Assignments for Congestion-Averse Agents: Seeking Competitive and Envy-Free Solutions

Jiehua Chen
Institute of Logic and Computation
TU Wien
Austria
jchen@ac.tuwien.ac.at

Jiong Guo
School of Computer Science and Technology
Shandong University
Qingdao, China
jguo@sdu.edu.cn

Yinghui Wen
Digital and Intelligent Center
Shandong Institute of Information Technology Industry Development
Jinan, China
yinghui.wen@foxmail.com

Abstract

We investigate congested assignment problems where agents have preferences over both resources and their associated congestion levels. These agents are *averse towards congestion*, i.e., consistently preferring lower congestion for identical resources. Such scenarios are ubiquitous across domains including traffic management and school choice, where fair resource allocation is essential. We focus on the concept of *competitiveness*, recently introduced by Bogomolnaia and Moulin [6], and contribute a polynomial-time algorithm that determines competitiveness, resolving their open question. Additionally, we explore two optimization variants of congested assignments by examining the problem of finding envy-free or maximally competitive assignments that guarantee a certain amount of social welfare for every agent, termed *top-guarantees* [6]. While we prove that both problems are NP-hard, we develop parameterized algorithms with respect to the number of agents or resources.

1 Introduction

In the realm of resource allocation and task assignment, the challenge often extends beyond mere allocation–it entails navigating the intricate balance between individual preferences and the congestions that other agents incur. *Congested assignments* epitomize this challenge and address the situation when agents are *congestion-averse*, i.e., the preferences are negatively correlated with the number of agents simultaneously assigned to the same resource, the so-called *congestion level*.

Congested assignments are pertinent in numerous real-world scenarios: From urban *traffic management*, where drivers select routes while considering traffic congestion, to educational contexts like *school choice* [35, 12] or student exercise *slot allocations*, where students' choices are influenced by class sizes. Similarly, in *cloud computing*, allocating computational tasks to servers must account for server load. In each of these scenarios, assigning an agent–whether a driver, a student, or a task executor–to a resource (jointly referred to as a *post*) may introduce additional costs, with detrimental effects on other agents. For example, the productivity in a shared office space diminishes as more people use it. Similarly, in traffic management, drivers might opt for less crowded routes since the mental load increases significantly with even a small increase in traffic. In other words, the agents' individual preference over a *post* is inversely proportional to its congestion level.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

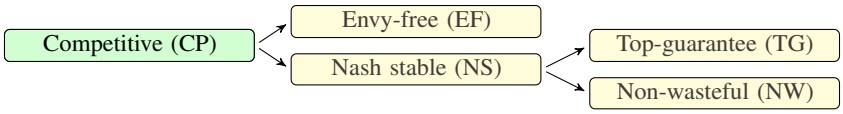


Figure 1: Relations among the different fairness concepts under congestion-averse preferences. All relations are strict; see Lemma 1.

The overarching task is to find an optimal assignment of the agents to posts that respects agents' congestion-averse preferences. Defining 'optimal' in this setting is non-trivial. In traffic management, for example, one might aim for a *Nash stable* assignment (aka. Nash equilibrium [33, 37]), meaning that no agent prefers to deviate to another post which increases the congestion by one. Milchtaich shows that Nash stable assignments for congestion-averse preferences are always attainable and any assignment can be turned Nash stable in polynomially many best-reply improvement steps. Although Nash stability may prevent systematic chaos, it can be quite unfair, particularly in slot allocations, as it may induce envy among agents, meaning that an agent may prefer to take over another agent's post (albeit for the same congestion). Notice that *envy-freeness* is also easy to achieve by simply assigning all agents to the same post, but this approach is often *wasteful* in the sense that there may exist an empty post which an agent prefers to his assignment.

To address the wastefulness in an envy-free assignment, Bogomolnaia and Moulin [6] propose *competitiveness* (CP) as a solution, which ensures that no agent is envious and no post is wasteful. They also introduce *top-guarantees*, a less demanding criterion than competitiveness, which requires every agent to be assigned to one of his top n choices out of all $m \times n$ choices, with m and n being the number of posts and agents, respectively; note that each choice represents a post coupled with a congestion level. Top-guarantees is easy to achieve by a simple sorting algorithm [6, Proposition 1]. Furthermore, top-guarantees is necessary for ensuring Nash stability as well as competitiveness, though the latter criterion – competitiveness – may not always exist.

The following example illustrates the different fairness criteria and Figure 1 depicts their relations.

Example 1. Consider two posts $A = \{a_1, a_2\}$ and three agents $V = \{v_1, v_2, v_3\}$. The preferences of the agents are as follows, with the second component in each tuple denoting the congestion level:

$$\begin{array}{l} v_1\colon\ (a_1,1) \succ (a_1,2) \sim (a_2,1) \succ (a_2,2) \succ \cdots, \\ v_2\colon\ (a_1,1) \sim (a_2,1) \succ (a_1,2) \sim (a_2,2) \succ \cdots, \\ v_3\colon\ (a_1,1) \succ (a_1,2) \succ (a_2,1) \succ (a_2,2) \succ \cdots. \end{array} \qquad \Pi_1\colon \begin{array}{c|c} a_1 & a_2 \\ \hline v_2, v_3 & v_1 \end{array} \qquad \Pi_2\colon \begin{array}{c|c} a_1 & a_2 \\ \hline v_1, v_3 & v_2 \end{array}$$

Roughly speaking, without congestion, both v_1 and v_3 strictly prefer a_1 to a_2. However, while v_1 is indifferent between sharing a_1 with another agent and being alone at a_2 (indicated by $(a_1, 2) \sim (a_2, 1)$), v_3 will only prefer to go to a_2 when a_1 has more than two congestions. Agent v_2 is indifferent between a_1 and a_2 in general and cares only about congestion.

There are several Nash stable assignments, e.g., Π_1: assigning v_2 and v_3 to a_1, and v_1 to a_2: v_1 will not deviate to a_1 since otherwise the congestion of a_1 will be increased to three and v_1 prefers $(a_2, 1)$ to $(a_1, 3)$ (due to the congestion aversion assumption), and similar reasoning applies to v_2 and v_3. However, it is not envy-free (and hence not competitive) since agent v_2 envies v_1. On the other hand, assigning agent v_1 and v_3 to post a_1, and agent v_2 to post a_2 (see Π_2) is competitive, and hence envy-free and top-guaranteed.

If both v_1 and v_3 have the same preferences as v_2, however, then no competitive assignments exist since at least one post a is assigned no more than one agent, so either it is wasteful or another agent will envy the agent assigned to a. In this case, Π_1 and Π_2 are still Nash stable (and hence top-guaranteed) but not envy-free or competitive anymore. □

Competitiveness offers a promising resolution for congested assignments, as it ensures non-wastefulness, envy-freeness, and top-guarantees, thereby securing a specific welfare level for all participants. Despite their appeal, the computational complexity of determining competitive assignments remains open, as noted by Bogomolnaia and Moulin [6]. This unresolved issue leads us to our core research question:

Q1: Is there an efficient way to determine whether a competitive assignment exists?

Since CP may not always exist, we consider two relaxations:

Q2: How hard is it to find a top-guaranteed and envy-free assignment?
Q3: How about top-guaranteed and maximally *competitive assignments?*

We tackle these questions by exploring the computational intricacies involved in achieving competitiveness, top-guarantees, and envy-freeness in congested assignments.

