memory usage on 25, 50, 75, 100 % samples of the eBay Canada dataset

(c) Scalability of GREEDY-B-LF on 25, 50, 75, 100 % samples of eBay US

Figure 9: Experiment plots for the ILP, LPR, and GREEDY-B algorithms on the budget-capped problem (GA-WBM-B)

Figure 9b shows the peak memory consumption of the algorithms for the previous experiment. As in Figure 8b, the LP-based approaches incur a 400 MB Matlab overhead (the "init" component on the bars), accounting here for half the memory consumption. LPR generally requires slightly more memory than ILP, except on the 100% sample. The better run time scalability of LP on GA-WBM-B, relative to GA-WBM-D, is offset by escalated memory costs: both ILP and LPR require, even excluding the "init" overhead, an order of magnitude more memory than GREEDY-B-LF.

Figure 9a contrasts the solution quality of the approximate algorithms to the exact solution produced by the ILP on the eBay Canada datasets. Both GREEDY-B-LF and LPR obtain 98% of the ILP score on all samples. LPR outperforms the greedy algorithm by 0.01-0.03%, except on the 25% sample, where the greedy algorithm provides the best approximate solution. Considering the large eBay US dataset in Figure 9c, on which the memory consumption is prohibitive for the LP-based algorithms, we observe that GREEDY-B-LF again achieves linear scalability with respect to both run time and memory consumption, albeit with a gentler slope than in the degree-constrained problem (c.f., Figure 8c). On the full graph, the greedy algorithm uses < 5 GB of memory, indicating that the two-level heap structure in the lazy forwarding scheme is quite compact.

To summarise the GA-WBM-B results, the ILP is very efficient on small graphs, producing optimal solutions, but it also demands a lot of memory relative to the greedy algorithm. This impedes its scalability to larger graphs. The greedy algorithm achieves excellent scalability while sacrificing $< 2\%$ of the solution quality and easily scales to 10^7 edges. LPR provides a modest improvement in scalability over ILP, coupled with a modest improvement in solution quality over the greedy algorithm, so is poised to handle boundary instances that are slightly too large for ILP.

7. RELATED WORK

GA-WBM-D is a special case of CA-WBM, first proposed by Chen et al. [10] where the authors presented a generalized formulation of CA-WBM in the context of E-commerce, where diverse matching results are often desired (e.g., movies of different genres and merchants selling products of different categories). They showed that CA-WBM is NP-hard and proposed approximate and randomized algorithms to solve CA-WBM. Since CA-WBM generalizes the classic WBM problem, it has extended applicability in related scientific fields, including resource allocation [5, 28], scheduling [13], Internet advertising [12, 27, 11, 26, 6] and recommender systems [1, 25]. GA-WBM-D is a special case of CA-WBM useful for studying transitive conflicts.

GA-WBM-B generalizes the maximum budgeted allocation (MBA) problem and therefore is NP-hard [23, 16, 2, 34]. For MBA, the integrality gap is 3/4 [2]. Chakrabarty and Goel [9] studied the approximability of MBA and achieved 3/4-approximation ratio by a linear programming based (Assignment-LP) iterative rounding algorithm. They also used hardness reductions to get better hardness results for other allocation problems such as submodular welfare max-

imization (SWM), generalized assignment problem (GAP) and maximum spanning star-forest (MSSF). Kalaitzis [18] obtained an improved $(3/4 + c)$-approximation ratio, for some constant $c > 0$, for MBA by by rounding solutions to an LP called the Configuration-LP. They showed that the Configuration-LP is strictly stronger than the Assignment-LP for MBA. GA-WBM-B is also related to monotone submodular set function maximization subject to the k-system constraint. Fisher, Nemhauser and Wolsey [31] showed that the natural greedy algorithm has a tight approximation ratio of $1/(k+1)$. Călinescu et al. [8] proposed a randomized $(1-1/e)$- approximation for any monotone submodular function and an arbitrary matroid. They also provided performance analysis of the greedy algorithm under the k-system constraint. To improve the efficiency of the greedy approach, Minoux [30] proposed accelerated greedy algorithms. Those techniques were recently used in [24, 25].

8. CONCLUSIONS

In this paper, we investigated generalisations of weighted bipartite B-matching. We gave a new hardness proof for CA-WBM by a reduction from Maximum Weight Independent Set, yielding the stronger result that CA-WBM is hard to approximate. We proposed GA-WBM-D and GA-WBM-B, which partition the right-hand vertex set plus constrain the degree and cap the budget, respectively, of the partitions. While GA-WBM-D can be solved efficiently with linear programming, the number of variables in the linear program creates scalability challenges. GA-WBM-B, on the other hand, is a generalisation of the NP-hard maximum budgeted allocation problem so cannot be solved efficiently. Nonetheless, for both problems we introduced intuitive greedy algorithms with approximation guarantees that, in practice, are nearly as accurate as the linear programs, and moreover can process a dataset with 11.3 million edges in about one minute.

9. ACKNOWLEDGMENTS

This research initiated at the 15th Workshop on Computational Geometry at the Bellairs Research Institute of McGill University and was partially supported by the FRIPRO program of the Norwegian Research Council (ExiBiDa project).

10. REFERENCES

[1] G. Adomavicius and Y. Kwon. Maximizing aggregate recommendation diversity: A graph-theoretic approach. In *Proc. DiveRS*, page 3, 2011.

