Many-to-one matchings with lower quotas: Algorithms and complexity

Ashwin Arulselvan1, Ágnes Cseh2, Martin Groß2, David F. Manlove3, and Jannik Matuschke2

1 Department of Management Science, University of Strathclyde,
e-mail: ashwin.arulselvan@strath.ac.uk
2 Institute for Mathematics, TU Berlin,
e-mail: \{cseh,gross,matuschke\}@math.tu-berlin.de
3 School of Computing Science, University of Glasgow,
e-mail: David.Manlove@glasgow.ac.uk

Abstract. We study a natural generalization of the maximum weight many-to-one matching problem. We are given an undirected bipartite graph \( G = (A \cup P, E) \) with weights on the edges in \( E \), and with lower and upper quotas on the vertices in \( P \). We seek a maximum weight many-to-one matching satisfying two sets of constraints: vertices in \( A \) are incident to at most one matching edge, while vertices in \( P \) are either unmatched or they are incident to a number of matching edges between their lower and upper quota. This problem, which we call maximum weight many-to-one matching with lower and upper quotas (wmlq), has applications to the assignment of students to projects within university courses, where there are constraints on the minimum and maximum numbers of students that must be assigned to each project.

In this paper, we provide a comprehensive analysis of the complexity of wmlq from the viewpoints of classic polynomial time algorithms, fixed-parameter tractability, as well as approximability. We draw the line between \( \text{NP} \)-hard and polynomially tractable instances in terms of degree and quota constraints and provide efficient algorithms to solve the tractable ones. We further show that the problem can be solved in polynomial time for instances with bounded treewidth; however, the corresponding runtime is exponential in the treewidth with the maximum upper quota \( u_{\text{max}} \) as basis, and we prove that this dependence is necessary unless \( \text{FPT} = \text{W}[1] \). Finally, we also present an approximation algorithm for the general case with performance guarantee \( u_{\text{max}} + 1 \), which is asymptotically best possible unless \( \text{P} = \text{NP} \).

Keywords. maximum matching, many-to-one matching, project allocation, inapproximability, bounded treewidth

* Supported by the Deutsche Telekom Stiftung and by COST Action IC1205 on Computational Social Choice. Part of this work was carried out whilst visiting the University of Glasgow.
** Supported by the DFG within project A07 of CRC TRR 154.
*** Supported by EPSRC grant EP/K010042/1.
† Supported by DAAD with funds of BMBF and the EU Marie Curie Actions.
1 Introduction

Many university courses involve some element of team-based project work. A set of projects is available for a course and each student submits a subset of projects as acceptable. For each acceptable student–project pair \((s, p)\), there is a weight \(w(s, p)\) denoting the utility of assigning \(s\) to \(p\). The question of whether a given project can run is often contingent on the number of students assigned to it. Such quota constraints also arise in various other contexts involving the centralized formation of groups, including organizing activity groups at a leisure center, opening facilities to serve a community and coordinating rides within car-sharing systems. In these and similar applications, the goal is to maximize the utility of the assigned agents under the assumption that the number of participants for each open activity is within the activity’s prescribed limits.

We model this problem using a weighted bipartite graph \(G = (A \cup P, E)\), where the vertices in \(A\) represent applicants, while the vertices in \(P\) are posts they are applying to. So in the above student–project allocation example, \(A\) and \(P\) represent the students and projects respectively, and \(E\) represents the set of acceptable student–project pairs. The edge weights capture the cardinal utilities of an assigned applicant–post pair. Each post has a lower and an upper quota on the number of applicants to be assigned to it, while each applicant can be assigned to at most one post. In a feasible assignment, a post is either open or closed: the number of applicants assigned to an open post must lie between its lower and upper quota, whilst a closed post has no assigned applicant. The objective is to find a maximum weight many-to-one matching satisfying all lower and upper quotas. We denote this problem by \(\text{wmlq}\).

In this paper, we study the computational complexity of \(\text{wmlq}\) from various perspectives: Firstly, in Section 2, we show that the problem can be solved efficiently if the degree of every post is at most 2, whereas the problem becomes hard as soon as posts with degree 3 are permitted, even when lower and upper quotas are all equal to the degree and every applicant has a degree of 2. Furthermore, we show the tractability of the case of pair projects, i.e., when all upper quotas are at most 2. Then, in Section 3, we study the fixed parameter tractability of \(\text{wmlq}\). To this end, we generalize the known dynamic program for maximum independent set with bounded treewidth to \(\text{wmlq}\). The running time of our algorithm is exponential in the treewidth of the graph, with \(u_{\text{max}}\), the maximum upper quota of any vertex, as the basis. This yields a fixed-parameter algorithm when parameterizing by both the treewidth and \(u_{\text{max}}\). We show that this exponential dependence on the treewidth cannot be completely separated from the remaining input by establishing a \(W[1]\)-hardness result for \(\text{wmlq}\) parameterized by treewidth. Finally, in Section 4, we discuss the approximability of the problem. We show that a simple greedy algorithm yields an approximation guarantee of \(u_{\text{max}} + 1\) for \(\text{wmlq}\) and \(\sqrt{|A|} + 1\) in the case of unit edge weights. We complement these results by showing that these approximation factors are asymptotically best possible, unless \(P = \text{NP}\).
Related work

Among various applications of centralized group formation, perhaps the assignment of medical students to hospitals has received the most attention. In this context, as well as others, the underlying model is a bipartite matching problem involving lower and upper quotas. The *Hospitals / Residents problem with Lower Quotas* (HRLQ) is a variant of WMLQ where applicants and posts have ordinal preferences over one another, and we seek a stable matching of residents to hospitals. Hamada et al. [11] considered a version of HRLQ where hospitals cannot be closed, whereas the model of Biró et al. [3] permitted hospital closures. Strategyproof mechanisms have also been studied in instances with ordinal preferences and no hospital closure [10].

The *Student / Project Allocation problem* [19, Section 5.6] models the assignment of students to projects offered by lecturers subject to upper and lower quota restrictions on projects and lecturers. Several previous papers have considered the case of ordinal preferences involving students and lecturers [11, 13, 20] but without allowing lower quotas. However, two recent papers [14, 21] do permit lower quotas together with project closures, both in the absence of lecturer preferences. Monte and Tumennasan [21] considered the case where each student finds every project acceptable, and showed how to modify the classical Serial Dictatorship mechanism to find a Pareto optimal matching. Kamiyama [14] generalized this mechanism to the case where students need not find all projects acceptable, and where there may be additional restrictions on the sets of students that can be matched to certain projects. This paper also permits lower quotas and project closures, but our focus is on cardinal utilities rather than ordinal preferences.

The unit-weight version of WMLQ is closely related to the *D-matching problem* [7, 17, 20], a variant of graph factor problems [24]. In an instance of the D-matching problem, we are given a graph G, and a domain of integers is assigned to each vertex. The goal is to find a subgraph G′ of G such that every vertex has a degree in G′ that is contained in its domain. Lovász [16] showed that the problem of deciding whether such a subgraph exists is NP-complete, even if each domain is either {1} or {0, 3}. On the other hand, some cases are tractable. For example, if for each domain D, the complement of D contains no consecutive integers, the problem is polynomially solvable [26]. As observed in [25], D-matchings are closely related to *extended global cardinality constraints* and the authors provide an analysis of the fixed-parameter tractability of a special case of the D-matching problem; see Section 3 for details.

The problem that we study in this paper corresponds to an optimization version of the D-matching problem. We consider the special case where G is bipartite and the domain of each applicant vertex is {0, 1}, whilst the domain of each post vertex p is {0} ∪ {ℓ(p), . . . , u(p)}, where ℓ(p) and u(p) denote the lower and upper quotas of p respectively. Since the empty matching is always feasible in our case, our aim is to find a domain-compatible subgraph G′ such that the total weight of the edges in G′ is maximum.
2 Degree- and quota-restricted cases

First, we provide a formal definition of the maximum weight many-to-one matching problem with lower quotas (WMLQ). Then, we characterize the complexity of the problem in terms of degree constraints on the two vertex sets: applicants and posts. At the end, we discuss the case of bounded upper quota constraints.