Our main contributions. We resolve Q1 affirmatively by showing that determining the existence of a competitive assignment is polynomial-time solvable. We also establish the NP-hardness of finding top-guaranteed and envy-free assignments (Q2), and of top-guaranteed and maximally competitive assignments (Q3).

To answer Q1, we first show two key insights: (1) We can restrict our search to CP assignments where every post is non-empty; see Lemma 3. (2) We can use maximum flow to find the congestion vector of a CP assignment with all posts being non-empty and derive the corresponding CP assignment (if it exists) in polynomial time. For Q2 and Q3, we provide hardness reduction from the NP-complete problems EXACT COVER BY 3-SETS and CLIQUE, respectively. We complement the hardness results by providing several parameterized algorithms.

Related work. The congested assignment problem is rooted in *congestion games*, introduced by Rosenthal [36]. In these games, agents select subsets of *resources* (posts), with each resource having a *cost function* dependent on its usage level. An agent's total cost is the sum of costs over their selected resources. Rosenthal proved the existence of *Nash equilibria*, while Milchtaich [33] extended the model with player-specific cost functions and demonstrated the same existence property. The field has evolved through both theoretical and computational perspectives [34, 28, 8, 10, 26, 2, 32, 11].

Recently, Bogomolnaia and Moulin [6] introduced two significant fairness concepts for congested assignments: *top-guarantees* and *competitiveness*. The former ensures a minimum welfare guarantee for each agent, while the latter addresses a more complex balance of envy-freeness and non-wastefulness. top-guarantees differs from the classical *minimax* principle in game theory: while minimax aims to minimize the maximum cost of any participant, top-guarantees establishes a hard constraint ensuring every agent achieves a certain welfare threshold. This connects our work to other assignment problems utilizing the minimax principle, such as MIN-REGRET STABLE MATCHING [24, 31], which finds stable matchings that minimize the worst rank of any assigned partner. Bogomolnaia and Moulin demonstrated that top-guarantees is straightforward to achieve, but left the complexity of finding competitive assignments as an open question. Our primary contribution is answering this question by providing a novel polynomial-time algorithm for determining competitiveness.

Congested assignment can be viewed as a restricted variant of the GROUP ACTIVITY SELECTION (GAS) problem, introduced by Darmann et al. [18, 19] and subsequently extended and studied by many others [30, 14–17, 27, 23, 21]. GAS extends the congested assignment concept by allowing agents' preferences over pairs of activities and group sizes, not necessarily exhibiting aversion to larger groups. Despite its broader scope, the specific challenges of applying competitiveness or top-guarantees within GAS have not been previously addressed.

Congestion-averse preferences in congested assignment are similar to the preferences of the agents in *anonymous hedonic games* [4], where agents form coalitions based solely on group size preferences rather than specific membership.

Congested assignment is also related to Copland's SCHOOL CHOICE WITH CLASS SIZE EXTERNALITIES (SC-CSE) problem [12], which generalizes the classical SCHOOL CHOICE problem [1]. Compared to our setting, every post in SC-CSE also has a capacity bound and a preference list, which ranks all agents in strict order and the goal is to find a stable assignment, i.e., an assignment without *justified envies* and *wastefulness*. Phan et al. [35] investigate a model similar to Copland. Instead of post-congestion pairs, they assume that the preferences are over pairs of posts and "resource ratio," and investigate an equilibrium notion that is specific to their model. When imposing lower and upper quotas on the capacities of the posts, our problem is related to STABLE MATCHINGS WITH UPPER AND LOWER QUOTAS (SM-ULQ) [5, 25, 3, 9]. Since SM-ULQ is NP-hard, one could obtain the same hardness for finding a competitive assignment with lower and upper quotas.

Finally, congested assignment may be related to the SCHEDULING TO MAXIMIZE PARTICIPATION problem [7], where servers correspond to posts and clients agents such that each client has capacity

bound for each server and will not be satisfied if that bound is exceeded. The goal is to assign clients to servers maximizing the number of "satisfied" clients.

For a comprehensive discussion on related works in congested assignments, we refer readers to the recent paper by Bogomolnaia and Moulin [6].

Structure of the paper. In Section 2, we introduce necessary definitions and concepts for the paper, and describe an approach to determining whether a competitive assignment for a given congestion vector exists. In Section 3, we present our main result, an efficient algorithm for competitive assignments. In Section 4, we show NP-hardness for top-guaranteed assignments that are envy-free or maximally competitive, respectively, and provide some parameterized algorithms for these two problems. We conclude in Section 5 for future research. Due to space constraints, the proofs of statements marked by ($\star$) are deferred to the appendix.

2 Preliminaries

Given a non-negative integer z, let $[z]$ denote the set $\{1, \dots, z\}$. We assume basic knowledge of parameterized complexity and refer to the textbook by Cygan et al. [13] for more details.

Let $A = \{a_1, \dots, a_m\}$ denote a finite set of m *posts* and $V = \{v_1, \dots, v_n\}$ a finite set of n *agents*.[1] The input of CONGESTED ASSIGNMENT consists of A, V, and for each agent $v \in V$, a *preference list* $\succeq_v$ (i.e., a weak order, which is transitive and complete) on the set of *tuples* $A \times [n]$ (i.e., ordered pairs). The weak order $\succeq_v$ specifies the preferences of agent v over the posts and their congestions (i.e., the number of agents that will simultaneously occupy the post). We use $\sim_v$ to denote the symmetric part of $\succeq_v$ and $\succ_v$ the asymmetric part; we neglect the subscript $_v$ if it is clear from the context which agent we refer to. The agents may be indifferent between different posts, but are *averse* to congestion, i.e., for each post $a_j \in A$ and each congestion level $d \in [n-1]$, each agent v has $(a_j, d) \succ_v (a_j, d+1)$. For instance, in Example 1, agent v_1: $(a_1, 2) \sim (a_2, 1)$ means that v_1 is indifferent between $(a_1, 2)$ and $(a_2, 1)$. This also implies that he has $(a_1, 1) \succ (a_2, 2)$ (because of aversion of congestion), meaning that he *strictly* prefers being assigned to post a_1 alone over to post a_2 with two agents.

An *assignment* of agents V to posts A is a partition $\Pi = (S_a)_{a \in A}$ of V where S_a is the set of agents assigned to post a so that every two sets S_a and S_b are mutually disjoint and $\bigcup_{a \in A} S_a = V$. The cardinality $|S_a|$ of S_a is called the *congestion* of post a. We say that a post a is *empty* if $S_a = \emptyset$. For brevity's sake, we use $\Pi(v)$ to refer to the post that agent v is assigned to and often use $\Pi(a)$ to refer to the set S_a of agents that are assigned to post a. We call $\vec{s} = (|\Pi(a)|)_{a \in A}$ the *congestion vector* of partition Π.

Definition 1 (Nash stable, envy-free, top-guaranteed, and competitive assignments)**.** Let $\Pi = (S_a)_{a \in A}$ denote an assignment for an instance $(A, V, (\succeq_v)_{v \in V})$ of CONGESTED ASSIGNMENT.

We say that Π is *Nash stable* (in short, *NS*) if no agent wishes to deviate to another post. Formally, Π is *NS* if for every agent $v \in V$ and every post $a \in A$ it holds that $(a^*, |S_{a^*}|) \succeq_v (a, |S_a| + 1)$, where a^* denotes the post that agent v is assigned to.

We say that agent v *envies* agent v' if $(b, |S_b|) \succ_v (a, |S_a|)$ where v is assigned to a and v' to b. Accordingly, we say that Π is *envy-free* (in short, *EF*) if no agent envies any other agent.