[2] N. Andelman and Y. Mansour. Auctions with budget constraints. In *Proc. SWAT*, pages 26–38, 2004.

[3] E. M. Arkin and E. B. Silverberg. Scheduling jobs with fixed start and end times. *Discrete Appl. Math.*, 18(1):1–8, 1987.

[4] A. S. Asratian, T. M. J. Denley, and R. Häggkvist. *Bipartite Graphs and Their Applications*. Cambridge University Press, New York, NY, USA, 1998.

[5] D. P. Bertsekas. A new algorithm for the assignment problem. *Math. Program.*, 21(1):152–171, 1981.

[6] A. Bhalgat, N. Korula, H. Leontyev, M. Lin, and V. S. Mirrokni. Partner tiering in display advertising. In *Proc. WSDM*, pages 133–142, 2014.

[7] R. Bhatia, J. Chuzhoy, A. Freund, and N. J. (Seffi). Algorithmic aspects of bandwidth trading. *TALG*, 3(1):10:1–10:19, 2007.

[8] G. Călinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011.

[9] D. Chakrabarty and G. Goel. On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and gap. *SIAM J. Comp.*, 39(6):2189–2211, 2010.

[10] C. Chen, L. Zheng, V. Srinivasan, A. Thomo, K. Wu, and A. Sukow. Conflict-aware weighted bipartite B-matching and its application to e-commerce. *TKDE*, 28(6):1475–1488, 2016.

[11] N. R. Devanur and T. P. Hayes. The adwords problem: online keyword matching with budgeted bidders under random permutations. In *Proc. EC*, pages 71–78, 2009.

[12] B. Edelman, M. Ostrovsky, and M. Schwarz. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *The American economic review*, 97(1):242–259, 2007.

[13] M. Fayyazi, D. Kaeli, and W. Meleis. Parallel maximum weight bipartite matching algorithms for scheduling in input-queued switches. In *Proc. IPDPS*, page 4, 2004.

[14] D. Fleder and K. Hosanagar. Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Manage. Sci.*, 55(5):697–712, 2009.

[15] S. Fujishige. *Submodular functions and optimization*. Annals of discrete mathematics. Elsevier, Amsterdam, Boston, Paris, 2005.

[16] R. Garg, V. Kumar, and V. Pandit. Approximation algorithms for budget-constrained auctions. In *Proc. APPROX*, pages 102–113, 2001.

[17] J. Håstad. Clique is hard to approximate withinn 1- ε. *Acta Mathematica*, 182(1):105–142, 1999.

[18] C. Kalaitzis. An improved approximation guarantee for the maximum budgeted allocation problem. In *Proc. SODA*, pages 1048–1066, 2016.

[19] A. W. J. Kolen, J. K. Lenstra, C. H. Papadimitriou, and F. C. R. Spieksma. Interval scheduling: A survey. *Naval Research Logistics*, 54(5):530–543, 2007.

[20] A. Krause and C. Guestrin. Beyond convexity: Submodularity in machine learning. *ICML Tutorials*, 2008.

[21] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[22] N. Lathia, S. Hailes, L. Capra, and X. Amatriain. Temporal diversity in recommender systems. In *SIGIR*, pages 210–217, 2010.

[23] B. Lehmann, D. Lehmann, and N. Nisan. Combinatorial auctions with decreasing marginal utilities. In *Proc. EC*, pages 18–28, 2001.

[24] J. Leskovec, L. A. Adamic, and B. A. Huberman. The dynamics of viral marketing. *TWEB*, 1(1):5, 2007.

[25] W. Lu, S. Chen, K. Li, and L. V. Lakshmanan. Show me the money: dynamic recommendations for revenue maximization. *PVLDB*, 7(14):1785–1796, 2014.

[26] A. Mehta. Online matching and ad allocation. *Found. and Trends in Theor. Comput. Sci.*, 8(4):265–368, 2013.

[27] A. Mehta, A. Saberi, U. V. Vazirani, and V. V. Vazirani. Adwords and generalized online matching. *J. ACM*, 54(5), 2007.

[28] R. Meir, Y. Chen, and M. Feldman. Efficient parking allocation as online bipartite matching with posted prices. In *Proc. AAMAS*, pages 303–310, 2013.

[29] J. Mestre. Greedy in approximation algorithms. In *ESA*, pages 528–539, 2006.

[30] M. Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In *Proc. Optimization Techniques*, pages 234–243, 1978.

[31] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—I. *Math. Program.*, 14(1):265–294, 1978.

[32] F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *ICML*, pages 784–791, 2008.

[33] K. R. Rebman. Total unimodularity and the transportation problem: a generalization. *Linear Algebra and its Applications*, 8(1):11 – 24, 1974.

[34] T. Sandholm and S. Suri. Side constraints and non-price attributes in markets. *Games and Economic Behavior*, 55(2):321–330, 2006.

[35] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1986.

[36] E. Vee, U. Srivastava, J. Shanmugasundaram, P. Bhat, and S. A. Yahia. Efficient computation of diverse query results. In *ICDE*, pages 228–236, 2008.

[37] M. Yannakakis. On a class of totally unimodular matrices. In *Proc. FOCS*, pages 10–16, Oct 1980.

[38] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *WWW*, pages 22–32, 2005.