2.1 Problem definition

In our problem, a set of applicants $A$ and a set of posts $P$ are given. $A$ and $P$ constitute the two vertex sets of an undirected bipartite graph $G = (V, E)$ with $V = A \cup P$. For a vertex $v \in V$ we denote by $\delta(v) = \{\{v, w\} \in E : w \in V\}$ the set of edges incident to $v$ and by $\Gamma(v) = \{w \in V : \{v, w\} \in E\}$ the neighborhood of $v$, i.e., the set of vertices that are adjacent to $v$. For a subset of vertices $V' \subset V$, we define $\delta(V') = \bigcup_{v \in V'} \delta(v)$. Each edge carries a weight $w : E \to \mathbb{R}_{\geq 0}$, representing the utility of the corresponding assignment. Each post is equipped with a lower quota $\ell : P \to \mathbb{Z}_{\geq 0}$ and an upper quota $u : P \to \mathbb{Z}_{\geq 0}$ so that $\ell(p) \leq u(p)$ for every $p \in P$. These functions bound the number of admissible applicants for the post (independent of the weight of the corresponding edges). Furthermore, every applicant can be assigned to at most one post. Thus, an assignment is a subset $M \subseteq E$ of the edges such that $|\delta(a) \cap M| \leq 1$ for every applicant $a \in A$ and $|\delta(p) \cap M| \in \{0, \ell(p), \ell(p) + 1, \ldots, u(p)\}$ for every $p \in P$. A post is said to be open if the number of applicants assigned to it is greater than 0, and closed otherwise. The size of an assignment $M$, denoted $|M|$, is the number of assigned applicants, while the weight of $M$, denoted $w(M)$, is the total weight of the edges in $M$, i.e., $w(M) = \sum_{e \in M} w(e)$. The goal is to find an assignment of maximum weight.

Remark 1. Note that when not allowing closed posts, the problem immediately becomes tractable. It is easy to see this in the unweighted case as any algorithm for maximum flow with lower capacities can be used to determine an optimal solution in polynomial time. This problem can be easily reduced to the classical maximum flow problem. The method can be naturally extended to the weighted case as the flow based linear program has integral extreme points due to its total unimodularity property.

Problem 1 WMLQ

Input: $I = (G, w, \ell, u)$; a bipartite graph $G = (A \cup P, E)$ with edge weights $w$.
Task: Find an assignment of maximum weight.
If $w = 1$ for all $e \in E$, we refer to the problem as MLQ.

Some trivial simplification of the instance can be executed right at start. If $u(p) > |\Gamma(p)|$ for a post $p$, then $u(p)$ can be replaced by $|\Gamma(p)|$. On the other hand, if $\ell(p) > |\Gamma(p)|$, then post $p$ can immediately be deleted, since no feasible solution can satisfy the lower quota condition. Moreover, posts with $\ell(p) = 1$ behave identically to posts without a lower quota. From now on we assume that the instances have already been simplified this way.
2.2 Degree-restricted cases

In this subsection, we will consider \( wmlq(i, j) \), a special case of \( wmlq \), in which we restrict us to instances in which every applicant submits at most \( i \) applications and every post receives at most \( j \) applications. In order to establish our first result, we reduce the maximum independent set problem (\( \text{MIS} \)) to \( \text{MLQ} \). In \( \text{MIS} \), a graph with \( n \) vertices and \( m \) edges is given and the task is to find an independent vertex set of maximum size. \( \text{MIS} \) is not approximable within a factor of \( n^{1-\varepsilon} \) for any \( \varepsilon > 0 \), unless \( \text{P} = \text{NP} \) [29]. The problem remains APX-complete even for cubic (3-regular) graphs [2].

**Theorem 1.** \( \text{MLQ}(2,3) \) is APX-complete.

**Proof.** First of all, \( \text{MLQ}(2,3) \) is in APX because feasible solutions are of polynomial size and the problem has a 4-approximation (see Theorem 7).

To each instance \( I \) of \( \text{MIS} \) on cubic graphs we create an instance \( I' \) of \( \text{MLQ} \) such that there is an independent vertex set of size at least \( K \) in \( I \) if and only if \( I' \) admits an assignment of size at least \( 3K \), yielding an approximation-preserving reduction. The construction is as follows. To each of the \( n \) vertices of graph \( G \) in \( I \), a post with upper and lower quota of 3 is created. The \( m \) edges of \( G \) are represented as \( m \) applicants in \( I' \). For each applicant \( a \in A \), \( |\Gamma(a)| = 2 \) and \( \Gamma(a) \) comprises the two posts representing the two end vertices of the corresponding edge. Since we work on cubic graphs, \( |\Gamma(p)| = 3 \) for every post \( p \in P \).

First we show that an independent vertex set of size \( K \) can be transformed into an assignment of at least \( 3K \) applicants. All we need to do is to open a post with its entire neighborhood assigned to it if and only if the vertex representing that post is in the independent set. Since no two posts stand for adjacent vertices in \( G \), their neighborhoods do not intersect. Moreover, the assignment assigns exactly three applicants to each of the \( K \) open posts.

To establish the opposite direction, let us assume that an assignment of cardinality at least \( 3K \) is given. The posts’ upper and lower quota are both set to 3, therefore, the assignment involves at least \( K \) open posts. No two of them can represent adjacent vertices in \( G \), because then the applicant standing for the edge connecting them would be assigned to both posts at the same time.

The reduction given here is an \( L \)-reduction [23] with constants \( \alpha = \beta = 3 \). Since \( \text{MLQ}(2,3) \) belongs to APX and \( \text{MIS} \) is APX-complete in cubic graphs, it follows that \( \text{MLQ}(2,3) \) is APX-complete. \( \square \)

So far we have established that if \( |\Gamma(a)| \leq 2 \) for every applicant \( a \in A \) and \( |\Gamma(p)| \leq 3 \) for every post \( p \in P \), then \( \text{MLQ} \) is NP-hard. In the following, we also show that these restrictions are the tightest possible. If \( |\Gamma(p)| \leq 2 \) for every post \( p \in P \), then a maximum weight matching can be found efficiently, regardless of \( |\Gamma(a)| \). Note that the case \( \text{WMLQ}(1,\infty) \) is trivially solvable.

**Theorem 2.** \( \text{WMLQ}(\infty, 2) \) is solvable in \( O(n^2 \log n) \) time, where \( n = |A| + |P| \).

**Proof.** After executing the simplification steps described after the problem definition, we apply two more changes to derive our helper graph \( H \). Firstly, if
\[ \ell(p) = 0, u(p) = 2 \text{ and } |\Gamma(p)| = 2, \]

we separate \( p \)'s two edges, splitting \( p \) into two posts with upper quota 1. After this step, all posts with \( u(p) = 2 \) also have \( \ell(p) = 2 \). All remaining vertices are of upper quota 1. Then, we substitute all edge pairs of posts with \( \ell(p) = u(p) = 2 \) with a single edge connecting the two applicants. This edge will carry the weight equal to the sum of the weights of the two deleted edges.

Clearly, any matching in \( H \) translates into an assignment of the same weight in \( G \) and vice versa. Finding a maximum weight matching in a general graph with \( n \) vertices and \( m \) edges can be done in \( O(n(m + n \log n)) \) time [9], which reduces to \( O(n^2 \log n) \) in our case.