We say that Π is *wasteful* if there exists an agent v and an empty post a such that v strictly prefers $(a, 1) \succ_v (a^*, |S_{a^*}|)$, where a^* denotes the post that agent v is assigned to; otherwise Π is *non-wasteful* (in short, *NW*).

We say that Π is *top-guaranteed* (in short, *TG*) if every agent $v \in V$ is assigned to a post $a = \Pi(v)$ such that $(a, |\Pi(a)|)$ is among the $|V|$ tuples in the preference list $\succeq_v$, breaking ties arbitrarily.

We say that an agent $v \in V$ is *satisfied* with Π if he neither envies any other agent nor prefers to move to an empty post, i.e., for every post $a \in A$ it holds that $(a^*, |S_{a^*}|) \succeq_v (a, \max(|S_a|, 1))$, where a^* denotes the post that agent v is assigned to. Otherwise, we say that v is *unsatisfied* with Π. Accordingly, Π is *competitive* (in short, *CP*) if every agent is satisfied with Π. Π is *maximally competitive* if it admits the fewest number of unsatisfied agents among all assignments. □

[1] A and V are standard notion for the set of alternatives and voters, respectively, from voting theory. We adopt them since the agents also have preferences.

ALGORITHM 1: Determining the existence of CP assignments

Input: An instance $I' = (A, V, (\succeq_v)_{v \in V})$.
Output: A CP assignment if it exists; otherwise no.

foreach $k = \max(0, m - n)$ *to* $m - 1$ **do**
  Compute $I = (A, V, (\succeq_v)_{v \in V})$ according to Construction 1 on input (I', k)
  ▷ Decide whether there exists a CP assignment for I with all posts being non-empty.
  $T[a] \leftarrow 1$ for all $a \in A$
  while $\sum_{a \in A} T[a] \leq |V|$ **do**
    ▷ Phase 1: Deciding existence of a perfect flow
    Let $(G = (\hat{A} \cup \hat{V} \cup \{s, t\}, E), \mathsf{c})$ be the network constructed by Construction 2 on input (I, T)
    Compute a max flow f of (G, c)
    if *f has value* $|V|$ **then return** the assignment derived from f as per Definition 2 without the k dummy agents. ;
    ▷ Phase 2: find an obstruction
    Find a vertex $\hat{v}^* \in \hat{V}$ with $f(\hat{v}^*, t) = 0$
    $V' \leftarrow \{\hat{v}^*\}$; $A' \leftarrow \emptyset$
    repeat
      $\hat{a} \leftarrow$ a vertex in $\hat{A} \setminus A'$ with $(\hat{a}, \hat{v}) \in E$ for some $\hat{v} \in V'$
      $A' \leftarrow A' \cup \{\hat{a}\}$
      $V' \leftarrow V' \cup \{\hat{v} \in \hat{V} \mid f(\hat{a}, \hat{v}) = 1\}$
    until *no* $\hat{a} \in \hat{A} \setminus A'$ *exists with* $(\hat{a}, \hat{v}) \in E$ *for some* $\hat{v} \in V'$;
    ▷ Phase 3: update T
    foreach $\hat{a} \in A'$ **do** $T[a] \leftarrow T[a] + 1$;
  return no
return no

For congestion-averse preferences, we observe the following relations among the five fairness concepts: CP, EF, NS, TG, NW; see Figure 1. Note that the relation of "CP implies NS" and "NS implies TG" have already been shown by Bogomolnaia and Moulin [6]. We provide proofs in the appendix for the sake of completeness.

Lemma 1 ($\star$). *(1) CP implies EF, but the converse does not hold.*
(2) CP implies NS, but the converse does not hold.
(3) NS implies TG and NW, but the converse does not hold.
(4) EF is incomparable to NS, to TG, and to NW, respectively; TG is incomparable to NW.

By Lemma 1(2)–(3), when searching for CP assignments, we only need to consider the first $|V|$ tuples of each preferences list. For each agent v and post a, we use $\lambda(v, a)$ to refer to the *maximum congestion* of v for post a in his preference list $\succeq_v$ such that $(a, \lambda(v, a))$ is among the first $|V|$ tuples, with ties broken arbitrarily; if no tuple containing a is among the first $|V|$ tuples, then $\lambda(v, a) = 0$. Further, we say that post a is *acceptable* to agent v if $\lambda(v, a) > 0$, in other words, a, together with some congestion d, is contained in one of the first $|V|$ tuples in $\succeq_v$.

The maximum congestions in Example 1 are: $\lambda(v_i, a_1) = 2$ and $\lambda(v_i, a_2) = 1$, $i \in [3]$.

Milchtaich [33] shows that NS assignments always exist. Indeed, he proves that there always exists a best-reply strategy path connecting an arbitrary assignment to an NS assignment, and such path can be found in polynomial time. CP assignments, however, do not always exist (see Example 1), and it is not clear how to adapt the NS-improvement approach to determine whether CP exists. However, if the congestion levels are given, we can determine in polynomial time whether there exists a CP assignment that match these levels. In the following, we provide an efficient approach to finding a maximally competitive assignment when a congestion vector is given. The idea is to guess (by brute-forcing) the number of unsatisfied agents and determine a perfect $\vec{b}$-matching in an appropriate bipartite graph between the agents and the posts.

Lemma 2 ($\star$). *Given a congestion vector $\vec{s}$ with $\sum_{a \in A} \vec{s}[a] = |V|$, in polynomial-time one can determine the smallest number t of unsatisfied agents among all assignments whose congestion vectors equal $\vec{s}$; the corresponding assignment can found in polynomial time.*

3 Algorithms for CP Assignments

In this section, we present a comprehensive analysis of our approach to CP assignments. We provide an overview of the procedure (Algorithm 1) before exploring its theoretical foundations in detail. The algorithm consists of two main nested loops. The outer loop enumerates possible values of k, representing the number of empty posts in a potential CP assignment. For each k, we construct an extended instance I that models the original problem with exactly k empty posts; see line 1. Within the inner **while**-loop (lines 4–16), we determine whether I admits a CP assignment with all posts being non-empty: Starting with a congestion vector T where $T[j] = 1$ for all $j \in [m]$, we iteratively refine this vector by building a flow network corresponding to T, determining the maximum flow, and either deriving a CP assignment when the maximum flow value is n, or increasing some entries in T, or terminating when $\sum_{j\in[m]} T[j] > n$, inferring no CP assignment exists where all posts are non-empty.

In the following, we provide rigorous justification for our approach. Section 3.1 establishes why we can restrict our search to assignments where every post is non-empty, while Section 3.2 demonstrates the correctness of our maximum flow formulation for finding CP assignments with all posts being non-empty, as well as a correctness proof of Algorithm 1.

3.1 Reducing to Determining CP Assignments with No Empty Posts

In this subsection, we show how to reduce our problem to the restricted problem of deciding a CP assignment with all posts being non-empty, ensuring line 2 in Algorithm 1 is correct. The basic idea is to guess the number k of empty posts (assuming CP exists) and augment the instance with k dummy agents, two auxiliary agents and two fallback posts such that any CP assignment must assign to each previously empty post a distinct dummy agent. The core of the reduction is described in Construction 1 below; see Appendix B.1 for an example.