### 2.3 Quota-restricted cases

In this section, we address the problem of \( \text{wmlq} \) with bounded upper quotas. Note that Theorem 1 already tells us that the case of \( u(p) \leq 3 \) for all posts \( p \in P \) is \( \text{NP} \)-hard to solve. We will now settle the complexity of the only remaining case, where we have instances with every post \( p \in P \) having an arbitrary degree and \( u(p) \leq 2 \). This setting models posts that need to be assigned to pairs of applicants.

The problem is connected to various known problems in graph theory, one of them being the \( S \)-path packing problem. In that problem, we are given a graph with a set of terminal vertices \( S \). The task is to pack the highest number of vertex-disjoint paths so that each path starts and ends at a terminal vertex, while all its inner vertices are non-terminal. The problem can be solved in \( O(n^{2.38}) \) time [6,27] with the help of matroid matching [18]. An instance of \( \text{mlq} \) with \( \ell(p) = u(p) = 2 \) for every post \( p \in P \) corresponds to an \( S \)-path packing instance with \( S = A \). The highest number of vertex-disjoint paths starting and ending in \( A \) equals half of the cardinality of a maximum assignment. Thus, \( \text{mlq} \) with \( \ell(p) = u(p) = 2 \) can also be solved in \( O(n^{2.38}) \) time. On the other hand, there is no straightforward way to model posts with \( u(p) = 1 \) in \( S \)-path packing and introducing weights to the instances also seems to be a challenging task. Some progress has been made for weighted edge-disjoint paths, but to the best of our knowledge the question is unsettled for vertex-disjoint paths [12].

Here we present a solution for the general case \( \text{wmlq} \) with \( u(p) \leq 2 \). Our algorithm is based on \( f \)-factors of graphs. In the \( f \)-factor problem, a graph \( G \) and a function \( f : V \rightarrow \mathbb{Z}_{\geq 0} \) is given. A set of edges \( F \subseteq E \) is called an \( f \)-factor if \( \deg_F(v) = f(v) \) for every \( v \in V \), where \( \deg_F(v) \) denotes the degree of \( v \) in the graph \( (V, F) \). Constructing an \( f \)-factor of maximum weight in a graph with \( n \) vertices and \( m \) edges or proving that none exists can be done in \( O(\phi(m + n \log n)) \) time, where \( \phi \) is the sum of all \( f \)-values in the graph [8,9].

**Theorem 3.** \( \text{wmlq} \) with \( u(p) \leq 2 \) for every \( p \in P \) can be solved in \( O(nm + n^2 \log n) \) time, where \( n = |V| \) and \( m = |E| \).

**Proof.** In the remainder of this section we assume that \( 1 \leq \ell(p) = u(p) \leq 2 \) for every post \( p \). Posts with \( \ell(p) \neq u(p) = 2 \) can be transformed into posts with
\( \ell(p) = u(p) = 2 \) by giving them a dummy edge with zero weight, allowing us to pick these edges in order to make up for the raised lower quota. Let us denote the set of posts with \( \ell(p) = u(p) = 1 \) by \( P_1 \).

The graph \( G' = (V', E') \) of the constructed \( f \)-factor instance contains the graph \( G = (V, E) \) of our \( wmlq \) instance, as shown in Figure 1. We add a dummy post \( p_d \) to \( V \) and connect it to every applicant in \( A \). For every post \( p_i \) in \( P_1 \) we add a dummy vertex \( q_{1i} \) and connect \( p_i \) to \( p_d \) with a path of length 2, and for every post \( p_i \) with \( \ell(p_i) = u(p_i) = 2 \) we add two dummy vertices \( q_1^i \) and \( q_2^i \) and a triangle containing \( p \) and two new dummy nodes. All new edges in \( E' \setminus E \) carry zero weight.

We set \( f(p_d) = K, f(p) = u(p) \) for every \( p \in P \) and \( f(v) = 1 \) for the rest of the vertices. In the initial version of our algorithm, we solve a weighted \( f \)-factor problem for every \( K \in \{0, 1, ..., |A| + |P_1|\} \), and later we will show a slightly modified version of the \( f \)-factor instance so that it is sufficient to construct only 2 instances.

First we show that if there is a feasible assignment \( M \) in \( G \) so that the number of unmatched applicants and the number of matched posts in \( P_1 \) add up to \( K \), then it can be extended to an \( f \)-factor \( M' \) of the same weight in \( G' \). We construct \( M' \) starting with \( M \) and add the following edges to it:

- \( \{p_d, a_i\} \) for every applicant \( a_i \) that is unmatched in \( M \);
- \( \{q_1^i, p_i\} \) and \( \{q_2^i, p_i\} \) for every post \( p_i \notin P_1 \) that is closed in \( M \);
- \( \{q_1^i, q_2^i\} \) for every post \( p_i \notin P_1 \) that is open in \( M \);
For all vertices \( v \neq p_d \), it immediately follows from the construction that 
\( \deg_{M'}(v) = f(v) \). The same holds for \( p_d \) as well, because edges are assigned to it if and only if an applicant is unmatched or a post in \( P_1 \) is matched and we assumed that this adds up to \( K \).

It is easy to see that if there is an \( f \)-factor \( M' \) in \( G' \), then its restriction to \( G \) is a feasible assignment \( M \) of the same weight so that the number of unmatched applicants and the number of matched posts in \( P_1 \) add up to \( K \). Since \( f(q_i^1) = 1 \) for every \( q_i^1 \) added to a post \( p_i \in P_1 \), it is either the case that \( p_i \) is closed in \( M \) or \( p_d q_i^1 \in M' \). Regarding posts outside of \( P_1 \), we need to show that the two edges incident to them are either both in \( G \) or neither of them are in \( G \). Assume without loss of generality that \( p_i q_i^1 \in M' \) and \( p_d q_i^1 \notin M' \) for some \( p_i \notin P_1 \). Since \( f(q_i^2) = 1 \) and \( \deg_{M'}(q_i^2) = 0 \), \( M' \) cannot be an \( f \)-factor.

So far we have shown that it is sufficient to test \( |A| + |P_1| + 1 \) values for \( f(p_d) \), and collect the optimal assignments given by the maximum weight \( f \)-factors. Comparing the weight of these locally optimal solutions delivers a global optimum. A slight modification on the \( f \)-factor instance will allow us to construct only two instances. Similarly to the triangles attached to posts in \( P \setminus P_1 \), triangles are added to \( p_d \) as well. The added vertices have \( f \)-value 1 and the added edges carry weight 0. The number of such triangles hanging on \( p_d \) is \( \left\lceil \frac{|A| + |P_1|}{2} \right\rceil \). These triangles can take up all the \( f \)-value of \( p_d \) if necessary, but choosing the edge not incident to \( p_d \) they can also let \( p_d \) to fill up its \( f \)-value with other edges. Since a triangle either takes up 0 or 2 of \( p_d \)'s \( f \)-value, we need to separate the two different parity cases. Thus, to cover all the \( |A| + |P_1| + 1 \) cases for possible values for \( f(p_d) \), once we set \( f(p_d) \) to \( |A| + |P_1| + 1 \) and in the other instance \( f(p_d) = |A| + |P_1| \).

\( \square \)

### 3 Bounded treewidth graphs

In this section, we investigate \textsc{wmlq} from the point of view of fixed-parameter tractability and analyze how efficiently the problem can be solved for instances with a bounded treewidth.

*Fixed-parameter tractability.* This field of complexity theory is motivated by the fact that in many applications of optimization problems certain input parameters stay small even for large instances. A problem, parameterized by a parameter \( k \), is fixed-parameter tractable (FPT) if there is an algorithm solving it in time \( f(k) \cdot \phi(n) \), where \( f : \mathbb{R} \to \mathbb{R} \) is a function, \( \phi \) is a polynomial function, and \( n \) is the input size of the instance. Note that this definition not only requires that the problem can be solved in polynomial time for instances where \( k \) is bounded by a constant, but also that the dependence of the running time on \( k \) is separable from the part depending on the input size. On the other hand, if a problem is shown to be \( W[1] \)-hard, then the latter property can only be fulfilled if \( \text{FPT} = W[1] \),
which would imply $\textbf{NP} \subseteq \textbf{DTIME}(2^{o(n)})$. For more details on fixed-parameter algorithms see, e.g., [22].