Construction 1 (Extended instance)**.** *Given an instance $I = (A, V, (\succeq_v)_{v\in V})$ of* CONGESTED ASSIGNMENT *with m posts and n agents, and a number k with $\max(0, m-n) \leq k \leq m-1$ construct a new instance $(A^*, V^*, (\succeq^*_v)_{v\in V^*})$ as follows. Create k dummy agents $u_1, \ldots, u_k$, two auxiliary agents p_1, p_2, and 2 dummy posts b_1, b_2. Set $V^* = V \cup \{u_1, \ldots, u_k, p_1, p_2\}$ and $A^* = A \cup \{b_1, b_2\}$. Finally, define the preferences of the agents as follows[2]. Here, v denotes an original agent from I and $\succeq_v$ his original preference list, but restricted to the first n tuples, $z \in [k]$, $N \coloneqq k + n + 2$.*

u_z: $(a_1, 1) \sim (a_2, 1) \sim \cdots \sim (a_m, 1) \succ (b_1, 1)$
$\quad \succ (b_1, 2) \succ \cdots \succ (b_1, N - m)$;
p_1: $(b_1, 1) \succ (b_2, 1) \succ (b_2, 2) \succ \cdots \succ (b_2, N - 1)$;
p_2: $(b_2, 1) \succ (b_1, 1) \succ (b_1, 2) \succ \cdots \succ (b_1, N - 1)$;
v: $(\succeq_v) \succ (b_1, 1) \succ (b_1, 2) \succ \cdots \succ (b_1, k + 2)$.

	a_1	$\cdots$	a_m	b_1	b_2
u_z	1	$\cdots$	1	$N - m$	0
p_1	0	$\cdots$	0	1	N
p_2	0	$\cdots$	0	N	1
v	$\lambda(v, a_1)$	$\cdots$	$\lambda(v, a_m)$	$k + 2$	0

The table above states the maximum congestion of each agent towards each post, where a_j, $j \in [m]$, and v denote the original post and agent, respectively.

To prove the correctness, we first make an observation about the extended instance.

Observation 1. *Let $I_k = (A^*, V^*, (\succeq^*_v)_{v\in V^*})$ denote the instance created by Construction 1 with $A^* = A \cup \{b_1, b_2\}$ and $V^* = V \cup \{u_i \mid i \in [k]\} \cup \{p_1, p_2\}$. Every CP assignment of I_k (if it exists) satisfies the following: (1) p_1 is assigned to b_1 alone, and p_2 to b_2 alone. (2) Every dummy u_z with $1 \leq z \leq k$ is assigned to some $a_j \in A$ alone. (3) Every original $v_i \in V$ is assigned to some original post.*

Proof. Let Π be a CP assignment of I_k with $\Pi = (S_a)_{a\in A^*}$. To show the first part of statement (1), we observe that if p_1 is assigned to b_1, then $|S_{b_1}| = 1$, due to the maximum congestion of p_1 towards b_1. Thus, it suffices to show that p_1 is indeed assigned to b_1. Suppose, for the sake of contradiction, that p_1 was assigned to b_2 instead; note that he will not be assigned to any original post a_j due to his maximum congestion towards a_j. Then, by the maximum congestion of p_2 towards b_2, agent p_2 could not be assigned to b_2. No original agent could be assigned to b_2 either, due to his maximum congestion towards b_2. Hence, we would have $S_{b_2} = \{p_1\}$ and p_2 would envy p_1, a contradiction to the competitiveness.

[2]By Lemma 1(2)–(3), we only need to consider the first $n + k + 2$ tuples in each preference list.

The second part of statement (1) follows directly from the first part since b_2 is the only acceptable post left for p_2 and his maximum congestion towards b_2 is one.

By statement (1) and the maximum congestions, every dummy agent u_z can only be assigned to some original post alone, proving statement (2). The last statement follows directly from the first two statements. □

Now, we show the correctness of the construction.

Lemma 3. *An instance I admits a CP assignment if and only if there exists an integer k with $\max(0, m-n) \leq k \leq m-1$ such that the instance I_k created by Construction 1 admits a CP assignment with all posts being non-empty.*

Proof. Let $I = (A, V, (\succeq_v)_{v \in V})$. The "only if" part is straightforward: Let $\Pi = (S_a)_{a \in A}$ denote a CP assignment of I, and let A' denote the set of empty posts under Π with $k = |A'|$. Clearly, $(0, m-n) \leq k \leq m-1$. Consider the instance I_k created according to Construction 1 on (I, k). We claim that the following assignment Π_k for I_k is CP where every post is non-empty.

- For each $a \in A'$, take a unique dummy agent u_z and assign $\Pi_k(a) = \{u_z\}$; note that there are exactly $k = |A'|$ many dummy agents.
- For each non-empty post $a \in A \setminus A'$, let $\Pi_k(a) = \Pi(a)$.
- Let $\Pi_k(b_1) = \{p_1\}$ and $\Pi_k(b_2) = \{p_2\}$.

We continue to show why the derived assignment Π_k is CP for I_k. Since no post is empty, showing competitiveness reduces to showing that no agent is envious. This is clearly the case for all dummy agents including p_1 and p_2 since they are assigned to one of their most preferred posts alone. No original agent envies any other original agent or any dummy agent since Π is CP for I. No original agent $v \in V$ envies p_1 or p_2 since p_1 and p_2 are assigned to b_1 and b_2, Π is TG for I, and b_1 and b_2 occurs at the end of $\succeq_v^*$. This shows that Π_k is CP for I_k, as desired.

For the "if" part, let k be an integer between $\max(0, m-n)$ and $m-1$ such that the created instance I_k admits a CP assignment Π_k without empty post. We claim that the assignment Π derived from Π_k by omitting all dummy agents and the posts p_1 and p_2 is CP for I.

We first show that Π is a valid assignment for I. By Observation 1(2), every dummy agent is assigned to some original post *alone*, and hence for each $a_j \in A$ that is not assigned any dummy agent (i.e., $\{u_1, \ldots, u_k\} \cap \Pi_k(a_j) = \emptyset$), we have $\Pi(a_j) = \Pi_k(a_j)$; and $\Pi(a_j) = \emptyset$, otherwise. This implies that $\Pi(a_j) \subseteq V$.

By Observation 1(3), every original agent is assigned to an original post, confirming that Π is indeed a valid assignment for I.

Next, suppose, for the sake of contradiction, that Π is not competitive and let v and a be an agent and a post, respectively, such that $(a, \max(|\Pi(a)|, 1)) \succ_v (a', |\Pi(a')|)$ where a' is the post that v is assigned to by Π. We infer that $\Pi(a)$ cannot be empty since otherwise by Construction 1 and by Observation 1(2) we would have that $\Pi_k(a) = \{u_z\}$ for some dummy agent u_z. This further implies that v envies u_z in I_k, a contradiction to the competitiveness of Π_k. Since $\Pi(a)$ is not empty and $v \in \Pi(a')$, again by Construction 1 and by Observation 1(2), we have that $\Pi(a) = \Pi_k(a)$ and $\Pi(a') = \Pi_k(a')$. Since $\succeq_v^*$ is an extension of $\succeq_v$ for each $v \in A$, we obtain that $(a, |\Pi_k(a)|) \succ_v^* (a', |\Pi_k(a')|)$ a contradiction to the competitiveness of Π_k. □

3.2 Determining CP Assignments with No Empty Posts

In this subsection, we show how the **while**-loop in lines 4–16 works. As already discussed in the beginning of Section 3, by Lemma 2, we need to determine the desired congestion vector. To achieve this, we iteratively update an integer table T, which stores the congestion level for each post that a CP assignment should not fall below. Each iteration has three phases. In Phase 1 (lines 5–7), we construct a flow network with capacities corresponding to T and determine whether there exists a *perfect* flow, i.e., the value of the flow is equal to the number of agents. If this is the case, we derive and return the corresponding CP assignment (line 7). Otherwise, we proceed with Phase 2 (lines 8–14), where we find an obstruction (see Definition 3) containing posts whose congestion levels need to be incremented necessarily. In Phase 3 (line 15), we update the table entries of the posts from the

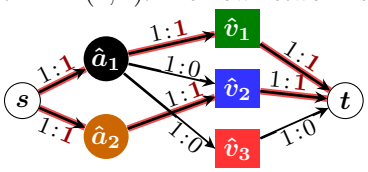
obstruction. In the remainder of the subsection, we address these phases in details. We first introduce necessary concepts, starting flow networks; see Example 2 for an illustration.

Construction 2 (Flow network)**.** *Given an instance $I = (A, V, (\succeq_v)_{v\in V})$ together with a congestion table T which has an entry $1 \le T[a] \le |V|$ for each post $a \in A$, we construct a network $N = (G, c)$, where $G = (\hat{A} \cup \hat{V} \cup \{s, t\}, E)$ is a directed graph with dedicated source s and target t, and $\mathsf{c}\colon E(G) \to [|V|]$ is a capacity function:*

(i) For each $a \in A$, create a vertex $\hat{a}$. Let $\hat{A} = \{\hat{a} \mid a \in A\}$.
(ii) For each $v \in V$, create a vertex $\hat{v}$. Let $\hat{V} = \{\hat{v} \mid v \in V\}$.
(iii) For each post $a \in A$, create an arc $(s, \hat{a})$ from the source s and set the capacity $\mathsf{c}(s, \hat{a}) = T[a]$.
(iv) For each agent $v \in V$, create an arc $(\hat{v}, t)$ to the target t and set the capacity $\mathsf{c}(\hat{v}, t) = 1$.
(v) For each agent $v \in V$ and each post a, if v considers $(a, T[a])$ as the most preferred tuple among all tuples $(a', T[a'])$, $a' \in A$, then create an arc $(\hat{a}, \hat{v})$ with capacity $\mathsf{c}(\hat{a}, \hat{v}) = 1$.

The capacity function c is $\mathsf{c}(e) = T[a]$ if $e = (s, \hat{a})$ with $a \in A$; otherwise $\mathsf{c}(e) = 1$.

Example 2. Consider the first instance in Example 1 and start with $T = (1, 1)$. The flow network is given on the right, where a label with "$x : y$" means the corresponding arc has capacity x and a maximum flow has value y on that arc. For this network, the maximum flow value is 2. It is not a perfect flow since agent v_3 is not assigned. This implies that no assignment with congestion vector $T = (1, 1)$ is CP.



We need the following concepts to derive an assignment from a flow.

Definition 2 (Perfect flows and the derived assignment)**.** Let (G, c) denote the network created by Construction 2 for an instance $I = (A, V, (\succeq_v)_{v\in V})$, together with a table $T \in [|V|]^{|A|}$. A *flow* of (G, c) is a function $f\colon E(G) \to [|V|] \cup \{0\}$[3] that assigns to each arc a value, satisfying the following: $f(e) \le \mathsf{c}(e)$ for all $e \in E(G)$, and $\sum_{(x,y)\in E(G)} f(x, y) = \sum_{(z,x)\in E(G)} f(z, x)$ for all $x \in \hat{A} \cup \hat{V}$. The *value* of a flow equals the net flow into the sink t: $v(f) = \sum_{(x,t)\in E(G)} f(x, t)$. A flow is called *perfect* if the value of f is $|V|$; this means that with a perfect flow, every arc $(\hat{v}, t)$ to the sink is saturated.

Given a perfect flow f, we derive a *congested assignment* Π for the original instance I by setting $\Pi(a) = \{v \mid f(\hat{a}, \hat{v}) = 1\}$ for each post $a \in A$. □

We also need the concept of obstructions, which are witnesses for the absence of a perfect flow. This is similar to the forbidden substructure of a perfect matching in Hall's marriage theorem.

Definition 3 (Obstruction)**.** Let (G, c) be a network with $G = (\hat{A} \cup \hat{V} \cup \{s, t\}, E)$. A pair (A', V') with $A' \subseteq \hat{A}$ and $V' \subseteq \hat{V}$ is called an *obstruction* for (G, c) if the following holds:

(i) $\emptyset \neq V' \subseteq \hat{V}$;
(ii) $A' = \{\hat{a} \in \hat{A} \mid \exists \hat{v} \in V'$ with $(\hat{a}, \hat{v}) \in E(G)\}$, i.e., A' is equal to the in-neighborhood of V';
(iii) For each $\hat{a} \in A'$, $\mathsf{c}(s, \hat{a}) < |\{\hat{v} \in V' \mid (\hat{a}, \hat{v}) \in E\}|$.
(iv) $\sum_{\hat{a}\in A'} \mathsf{c}(s, \hat{a}) < |V'|$.

A' can be seen as a *minimal* set of posts a with congestions that are not enough to accommodate all agents from V' according the congestion table T. □

Throughout the remainder of the section, by an *iteration*, we mean the execution of lines 5–7 if a CP assignment is found, and lines 5–15 otherwise. For each iteration $z \ge 1$, we use T_z to denote the table at the beginning of iteration z (i.e., at line 5).

The correctness of the three phases is based on Lemmas 4 to 6 which we present next. Lemma 4 guarantees that the second phase always finds some critical posts to increment their congestions.

Lemma 4 ($\star$)**.** *Each (A', V') computed in Phase 2 in lines 8–14 is an obstruction.*

Lemma 5 ensures that increasing the table entries is safe and that the no-answer in line 16 is correct.

[3]Note that since the capacity values are integral, we can assume without loss of generality that the flow is a also integral.

Lemma 5 ($\star$). *Assume that I admits a CP assignment Π with no posts being empty. Then, for each iteration $z \geq 1$, each obstruction (A', V') found in iteration z, and each post $a \in A$, the following holds. If $\hat{a} \in A'$, then $|\Pi(a)| \geq T_z[a] + 1$; otherwise $|\Pi(a)| \geq T_z[a]$.*

Lemma 6 ensures that the returned assignment is CP.

Lemma 6 ($\star$). *If Π is an assignment returned in line 7, then Π is CP and has no empty post.*

We have everything ready to show the correctness.

Theorem 1 ($\star$). *Algorithm 1 correctly decides whether an instance has a CP assignment in $O(m^2 \cdot (n+m)^2)$ time, where m and n denote the number of posts and agents, respectively.*

Proof. Let I' be an instance. By Lemma 3, we only need to show that I' is a yes instance if and only if there exists a $k \in \{\max(0, m-n), \ldots, m-1\}$ such that line 7 returns an assignment where every post is non-empty and is CP for the instance I constructed in line 2.

This reduces to showing that lines 3–16 correctly decide whether I admits a CP assignment with all posts being non-empty. Clearly, if line 7 returns an assignment Π, then by Lemma 6, Π is CP and every post is assigned at least one agent. Hence, to show the correctness, we need to show that whenever line 16 returns no, I does not admit a CP assignment where every post is non-empty. Towards a contradiction, suppose that I admits a CP assignment, say Π, where every post is non-empty. Since line 16 returns no, in the second last iteration z, we have $\sum_{a \in A} T_z[a] \leq |V|$, but after the update of some table entries the sum exceeds $|V|$. Let T_{z+1} denote the table entries at the end of iteration z. By assumption, $\sum_{a \in A} T_{z+1}[a] > |V|$ and $\sum_{a \in A} T_z[a] \leq |V|$.