**Treewidth.** In case of WMLQ we focus on the parameter *treewidth*, which, on an intuitive level, describes the likeness of a graph to a tree. A *tree decomposition* of graph $G$ consists of a tree whose nodes—also called *bags*—are subsets of $V(G)$. These must satisfy the following three requirements.

1. Every vertex of $G$ belongs to at least one bag of the tree.
2. For every edge $\{a, p\} \in E(G)$, there is a bag containing both $a$ and $p$.
3. If a vertex in $V(G)$ occurs in two bags of the tree, then it also occurs in all bags on the unique path connecting them.

The *width* of a tree decomposition with a set of bags $B$ is $\max_{b \in B} |b| - 1$. The *treewidth* of a graph $G$, $\text{tw}(G)$, is the smallest width among all tree decompositions of $G$. It is well known that a tree decomposition of smallest width can be found by a fixed-parameter algorithm when parameterized by $\text{tw}(G)$ [4].

In the following, we show that WMLQ is fixed-parameter tractable when parameterized simultaneously by the treewidth and $u_{\text{max}}$, whereas it remains $W[1]$-hard when only parameterized by the treewidth. A similar study of the fixed-parameter tractability of the related *extended global cardinality constraint problem* (EGCC) was conducted in [25]. EGCC corresponds to the special case of the $D$-matching problem where the graph is bipartite and on one side of the bipartition all vertices have the domain $\{1\}$. Differently from WMLQ, EGCC is a feasibility problem (note that the feasibility version of WMLQ is trivial, as the empty assignment is always feasible). The authors of [25] provide a fixed-parameter algorithm for EGCC when parameterized simultaneously by the treewidth of the graph and the maximum domain size, and they show that the problem is $W[1]$-hard when only parameterized by the treewidth. These results mirror our results for WMLQ, and indeed both our FPT-algorithm for WMLQ and the one in [25] are extensions of the same classic dynamic program for the underlying maximum independent set problem. However, our hardness result uses a completely different reduction than the one in [25]. The latter makes heavy use of the fact that the domains can be arbitrary sets, whereas in WMLQ, we are confined to intervals.

**Algorithm for bounded treewidth graphs**

For every tree decomposition with a specific treewidth, a nice tree decomposition of the same treewidth can be found in linear time [17]. A nice tree decomposition is characterized by an exclusive composition of the following four kinds of bags:

1. leaf bag: $|b| = 1$ and $b$ has no child;
2. introduce bag: $b$ has exactly one child $b_1$, so that $b_1 \subset b$ and $|b \setminus b_1| = 1$;
3. forget bag: $b$ has exactly one child $b_1$, so that $b \subset b_1$ and $|b_1 \setminus b| = 1$;
4. join bag: $b$ has exactly two children $b_1$ and $b_2$, so that $b = b_1 = b_2$. 


Proof. Note that Lemma 2. Let \( \alpha \in \mathbb{Z}^U \) for some \( U \subseteq V \). We define the notion of extension and restriction of such a vector \( \alpha \). For \( U' \subseteq U \) and \( \alpha \in \mathbb{Z}^U \) define \( \alpha|_{U'} \) as the restriction of \( \alpha \) to \( U' \), i.e., \( \alpha|_{U'} \in \mathbb{Z}^{U'} \) and \( \alpha|_{U'}(v) = \alpha(v) \) for all \( v \in U' \). For \( v \in V \setminus U \) and \( i \in \mathbb{Z} \) let further \( [\alpha, i]_v \) be the extension of \( \alpha \) to \( U \cup \{v\} \) defined by \( [\alpha, i]_v(v') := \alpha(v') \) for all \( v' \in U \) and \( [\alpha, i]_v(v) := i \). For a set of edges \( S \subseteq E_b \) we define \( \alpha_{S,U}(v) \in \mathbb{Z}^U \) by \( \alpha_{S,U}(v) := [\delta(v) \cap S], \) for all \( v \in U \).

Assignment vectors. For any bag \( b \), let \( V_b \subseteq V \) denote the set of vertices contained in the union of bags present in the subtree rooted at \( b \). We will define graph \( G_b = (V_b, E_b) \) where \( E_b := E[V_b] \setminus E[b] \). A partial assignment for bag \( b \) is an assignment \( M \subseteq E_b \) of \( G_b \) such that \( |M \cap \delta(v)| = 0 \) or \( \ell(v) \leq |M \cap \delta(v)| \leq u(v) \) for all \( v \in V_b \setminus b \). An assignment vector for bag \( b \) is a vector \( \alpha \in X_b := \{0, \ldots, u_{\text{max}}\}^b \). We say a partial assignment \( M \) for \( b \) agrees with an assignment vector \( \alpha \in X_b \), if \( \alpha(v) = |M \cap \delta(v)| \) for all \( v \in b \). For every bag \( b \) and every \( \alpha \in X_b \), let \( M_b(\alpha) \) be the set of partial assignments for \( b \) that agree with \( \alpha \) and let \[
W_b(\alpha) := \max \{w(M) : M \in M_b(\alpha)\} \cup \{-\infty\}
\]
denote the optimal value of any assignment that agrees with \( \alpha \) for the graph \( G_b \) (note that a value of \( -\infty \) implies that \( \alpha \) cannot be attained). We further denote \[
M_b^*(\alpha) := \{M \in M_b(\alpha) : w(M) = W_b(\alpha)\}.
\]

In the following, we will provide a series of lemmas that reveals how to efficiently obtain an element of \( M_b^*(\alpha) \) for every \( \alpha \in X_b \) for a bag \( b \) (or showing \( W_b(\alpha) = -\infty \)), assuming such representatives of each set \( M_b^*(\alpha) \) have already been computed for every child \( b' \) of \( b \) for all \( \alpha \in X_{b'} \).

Lemma 1. Let \( b \) be a leaf bag. Then \( M_b^*(\alpha) = \{\emptyset\} \) and \( M_b^*(\alpha) = \emptyset \) for any \( \alpha \in X_b \setminus \{0\} \).

Proof. This follows directly from the fact that \( E_b = \emptyset \) for all leaf bags and thus the only element in \( b \) cannot be assigned. \( \square \)

Lemma 2. Let \( b \) be an introduce bag such that \( b' \) is the only child of \( b \) and \( b \setminus b' = \{v'\} \). Let \( \alpha \in X_b \). Then \[
M_b^*(\alpha) = \begin{cases} M_{b'}^*(\alpha|_{b'}) & \text{if } \alpha(v') = 0, \\ \emptyset & \text{otherwise.} \end{cases}
\]

Proof. Note that \( \Gamma(v') \cap V_b \subseteq b \) by Properties 2 and 3 of the tree decomposition. This implies \( \delta(v') \cap E_b = \emptyset \) and hence the lemma. \( \square \)

\(^4\) Note that this definition allows applicants and posts in \( b \) to be assigned arbitrarily often and that by definition of \( G_b \), no vertex in \( b \) is assigned to another vertex in \( b \).
Lemma 3. Let $b$ be a forget bag such that $b'$ is the unique child of $b$ and $b = b' \setminus \{v'\}$ for some $v' \in b'$. Let $\alpha \in X_b$. Let $(S^*, i^*)$ be an optimal solution to
\[
[\text{forget}] \quad \max \ w(S) + W_{b'}(\alpha - \alpha_{S, b}, i - |S|_{v'}) \\
\text{s.t.} \quad |S| \leq i, \\
\alpha_{S, b} \leq \alpha \\
S \subseteq \delta(v') \cap \delta(b), \\
i \in \{0, \ell(v'), \ldots, u(v')\}.
\]
Then $M \cup S^* \in \mathcal{M}_b^*(\alpha)$ for all $M \in \mathcal{M}_b^*([\alpha - \alpha_{S, b}, i - |S|_{v'}])$. If the optimal solution to [forget] has value $-\infty$, then $\mathcal{M}_b^*(\alpha) = \emptyset$.

Proof. Assume $\mathcal{M}_b(\alpha) \neq \emptyset$ and let $M' \in \mathcal{M}_b^*(\alpha)$. Let $S' := M' \cap \delta(v') \cap \delta(b)$ and let $i' := |M' \cap \delta(v')|$. Observe that $(S', i')$ is a feasible solution to [forget] and that $M' \setminus S' \in \mathcal{W}_b([\alpha - \alpha_{S', b}, i' - |S'|_{v'}])$. We conclude that $w(M') \leq w(S') + W_{b'}(\alpha, i' - |S'|_{v'}) \leq w(S^*) + W_{b'}([\alpha, i^* - |S^*|_{v'}])$. In particular, this implies that the optimal solution value of [forget] is finite and that $\mathcal{M}^*([\alpha - \alpha_{S', b}, i^* - |S^*|_{v'}]) \neq \emptyset$.

Thus let $M^* := M \cup S^*$ for some $M \in \mathcal{M}^*([\alpha - \alpha_{S', b}, i^* - |S^*|_{v'}])$. Observe that indeed $|M^* \cap \delta(v)| = |M \cap \delta(v)| + |S^* \cap \delta(v)| = \alpha(v) - \alpha_{S', b}(v) + \alpha_{S', b}(v) = \alpha(v)$ for all $v \in b$. Furthermore $|M^* \cap \delta(v)| = |M \cap \delta(v)| \in \{0, \ell(v), \ldots, u(v)\}$ for all $v \in V_b \setminus b'$ by feasibility of $M$ and $|M^* \cap \delta(v')| = i^* \in \{0, \ell(v'), \ldots, u(v')\}$, implying $M^* \in \mathcal{M}_b(\alpha)$. As $w(M^*) = w(S^*) + W_{b'}([\alpha, i^* - |S^*|_{v'}]) \geq w(M')$, we conclude that indeed $M^* \in \mathcal{M}_b^*(\alpha)$.

Lemma 4. Let $b$ be a join bag such that $b = b_1 = b_2$ for the two children $b_1, b_2$ of $b$. Let $\alpha \in X_b$. Let $(\alpha_1^*, \alpha_2^*)$ be optimal solutions to
\[
[\text{join}] \quad \max \ W_{b_1}(\alpha_1) + W_{b_2}(\alpha_2) \\
\text{s.t.} \quad \alpha_1 + \alpha_2 = \alpha \\
\alpha_1 \in X_{b_1}, \ \alpha_2 \in X_{b_2}
\]
Then $M_1 \cup M_2 \in \mathcal{M}_b^*(\alpha)$ for all $M_1 \in \mathcal{M}_{b_1}^*(\alpha_1)$, $M_2 \in \mathcal{M}_{b_2}^*(\alpha_2)$. If the optimal solution of [join] has value $-\infty$, then $\mathcal{M}_b^*(\alpha) = \emptyset$.

Proof. Let $M^* := M_1 \cup M_2$ for some $M_1 \in \mathcal{M}_{b_1}^*(\alpha_1)$, $M_2 \in \mathcal{M}_{b_2}^*(\alpha_2)$. We first observe that $V_{b_1} \cap V_{b_2} = b$ by Properties 2 and 3 of the tree decomposition and hence $M_1 \cap M_2 = \emptyset$. This implies that
\[
|M^* \cap \delta(v)| = \begin{cases} 
|M_1 \cap \delta(v)| \in \{0, \ell(v), \ldots, u(v)\} & \text{if } v \in V_{b_1} \setminus b, \\
|M_2 \cap \delta(v)| \in \{0, \ell(v), \ldots, u(v)\} & \text{if } v \in V_{b_2} \setminus b, \\
|M_1 \cap \delta(v)| + |M_2 \cap \delta(v)| = \alpha(v) & \text{if } v \in b.
\end{cases}
\]
This implies $M^* \in \mathcal{M}_b(\alpha)$.

Now let $M^* \in \mathcal{M}_b(\alpha)$. Let $M_1' := M' \cap E_{b_1}$ and $M_2' := M' \cap E_{b_2}$. For $i \in \{1, 2\}$ let $\alpha_i' \in \mathbb{Z}^b$ be defined by $\alpha_i'(v) := |M_i' \cap \delta(v)|$ for all $v \in b_i$. We observe that $(\alpha_1', \alpha_2')$ is a feasible solution to [join] and hence $w(M') = w(M_1') + w(M_2') \leq w(M_1) + w(M_2) = w(M^*)$. \qed
Finally, we observe that after computing $W_r(\alpha)$ and corresponding elements of $\mathcal{M}_r(\alpha)$ for each $\alpha$ for the root bag $r$, an optimal assignment for $G$ can be easily obtained.

**Lemma 5.** Let $(S^*, \alpha^*)$ be an optimal solution of

$$[\text{root}] \quad \max \ W_r(\alpha) + w(S)$$

s.t. $\alpha(v) + |\delta(v) \cap S| \in \{0, \ell(v), \ldots, u(v)\}$ \quad $\forall v \in r$

$\alpha \in X_r, \ S \subseteq E[r].$

Then $S^* \cup M$ is an optimal solution to WMLQ for any $M \in \mathcal{M}_r(\alpha^*)$.

**Proof.** Let $M^* := S^* \cup M$ for some $M \in \mathcal{M}_r(\alpha^*)$. Note that $S^* \cap \delta(v) = \emptyset$ for $v \in V \setminus r$ and in this case $|M^* \cap \delta(v)| = |\delta(v) \cap M| \in \{0, \ell(v), \ldots, u(v)\}$ by feasibility $M$. On the other hand, if $v \in r$, then $|M^* \cap \delta(v)| = \alpha^*(v) + |S^* \cap \delta(v)| \in \{0, \ell(v), \ldots, u(v)\}$ by feasibility of $S^*$ for [root]. We conclude that $M^*$ is indeed a feasible solution to WMLQ.

Now let $M' \subseteq E$ be any solution to WMLQ and let $\alpha' \in Z^r$ be defined by $\alpha'(v) := |M' \cap \delta(v)|$ for all $v \in r$. Define $S' := M' \cap E[r]$. Observe that $(S', \alpha' - \alpha'_{S', r})$ is a feasible solution to [root] and $M' \setminus S' \in \mathcal{M}_r(\alpha' - \alpha'_{S', r})$. We conclude that

$$w(M') \leq W_r(\alpha') + w(S') \leq W_r(\alpha^*) + w(S^*) = w(M^*),$$

and thus $M^*$ is indeed an optimal solution to WMLQ. \hfill \Box

**Theorem 4.** WMLQ can be solved in time $O(T + (u_{\max})^{3\text{tw}(G)}|E|)$, where $T$ is the time needed for computing a tree decomposition of $G$. In particular, WMLQ can be solved in polynomial time when restricted to instances of bounded treewidth, and WMLQ parameterized by $\max\{\text{tw}(G), u_{\max}\}$ is fixed-parameter tractable.