Since we updated some table entries, we must have found an obstruction (A', V') in iteration z according to which we made the update. By Lemma 5, we infer that for all posts $a \in A$, it holds that $|\Pi(a)| \geq T_z[a] + 1 = T_{z+1}[a]$ if $\hat{a} \in A'$, and $|\Pi(a)| \geq T_z[a] = T_{z+1}[a]$ if $\hat{a} \notin A'$. This implies that $|\Pi(a)| \geq T_{z+1}[a]$ holds for all $a \in A$. Then, $\sum_{a \in A} |\Pi(a)| \geq \sum_{a \in A} T_{z+1}[a] > |V|$, a contradiction to Π being a valid assignment. The running time analysis is deferred to Appendix B.6 □

4 Two Optimization Variants

In this section, we continue with Q2 and Q3 from the introduction. Specifically, we investigate the computational complexity of finding TG assignments that are additionally either EF or maximally CP.

EF and TG assignments. We first focus on EF and TG assignments, and show that it is NP-hard to find such assignments. Let EF+TG refer to the problem of determining whether a given instance has an EF and TG assignment. We reduce from the NP-complete EXACT COVER BY 3-SETS (X3C) problem. The input of X3C is a pair $(U, \mathcal{S})$, where $U = \{u_1, \ldots, u_{3n}\}$ is a finite set of $3n$ elements, and $\mathcal{S}$ is a family of subsets $\mathcal{S} = \{C_1, \ldots, C_m\}$ with $C_j \subseteq U$ and $|C_j| = 3$ for each $j \in [m]$. The question is whether there exist an *exact cover* $J \subseteq [m]$ for U, i.e., $|J| = n$ and $\bigcup_{j \in J} C_j = U$. Note that X3C remains NP-hard even if each element appears in exactly three subsets [22], meaning that $m = 3n$.

Theorem 2 ($\star$). EF+TG *is NP-complete; hardness holds even if there are no ties.*

Proof. NP-containment is clear since one can check in polynomial time whether a given assignment is EF and TG. Now, we focus on NP-hardness and reduce from X3C. Let $I = (U, \mathcal{S})$ denote an instance of X3C with $U = \{u_1, \ldots, u_{3n}\}$and $\mathcal{S} = (C_j)_{j \in [m]}$ such that every element in U appears in exactly three members of $\mathcal{S}$; note that $m = 3n \geq 3$.

We create an instance I' of CONGESTED ASSIGNMENT as follows. For each member $C_j \in \mathcal{S}$, create a *set-post* a_j; For each element $u_i \in U$, create an *element-agent* v_i; Create two dummy posts b_1 and b_2 and $4m$ dummy agents $p_1, p_2, \ldots, p_{2m}$, $q_1, q_2, \ldots, q_{2m}$. Let $A = \{a_j \mid j \in [m]\} \cup \{b_1, b_2\}$ and $V = \{v_i \mid i \in [3n]\} \cup \{p_j, q_j \mid j \in [2m]\}$.

We describe the preferences of the agents, where the last "$\cdots$" denote an arbitrary but fixed order of the rest of the tuples. $\langle \alpha, s, t \rangle = (\alpha, s) \succ (\alpha, s+1) \succ \cdots \succ (\alpha, t)$ depict the preference list on tuples for post α and congestions ranging between s and t with $s \leq t$.

– The preferences of agent v_i is defined as follows, where C_j, C_k, C_t denote the three members in $\mathcal{S}$ that contain u_i with $j < k < t$: v_i: $(a_j, 1) \succ (a_k, 1) \succ (a_t, 1) \succ (a_j, 2) \succ (a_k, 2) \succ (a_t, 2) \succ (a_j, 3) \succ (a_k, 3) \succ (a_t, 3) \succ \langle b_2, 1, 3n + 4m - 9\rangle \succ \cdots$.
In other words, each agent v_i considers the three posts which correspond to the sets that contain u_i most acceptable, followed by b_2. He does not consider any other post acceptable.
– All dummy agents p_j, $j \in [2m]$, have p_j: $(a_1, 1) \succ \ldots \succ (a_m, 1) \succ (a_1, 2) \succ \ldots \succ (a_m, 2) \succ \langle b_1, 1, 2m + 3n\rangle \succ \cdots$.
Briefly put, each dummy agent p_j always wants to go to a set-post with congestion one or two.
– All dummy agents q_j, $j \in [2m]$, only consider b_1 and b_2 acceptable, but prefers b_2 over b_1:
q_j: $\langle b_2, 1, 2m\rangle \succ \langle b_1, 1, 2m + 3n\rangle \succ \cdots$.

Clearly, the construction can be done in polynomial time. One can verify that the constructed preferences do not contain ties. The maximum congestions of the agents are depicted in the following table, where v_i is an element-agent with u_i appearing in C_1, C_2, C_m:

	a_1	a_2	a_3	$\cdots$	a_m	b_1	b_2
v_i	3	3	0	$\cdots$	3	0	$3n + 4m - 9$
p_z	2	2	2	$\cdots$	2	$3n + 2m$	0
q_z	0	0	0	$\cdots$	0	$3n + 2m$	$2m$

The correctness proof is deferred to Appendix C.1 □

We conclude the study of EF+TG with two simple FPT algorithms. The first algorithm is based on brute-force searching all possible TG assignments while the second one on guessing the empty posts and applying Algorithm 1 that checks whether a CP assignment exists.

Theorem 3 (⋆)**.** EF+TG *is FPT with respect to the number n of agents and the number m of posts, respectively.*

Maximally CP assignments. Now, we turn to maximally CP assignments and define the decision variant MAXCP+TG: Given an instance I and a non-negative integer t, does there exists a TG assignment with at most t unsatisfied agents? We first show that MAXCP+TG is W[1]-hard wrt. the number t of unsatisfied agents; the W[1]-hardness is via reducing from the W[1]-complete CLIQUE problem [20]. Fortunately, when t is constant, the problem can be solved in polynomial time.

Theorem 4 (⋆)**.** MAXCP+TG *is W[1]-hard and in XP with respect to the number t of unsatisfied agents. The W[1]-hardness holds even if there are no ties.*

Using an idea similar to the one for Theorem 3 and by applying the algorithm behind Lemma 2, we obtain further parameterized algorithms for MAXCP+TG.

Theorem 5 (⋆)**.** MAXCP+TG *is FPT with respect to n, and in XP with respect to m, where n and m denote the number of agents and the number of posts, respectively.*

Finally, we show that finding maximally CP assignment remains W[1]-hard even if we give up TG.

Theorem 6 (⋆)**.** *Deciding whether an instance of* CONGESTED ASSIGNMENT *has an assignment with at most t unsatisfied agents is W[1]-hard with respect to t.*

5 Conclusion

We investigated congested assignments with congestion-averse agents, focusing on competitiveness (CP), envy-freeness (EF), and maximal competitiveness (maxCP). We devised a novel network-flow-based algorithm to identify CP assignments. We then proved NP-hardness of finding an assignment that is top-guaranteed and either EF or maxCP. We complement these hardness results with several parameterized algorithms. We also show that relaxing top-guarantees does not reduce the complexity: Finding a maxCP assignment remains NP-hard.

For future research, we suggest exploring congested assignments with *weighted* agents [6] and scenarios where agents have varying responses to congestion. Additionally, in applications like urban traffic, exploring control management strategies, such as identifying the minimum number of posts to remove to achieve a competitive assignment, presents another intriguing avenue. Considering that our work mainly focuses on providing theoretical analysis of congested assignment complexity, empirically validating the algorithms shown in this paper could be another interesting direction.

Acknowledgement

This work was supported by the Vienna Science and Technology Fund (WWTF) [10.47379/VRG18012] and the National Natural Science Foundation of China (Grants No. 61772314, 61761136017 and 62072275). We would like to thank the reviewers for their helpful comments.

References

[1] A. Abdulkadiroğlu and T. Sönmez. School choice: A mechanism design approach. *American economic review*, 93(3):729–747, 2003.

[2] H. Ackermann, H. Röglin, and B. Vöcking. On the impact of combinatorial structure on congestion games. *Journal of the ACM*, 55(6):25:1–25:22, 2008.

[3] H. Aziz, S. Gaspers, Z. Sun, and T. Walsh. From matching with diversity constraints to matching with regional quotas. In *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems*, page 377–385, 2019. ISBN 9781450363099.

[4] C. Ballester. NP-completeness in hedonic games. *Games and Economic Behavior*, 49(1):1–30, 2004.

[5] P. Biró, T. Fleiner, R. W. Irving, and D. Manlove. The College Admissions problem with lower and common quotas. *Theoretical Computer Science*, 411(34-36):3136–3153, 2010.

[6] A. Bogomolnaia and H. Moulin. Fair congested assignment. *Mathematics of Operation Research*, pages 1–19, 2025. URL `https://doi.org/10.1287/moor.2024.0581`.

[7] I. Caragiannis, C. Kaklamanis, P. Kanellopoulos, and E. Papaioannou. Scheduling to maximize participation. *Theoretical Computer Science*, 402:142–155, 2008.

[8] D. Chakrabarty, A. Mehta, and V. Nagarajan. Fairness and optimality in congestion games. In *Proceedings 6th ACM Conference on Electronic Commerce*, pages 52–57. ACM, 2005.

[9] J. Chen, R. Ganian, and T. Hamm. Stable matchings with diversity constraints: Affirmative action is beyond NP. In C. Bessiere, editor, *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pages 146–152, 2020.

[10] G. Christodoulou and E. Koutsoupias. The price of anarchy of finite congestion games. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 67–73, 2005.

[11] G. Christodoulou, M. Gairing, Y. Giannakopoulos, D. Poças, and C. Waldmann. Existence and complexity of approximate equilibria in weighted congestion games. *Mathematics of Operations Research*, 48(1):583–602, 2023.

[12] A. Copland. School choice and class size externalities, 2023.

[13] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, 2015.

[14] A. Darmann. A social choice approach to ordinal group activity selection. *Mathematical Social Sciences*, 93:57–66, 2018.

[15] A. Darmann. Stable and pareto optimal group activity selection from ordinal preferences. *International Journal of Game Theory*, 47(4):1183–1209, 2018.

[16] A. Darmann. Manipulability in a group activity selection problem. *Social Choice and Welfare*, 52(3):527–557, 2019.

[17] A. Darmann and J. Lang. Group activity selection problems. In U. Endriss, editor, *Trends in computational social choice*, pages 87–103. AI Access, 2017.

[18] A. Darmann, E. Elkind, S. Kurz, J. Lang, J. Schauer, and G. J. Woeginger. Group activity selection problem. In *Proceedings of the 8th Workshop on Internet and Network Economics*, volume 7695 of *Lecture Notes in Computer Science*, pages 156–169, 2012.

[19] A. Darmann, E. Elkind, S. Kurz, J. Lang, J. Schauer, and G. J. Woeginger. Group activity selection problem with approval preferences. *International Journal of Game Theory*, 47(3): 767–796, 2018.

[20] R. G. Downey and M. R. Fellows. Fixed-parameter tractability and completeness II: On completeness for W[1]. *Theoretical Computer Science*, 141(1&2):109–131, 1995.

[21] R. Ganian, S. Ordyniak, and C. S. Rahul. Group activity selection with few agent types. *Algorithmica*, 85(5):1111–1155, 2023.

[22] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical computer science*, 38:293–306, 1985.

[23] S. Gupta, S. Roy, S. Saurabh, and M. Zehavi. Group activity selection on graphs: Parameterized analysis. In *Proceedings of the 10th international symposium on algorithmic game theory*, volume 10504 of *Lecture Notes in Computer Science*, pages 106–118, 2017.

[24] D. Gusfield. Three fast algorithms for four problems in stable marriage. *SIAM J. Comput.*, 16 (1):111–128, 1987.

[25] K. Hamada, K. Iwama, and S. Miyazaki. The Hospitals/Residents problem with lower quotas. *Algorithmica*, 74(1):440–465, 2016.

[26] A. Hayrapetyan, É. Tardos, and T. Wexler. The effect of collusion in congestion games. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 89–98, 2006.

[27] A. Igarashi, D. Peters, and E. Elkind. Group activity selection on social networks. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 565–571, 2017.

[28] H. Konishi, S. Weber, and M. L. Breton. Free mobility equilibrium in a local public goods economy with congestion. *Research in Economics*, 51:19–30, 1997.

[29] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer, 2007.

[30] H. Lee and Y. Shoham. Stable invitations. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 965–971, 2015.

[31] D. F. Manlove, R. W. Irving, K. Iwama, S. Miyazaki, and Y. Morita. Hard variants of stable marriage. *Theor. Comput. Sci.*, 276(1-2):261–279, 2002.

[32] C. A. Meyers and A. S. Schulz. The complexity of welfare maximization in congestion games. *Networks*, 59(2):252–260, 2012.

[33] I. Milchtaich. Congestion games with player-specific payoff functions. *Games & Economic Behavior*, 13(27):111–124, 1996.

[34] M. Milinski. An evolutionarily stable feeding strategy in sticklebacks. *Ethology*, 51(1):36–40, 1979.

[35] W. Phan, R. Tierney, and Y. Zhou. Crowding in school choice. Technical report, Kyoto University, 2021.

[36] R. W. Rosenthal. A class of games possessing pure-strategy nash equilibria. *International Journal of Game Theory*, 2:65–67, 1973.

[37] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge, 2012.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We provide proofs for all stated results. They can be found in the main part or appendix.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The model studied in our paper is a theoretical and abstract model. Our analysis is based on worst case analysis. There are no experiments.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We provide proofs for all stated results. They can be found in the main part or appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [NA]

   Justification: There are no experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [NA]

Justification: As mentioned, there are no experiments. But all stated results are proved in the main part or appendix.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [NA]

   Justification: As mentioned, there are no experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: As mentioned, there are no experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [NA]

   Justification: As mentioned, there are no experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research is theoretical and has no harm to the society or human.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We investigated fair assignment with congestion-averse agents. Fairness is a relevant property that a society would want to achieve. Hence, it could have potentially positive impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our research does not have experiments.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: As mentioned, there are no experiments.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: As mentioned, there are no experiments.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: As mentioned, there are no experiments.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

    Justification: As mentioned, there are no experiments.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: Our research is original.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Supplementary Material for the Paper "Assignments for Congestion-Averse Agents: Seeking Competitive and Envy-Free Solutions"

A Additional Material for Section 2

A.1 Proof of Lemma 1

Lemma 1 ($\star$). *(1) CP implies EF, but the converse does not hold.*
(2) CP implies NS, but the converse does not hold.
(3) NS implies TG and NW, but the converse does not hold.
(4) EF is incomparable to NS, to TG, and to NW, respectively; TG is incomparable to NW.