**Proof.** In order to solve a given WMLQ instance, the algorithm starts by computing a nice tree decomposition of $G$. Note that $T$ is of the same order for tree decompositions and nice tree decompositions. Using Lemmas 1 to 5, we can inductively compute a representative $M \in \mathcal{M}_r(\alpha)$ for every bag $b$ and every $\alpha \in X_b$, or deduce that $\mathcal{M}_r(\alpha) = \emptyset$. We first observe that $|X_b| = (u_{\max})^{\text{tw}(G)}$, thus only $(u_{\max})^{\text{tw}(G)}$ representatives have to be computed per bag. Furthermore, for each of the above lemmas, the necessary computations to derive an $M \in \mathcal{M}_r(\alpha)$ from representatives $\mathcal{M}_r(\alpha')$ of children $b'$ of $b$ can be done in time $O((u_{\max})^{2\text{tw}(G)+1})$. This is obvious for Lemmas 1 and 2. For Lemmas 3 to 5 we observe that the sets of feasible solutions for the corresponding optimization problems [forget], [join], and [root] have size at most $2|b| \times (u_{\max} + 1)$, $(u_{\max})^{2\text{tw}(G)}$, and $2|v|^2 \times (u_{\max})^{\text{tw}(G)}$, respectively (note that without loss of generality we can assume $|v|$ to be of constant size by introducing at most $\text{tw}(G)$ additional forget bags). The theorem then follows from the fact that the number of bags is linear. \hfill \Box
While our algorithm runs in polynomial time for bounded treewidth, the degree of the polynomial depends on the treewidth and the algorithm only becomes a fixed-parameter algorithm when parameterizing by treewidth and \( u_{\text{max}} \) simultaneously. We will now show by a reduction from Minimum Maximum Outdegree that this dependence is necessary under the assumption that FPT \( \neq \mathcal{W}[1] \).

**Problem 2** Minimum Maximum Outdegree

**Input:** A graph \( G = (V, E) \), edge weights \( w : E \to \mathbb{Z}_+ \) encoded in unary, a degree-bound \( r \in \mathbb{Z}_+ \).

**Task:** Find an orientation \( D \) of \( G \) such that \( \sum_{e \in \delta_D^+(v)} w(e) \leq r \) for all \( v \in V \), where \( \delta_D^+(v) \) stands for the set of edges oriented so that their tail is \( v \).

**Theorem 5** (Theorem 5 from [28]). Minimum Maximum Outdegree is \( \mathcal{W}[1] \)-hard when parameterized by treewidth.

**Theorem 6.** MLQ is \( \mathcal{W}[1] \)-hard when parameterized by treewidth, even when restricted to instances where \( \ell(p) \in \{0, \ell(p)\} \) for every \( p \in P \).

**Proof.** Given an instance \((G = (V, E), w, r)\) of Minimum Maximum Outdegree, we construct an instance \((G' = (A \cup P, E'), \ell, u)\) of MLQ as follows. For every vertex \( v \in V \) we introduce a post \( p_v \in P \) and let \( \ell(p_v) = 0 \) and \( u(p_v) = r \). Furthermore, for every edge \( e = \{v, v'\} \in E \), we introduce two posts \( p_{e,v} \) and \( p_{e,v'} \) with \( \ell(p_{e,v}) = \ell(p_{e,v'}) = u(p_{e,v}) = u(p_{e,v'}) = w(e) + 1 \), and \( 2w(e) + 1 \) applicants \( a_{e,v}^1, \ldots, a_{e,v}^{w(e)}, a_{e,v'}^1, \ldots, a_{e,v'}^{w(e)} \), \( z_e \), for which we introduce the edges \( \{p_{e,v}, a_{e,v}^1\}, \{a_{e,v}^{w(e)}, p_{e,v}\}, \{p_{e,v'}, a_{e,v'}^1\}, \{a_{e,v'}^{w(e)}, p_{e,v'}\} \) for \( i \in \{1, \ldots, w(e)\} \) as well as \( \{p_{e,v}, z_e\} \) and \( \{z_e, p_{e,v'}\} \).

We show that the constructed instance has a solution serving all applicants if and only if the Minimum Maximum Outdegree instance has an orientation respecting the bound on the outdegree.

First assume there is an orientation \( D \) of \( G' \) with maximum outdegree at most \( r \). Then consider the assignment that assigns for every oriented edge \((v, v') \in D\) the \( w(e) \) applicants \( a_{e,v}^i \) to \( p_v \) and the \( w(e) + 1 \) applicants \( a_{e,v'}^i \) and \( z_e \) to \( p_{e,v'} \). As the weighted outdegree of vertex \( v \) is at most \( r \), every post \( p_v \) gets assigned at most \( r = u(p_v) \) applicants.

Now assume \( M \) is a feasible assignment of applicants to posts serving every applicant. In particular, for every edge \( e = \{v, v'\} \in E \), applicant \( z_e \) is assigned to either \( p_{e,v} \) or \( p_{e,v'} \) and exactly one of these two posts is open because the lower bound of \( w(e) + 1 \) can only be met if \( z_e \) is assigned to the respective post. If \( p_{e,v} \) is open then all \( w(e) \) applicants \( a_{e,v}^i \) are assigned to \( p_{e,v} \) and none of the applicants \( a_{e,v}^i \) is assigned to \( p_{e,v'} \), and vice versa if \( p_{e,v'} \) is open. Consider the orientation obtained by orienting every edge \( e \) from \( v \) to \( v' \) if and only if \( p_{e,v} \) is open. By the above observations, the weighted outdegree of vertex \( v \) corresponds to the number of applicants assigned to post \( p_v \), which is at most \( r \).

Finally, note that \( G' \) can be constructed in time polynomial in the input size of the Minimum Maximum Outdegree instance as the weights are encoded in unary there. Furthermore, the treewidth of \( G' \) is at most max\{tw(G), 3\}. To
see this, start with a tree decomposition of $G$ and identify each vertex $v \in V$ with the corresponding post $p_v$. For every edge $e = \{v, v'\} \in E$, there is a bag $B_e$ with $p_{v, e, v'}$ as a child to $B$. We further add the bags $B_{z_{e,v}} = \{p_{v, e, v}, p_{v, e, v'}, z\}$, $B_{a_{i,e,v}} = \{p_{v, e, v}, p_{v, e, v'}, a_i\}$ and $B_{a_{i,e,v}'} = \{p_{v', e, v'}, p_{v', e, v''}, a_{i,e,v}'\}$ for $i \in \{1, \ldots, w(e)\}$ as children to $B_e$. Observe that the tree of bags generated by this construction is a tree decomposition. Furthermore, since we did not increase the size of any of the existing bags and added only bags of size at most 4 the treewidth of $G'$ is at most $\max\{\tw(G), 3\}$.

4 Approximation

Having established the hardness of WMLQ even for very restricted instances in Theorem 1, we turn our attention towards approximability. In this section, we give an approximation algorithm and corresponding inapproximability bounds expressed in terms of $|A|, |P|$ and upper quotas in the graph.

The method, which is formally listed in Algorithm 1, is a simple greedy algorithm. We say a post $p$ is admissible if it is not yet open and $|\Gamma(p)| \geq \ell(p)$. The algorithm iteratively opens an admissible post maximizing the assignable weight, i.e., it finds a post $p' \in P$ and a set $A'$ of applicants in its neighborhood $\Gamma(p')$ with $\ell(p') \leq |A'| \leq u(p')$ such that $\sum_{a \in A'} w(a, p')$ is maximized among all such pairs. It then removes the assigned applicants from the graph (potentially rendering some posts inadmissible) and re-iterates until no admissible post is left.

Algorithm 1 Greedy algorithm for WMLQ

```
Initialize $P_0 = \{p \in P : |\Gamma(p)| \geq \ell(p)\}$.
Initialize $A_0 = A$.
while $P_0 \neq \emptyset$ do
    Find a pair $p' \in P_0$ and $A' \subseteq \Gamma(p')$ with $|A'| \leq u(p')$ such that $\sum_{a \in A'} w(a, p')$ is maximized among all such pairs.
    Open $p'$ and assign all applicants in $A'$ to it.
    Remove $p'$ from $P_0$ and remove the elements of $A'$ from $A_0$.
    for $p \in P_0$ with $\ell(p) > |\Gamma(p) \cap A_0|$ do
        Remove $p$ from $P_0$.
    end for
end while
```

We point out a reduction from WMLQ to the set packing problem here. The elements in the universe of the set packing problem would be $A \cup P$. For each post $p$ and for each subset $S \subseteq \Gamma(p)$, such that $l(p) \leq |S| \leq u(p)$, we create a set $S \cup \{p\}$ for the set packing instance. However, if the difference between upper and lower quota is not bounded, this would create an exponential sized input for the set packing problem and we could only employ an oracle based algorithm known for set packing problem to solve WMLQ. The greedy algorithm
known for set packing problem \cite{5} can be made to work in a fashion similar to the algorithm presented above.

In the following we give a tight analysis of the algorithm, establishing approximation guarantees in terms of the number of posts \(|P|\), number of applicants \(|A|\), and the maximum upper quota \(u_{\text{max}} := \max_{p \in P} u(p)\) over all posts. We also provide two examples that show that our analysis of the greedy algorithm is tight for each of the described approximation factors. We further show there that the approximation ratios given above for WMLQ are almost tight from the point of view of complexity theory.

**Theorem 7.** Algorithm 1 is an \(\alpha\)-approximation algorithm for WMLQ with 
\(\alpha = \min\{|P|, |A|, u_{\text{max}} + 1\}\). Furthermore, for MLQ, Algorithm 1 is a \(\sqrt{|A|} + 1\)-approximation algorithm. It can be implemented to run in time \(O(|E| \log |E|)\).

**Proof.** Let \(p'_1, \ldots, p'_k\) be the posts chosen by the algorithm and let \(A'_1, \ldots, A'_k\) be the corresponding sets of applicants. Furthermore, consider an optimal solution of weight OPT, consisting of open posts \(p_1, \ldots, p_k\) and the corresponding sets of applicants \(A_1, \ldots, A_k\) assigned to those posts.

We first observe that the first two approximation ratios of \(|P|\) and \(|A|\) are already achieved by the initial selection of \(p'_1\) and \(A'_1\) chosen in the first round of the algorithm: For every \(i \in \{1, \ldots, k\}\), post \(p_i\) is an admissible post in the first iteration of the algorithm. The first iteration’s choice of the pair \((p'_1, A'_1)\) implies \(\sum_{a \in A'_1} w(a, p'_1) \geq \sum_{a \in A_1} w(a, p_i) \geq w(a', p_i)\) for every \(a' \in A_1\). As the optimal solution opens at most \(|P|\) posts and serves at most \(|A|\) applicants, we deduce that \(\sum_{a \in A'_1} w(a, p'_1) \geq \min\{|P|, |A|\} \cdot \text{OPT}\).

We now turn our attention to the remaining approximation guarantees, which are \(u_{\text{max}} + 1\) for WMLQ and \(\sqrt{|A|} + 1\) for MLQ. For every \(i \in \{1, \ldots, k\}\), let \(\pi(i)\) denote the first iteration of the algorithm such that \(A'_{\pi(i)} \cap A_i \neq \emptyset\) or \(p'_{\pi(i)} = p_i\). This iteration is the one in which post \(p_i\) is open or an applicant assigned to it in the optimal solution becomes assigned. Note that such an iteration exists, because \(p_i\) is not admissible after the termination of the algorithm. Furthermore, observe that \(\sum_{a \in A'_{\pi(i)}} w(a, p'_{\pi(i)}) \geq \sum_{a \in A_i} w(a, p_i)\), because the pair \((p_i, A_i)\) was a valid choice for the algorithm in iteration \(\pi(i)\). Now for iteration \(j\) define \(P_j := \{i : \pi(i) = j\}\) and observe that \(|P_j| \leq |A'_j| + 1\), because \(P_j\) can only contain one index \(i'\) with \(p'_{i'} = p'_j\), and all other \(i \in P_j \setminus \{i'\}\) must have \(A_i \cap A'_j = \emptyset\) (where the sets \(A_i\) are disjoint). We conclude that

\[
\text{OPT} = \sum_{i=1}^{k} \sum_{a \in A_i} w(a, p_i) \leq \sum_{i=1}^{k} \sum_{a \in A'_{\pi(i)}} w(a, p'_{\pi(i)})
\leq \sum_{j=1}^{\ell} |P_j| \sum_{a \in A'_j} w(a, p'_j) \leq \sum_{j=1}^{\ell} (|A'_j| + 1) \sum_{a \in A'_j} w(a, p'_j).
\]

Note that \(|A'_j| \leq u_{\text{max}}\) and therefore \(\text{OPT} \leq (u_{\text{max}} + 1) \sum_{j=1}^{\ell} \sum_{a \in A'_j} w(a, p'_j)\), proving the third approximation guarantee. Now consider the case that \(w(a, p) =
1 for all \( p \in P \) and \( a \in A \) and define \( A' = \bigcup_{j=1}^{k} A'_j \). If \( |A'| \geq \sqrt{A} \), then \( \sqrt{A}|A'| \geq |A| \geq \text{OPT} \). Therefore assume \( |A'| < \sqrt{A} \). Note that in this case, the above inequalities imply \( \text{OPT} \leq (|A'| + 1)|A'| \leq (\sqrt{A} + 1)|A'| \), proving the improved approximation guarantee for MLQ.

We now turn to proving the bound on the running time. We will describe how to implement the search for the greedy choice of the pair \((p', A')\) in each iteration efficiently using a heap data structure. Initially, for every post \( p \), we sort the applicants in its neighborhood by non-increasing order of \( w \). This takes time at most \( O(|E| \log |E|) \) as the total number of entries to sort is \( \sum_{p \in P} |\Gamma(p)| = |E| \).

We then introduce a heap containing all admissible posts, and associate with each post \( p \) the total weight of the first \( u(p) \) edges in its neighborhood list. Note that these entries can be easily kept up to date by simply replacing applicants assigned to other posts with the first not-yet-assigned entry in the neighborhood list (or removing the post if less than \( \ell(p) \) applicants are available). As every edge in the graph can only trigger one such replacement, only \( O(|E|) \) updates can occur and each of these requires \( O(\log |P|) \) time for reinserting the post at the proper place in the heap. Now, in each iteration of the algorithm, the optimal pair \((p', A')\) can be found by retrieving the maximum element from the heap. This happens at most \( |P| \) times and requires \( O(\log |P|) \) time in each step. \( \square \)

**Example 8** The following two examples show that our analysis of the greedy algorithm is tight for each of the described approximation factors.

(a) Consider an instance of MLQ with \( k + 1 \) posts \( p_0, \ldots, p_k \) and \( k(k+1) \) applicants \( a_{0,1}, \ldots, a_{0,k}, a_{1,1}, \ldots, a_{k,k} \). Let \( \ell(p_i) = u(p_i) = k \) for \( i \in \{0, \ldots, k\} \).

Each applicant \( a_{i,j} \) applies for posts \( i \) and additionally to post 0. For the greedy algorithm, opening post \( p_0 \) and assigning applicants \( a_{1,1}, \ldots, a_{k,k} \) to it is a valid choice in its first iteration, after which no further posts are admissible. Thus, it only assigns \( k \) applicants in total. The optimal solution, however, can assign all \( k(k+1) \) applicants by assigning applicants \( a_{1,1}, \ldots, a_{k,k} \) to \( p_i \) for each \( i \). Therefore, the greedy algorithm cannot achieve an approximation factor better than \( k+1 \) on this family of instances, for which \( |P| = k+1, \sqrt{|A|} < k+1 \), and \( u_{\text{max}} = k \).

(b) To see that the approximation ratio of \(|A|\) is tight for WMLQ consider the following instance with \( k \) posts \( p_1, \ldots, p_k \) and \( k \) applicants \( a_1, \ldots, a_k \). Let \( \ell(p_1) = 0 \) and \( u(p_i) = k \) for every \( i \). Every applicant applies for every post, and \( w(a_i, p_1) = 1 \) for every \( i \) but \( w(a_i, p_j) = \varepsilon \) for every \( j \neq i \) for some arbitrarily small \( \varepsilon > 0 \). In its first iteration, the greedy algorithm might choose to open post \( p_1 \) and assign all applicants to it. This solution accumulates a weight of \( 1 + (k-1)\varepsilon \), while the weight of the optimal solution is \( k = |A| \).

**Theorem 9.** MLQ is not approximable within a factor of \(|P|^{1-\varepsilon} \) or \( \sqrt{|A|}^{1-\varepsilon} \) or \( u_{\text{max}}^{1-\varepsilon} \) for any \( \varepsilon > 0 \), unless \( P = \text{NP} \), even when restricting to instances where \( \ell(p) = u(p) \) for every \( p \in P \) and \( |\Gamma(a)| \leq 2 \) for every \( a \in A \).
Proof. Once again we use the maximum independent vertex set problem. Given an instance of MIS on a graph $G = (V, E)$ with $|V| = n$ and $|E| = m$, we create a MLQ instance with $n$ posts $p_1, \ldots, p_n$, post $p_i$ corresponding to vertex $v_i$. We also introduce $n^2 - m$ applicants as follows. Initially, we introduce $n$ applicants $a_{i,1}, a_{i,2}, \ldots, a_{i,n}$ applying for each post $p_i$. Then, for every edge $\{v_i, v_j\} \in E$, we merge the applicants $a_{i,j}$ and $a_{j,i}$, obtaining a single applicant applying for both $p_i$ and $p_j$. Furthermore, we set $\ell(p_j) = u(p_j) = n$ for every post.

Note that due to the choice of upper and lower bounds, any open post must be assigned to all the applicants in its neighborhood. Thus, a solution to the MLQ instance is feasible if and only if $\Gamma(p_i) \cap \Gamma(p_j) = \emptyset$ for all open posts $p_i$ and $p_j$ with $i \neq j$, which is equivalent to $v_i$ and $v_j$ not being adjacent in $G$ by construction of the instance. Therefore, the MLQ instance has a feasible solution opening $k$ posts (and thus serving $kn$ applicants) if and only if there is an independent set of size $k$ in $G$. We conclude that $\text{OPT}_{\text{MLQ}} = n \cdot \text{OPT}_{\text{MIS}}$ for the two instances under consideration.

Note that in the constructed MLQ instance, $n = |P| = u_{\text{max}} \geq \sqrt{|A|}$. Therefore any approximation algorithm with a factor better than $|P|^{1-\varepsilon}$ or $\sqrt{|A|}^{1-\varepsilon}$ or $u_{\text{max}}^{1-\varepsilon}$ for $\varepsilon > 0$ yields a solution of the instance serving at least $(1/n^{1-\varepsilon})\text{OPT}_{\text{MLQ}}$ applicants, thus opening at least $(1/n^{1-\varepsilon})\text{OPT}_{\text{MLQ}} = (1/n^{1-\varepsilon})\text{OPT}_{\text{MIS}}$ posts, corresponding to an independent set of the same size. By \cite{29}, this implies $P = NP$. \hfill \Box

Acknowledgements

We would like to thank András Frank and Kristóf Bérczi for their observations that led us to Theorem 3 and the anonymous reviewers for their valuable comments, which have helped to improve the presentation of this paper.

References

1. D. J. Abraham, R. W. Irving, and D. F. Manlove. Two algorithms for the Student-Project allocation problem. Journal of Discrete Algorithms, 5(1):79–91, 2007.
2. P. Alimonti and V. Kann. Some APX-completeness results for cubic graphs. Theoretical Computer Science, 237(1-2):123–134, 2000.
3. P. Biró, T. Fleiner, R. W. Irving, and D. F. Manlove. The College Admissions problem with lower and common quotas. Theoretical Computer Science, 411:3136–3153, 2010.
4. H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. SIAM Journal on Computing, 25(6):1305–1317, 1996.
5. B. Chandra and M. M. Halldórsson. Greedy local improvement and weighted set packing approximation. Journal of Algorithms, 39(2):223–240, 2001.
6. H. Y. Cheung, L. C. Lau, and K. M. Leung. Algebraic algorithms for linear matroid parity problems. ACM Transactions on Algorithms (TALG), 10(3):10:1–10:26, 2014.
7. G. Cornuéjols. General factors of graphs. Journal of Combinatorial Theory, Series B, 45(2):185–198, 1988.
8. H. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In Proceedings of STOC ’83: the 15th Annual ACM Symposium on Theory of Computing, pages 448–456. ACM, 1983.
9. H. Gabow. Data structures for weighted matching and nearest common ancestors with linking. In Proceedings of SODA ’90: the 1st ACM-SIAM Symposium on Discrete Algorithms, pages 434–443. ACM-SIAM, 1990.
10. M. Goto, N. Hashimoto, A. Iwasaki, Y. Kawasaki, S. Ueda, Y. Yasuda, and M. Yokoo. Strategy-proof matching with regional minimum quotas. In Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems, AAMAS ’14, pages 1225–1232, Richland, SC, 2014.
11. K. Hamada, K. Iwama, and S. Miyazaki. The Hospitals/Residents problem with lower quotas. Algorithmica, pages 1–26, 2014.
12. H. Hirai and G. Pap. Tree metrics and edge-disjoint $S$-paths. Mathematical Programming, 147(1-2):81–123, 2014.
13. K. Iwama, S. Miyazaki, and H. Yanagisawa. Improved approximation bounds for the student-project allocation problem with preferences over projects. Journal of Discrete Algorithms, 13:59–66, 2012.
14. N. Kamiyama. A note on the serial dictatorship with project closures. Operations Research Letters, 41:559–561, 2013.
15. T. Kloks. Treewidth, Computations and Approximations, volume 842 of Lecture Notes in Computer Science. Springer, 1994.
16. L. Lovász. On the structure of factorizable graphs. Acta Mathematica Academiae Scientiarum Hungaricae, 23(1-2):179–195, 1972.
17. L. Lovász. Antifactors of graphs. Periodica Mathematica Hungarica, 4(2-3):121–123, 1973.
18. L. Lovász. Matroid matching and some applications. Journal of Combinatorial Theory, Series B, 28(2):208–236, 1980.
19. D. F. Manlove. Algorithms of Matching Under Preferences. World Scientific, 2013.
20. D. F. Manlove and G. O’Malley. Student project allocation with preferences over projects. Journal of Discrete Algorithms, 6:553–560, 2008.
21. D. Monte and N. Tumennasan. Matching with quorums. Economics Letters, 120:14–17, 2013.
22. R. Niedermeier. Invitation to Fixed-Parameter Algorithms. Oxford University Press, 2006.
23. C. Papadimitriou and M. Yannakakis. Optimization, approximation and complexity classes. Journal of Computer and System Sciences, 43(3):425–440, 1991.
24. M. Plummer. Graph factors and factorization: 1985–2003: A survey. Discrete Mathematics, 307(7-8):791–821, 2007.
25. M. Samer and S. Szeider. Tractable cases of the extended global cardinality constraint. Constraints, 16:1–24, 2011.
26. A. Sebő. General antifactors of graphs. Journal of Combinatorial Theory, Series B, 58(2):174–184, 1993.
27. A. Sebő and L. Szegő. The path-packing structure of graphs. In D. Bienstock and G. Nemhauser, editors, Integer Programming and Combinatorial Optimization, volume 3064 of Lecture Notes in Computer Science, pages 256–270. Springer Berlin Heidelberg, 2004.
28. S. Szeider. Not so easy problems for tree decomposable graphs. CoRR, abs/1107.1177, 2011.
29. D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. Theory of Computing, 3(6):103–128, 2007.