Proof. (1) Clearly, CP implies EF by definition. Now, to show that the converse does not hold, let us consider Example 1. Clearly assigning every agent to post a_1 is EF, but it is not CP since all agents prefer $(a_2, 1)$ to $(a_1, 3)$.

(2) As already mentioned, the implication has been discussed by Bogomolnaia and Moulin [6] already. For the sake of completeness, we provide a proof by showing the contra-positive.
Let Π be an assignment that is not NS and let there be an agent $v \in V$ and a post $a \in A$ such that v prefers $(a, |\Pi(a)| + 1)$ to $(a^*, |\Pi(a^*)|)$ where a^* denotes the post that v is assigned to. If a is empty, then clearly, Π is wasteful and hence not CP. If a is non-empty, then since every agent is averse against congestions, we infer that v prefers $(a, |\Pi(a)|)$ to $(a^*, |\Pi(a^*)|)$, and hence not CP either.
Now, to show that the converse does not hold, let us consider Example 1 again. As already discussed there, Π_1 is NS, but not CP.

(3) That NS implies NW follows directly from definition. That "NS implies TG" has also been shown by Bogomolnaia and Moulin [6]. Again, for the sake of completeness, we provide a proof here by showing the contra-positive. Let Π be an assignment that is not TG, and let $v \in V$ be an agent and a^* a post such that v is assigned to a^* while $(a^*, |\Pi(a^*)|)$ is *not* among his top-$|V|$ choices. Let $X = \{(a, d) \mid (a, d) \succ_v (a^*, |\Pi(a^*)|)\}$ be the set consisting of all tuples that v prefers to $(a^*, |\Pi(a^*)|)$. Then, $|X| \geq |V|$. For each $a \in A$, let $\delta(a)$ denote the largest congestion such that $(a, \delta(a)) \in X$, i.e., $\delta(a) = \max_{(a,d)\in X}\{d\}$ and $\delta(a) = 0$ if no tuple (a, d) exists in X. Then, $\sum_{a\in A} \delta(a) = |X| \geq |V| = \sum_{a\in A} |\Pi(a)|$. By assumption, we have that $\delta(a^*) < |\Pi(a^*)|$. This implies that there must exist a post $a \in A \setminus \{a^*\}$ such that $\delta(a) > |\Pi(a)|$. By definition, we infer that $(a, |\Pi(a)| + 1) \in X$, and hence $(a, |\Pi(a)| + 1) \succ_v (a^*, |\Pi(a^*)|)$, witnessing that Π is not NS.
It is quite straightforward to come up with a TG and NW assignment which is not NS. Let us consider the following example.

$v_1\colon\ (a_1, 1) \succ (a_1, 2) \succ (a_2, 1) \succ \cdots,$
$v_2\colon\ (a_1, 1) \succ (a_2, 1) \succ (a_1, 2) \succ \cdots,$
$v_3\colon\ (a_3, 1) \succ (a_3, 2) \succ (a_3, 3) \succ \cdots.$

$\Pi_3\colon$

a_1	a_2	a_3
v_2	v_1	v_3

$\Pi_4\colon$

a_1	a_2	a_3
v_1, v_2		v_3

Π_3 is clearly TG and NW. It is not NS however, since v_1 prefers $(a_1, 2)$ to $(a_2, 1)$.

(4) Let us consider Example 1. Assigning every agent to the same post is clearly EF, but not TG and not NW. Hence, it is not NS by Statement (3). As already argued in Example 1, Π_1 is NS, TG, and NW, but not EF.
Assigning v_1 and v_2 to a_2, and v_3 to a_1 is NW, but not TG: For v_1, tuple $(a_2, 2)$ is *not* in his top 3 choices.
Now, let us consider the example from item. Π_4 is TG but not NW. □

By definition, we observe the following:

Observation 2. *For an arbitrary tie-breaking rule, $\sum_{a\in A} \lambda(v, a) = |V|$ holds for every $v \in V$.*

A.2 Proof of Lemma 2

Lemma 2 ($\star$). *Given a congestion vector $\vec{s}$ with $\sum_{a\in A} \vec{s}[a] = |V|$, in polynomial-time one can determine the smallest number t of unsatisfied agents among all assignments whose congestion vectors equal $\vec{s}$; the corresponding assignment can found in polynomial time.*

Proof. The idea is to iterate over all possible number $t \in \{0, 1, \ldots, |V|\}$ and check whether there exists an assignment with congestion vector $\vec{s}$ and exactly t unsatisfied agents. The later problem can be solved via determining whether a perfect $\vec{b}$-matching exists, which can be done in polynomial time [29, Chapter 12].

Let $(A, V, (\succeq)_{v \in V})$ be an instance of CONGESTED ASSIGNMENT. To check whether there exists an assignment with congestion vector $\vec{s}$ and exactly t unsatisfied agents, we construct a bipartite graph G on two disjoints X and Y with $X = A \cup \{a_0\}$ and $Y = V \cup \{w_1, \ldots, w_t\}$, where the w_z's are the dummy *agent-vertices* and a_0 is a dummy *post-vertex*.

We add an edge between every original post a_j and every dummy agent-vertex w_z, and an edge between the dummy post-vertex a_0 and every original agent v_i. We also add an edge between every original post and original agent, but will delete some according to the congestion vector. In other words, the graph on X and Y is almost a complete bipartite graph, except there are no edges between the dummy post a_0 and any dummy agent w_z, $z \in [t]$.

We delete from the bipartite graph the following edges: For each original agent v and each two original posts a and a', if v prefers $(a, \max(1, \vec{s}[a]))$ to $(a', \vec{s}[a'])$, then we delete the edge $\{a', v\}$. This is because if v would be satisfied, he will never be assigned to a' since either the post is wasteful or he envies some agent that is assigned to a.

This completes the construction of the graph G. We check whether there exists a *perfect* $\vec{b}$-matching[4] for G where $\vec{b}[a_0] = t$, $\vec{b}[a_j] = \vec{s}[a_j]$ for all $a_j \in A$, and $\vec{b}[y] = 1$ for all $y \in Y$. We answer no if no such matching exists. Otherwise, let M be the perfect $\vec{b}$-matching, and we return the following partition as assignment Π. For each post $a_j \in A$ and agent $v_i \in V$, let $\Pi(v_i) = a_j$ if $\{v_i, a_j\} \in M$. Let V' be the remaining agents that are unassigned; note that these agents are matched to a_0 by definition. As long as the congestion of some $a_j \in A$ is not equal to $\vec{s}[a_j] = \vec{b}[a_j]$, pick an agent from V' and assign him to a_j.

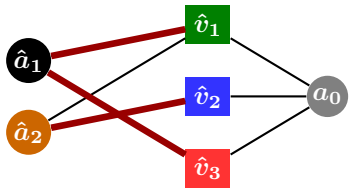
For the correctness, it is straightforward that if Π is an assignment with congestion vector $\vec{s}$ and exactly t unsatisfied agents V', then the following matching is a perfect $\vec{b}$ matching: Let $M(v_i) = a_j$ if $v_i \in V \setminus V'$, and $M(v_i) = a_0$ if $v_i \in V'$. Finally, for each original post a_j, if it is assigned $\hat{n}$ unsatisfied agents, then we pick $\hat{n}$ distinct dummy agent-vertices and match them to a_j.

If M is a perfect $\vec{b}$-matching for G, then a_0 is matched with exactly t original agents V' who will be the unsatisfied agents. Clearly, the assignment Π given by our algorithm above has the desired congestion vector $\vec{s}$. We show that only the agents from V' may be unsatisfied. Consider an arbitrary agent $v_i \in V \setminus V'$ and post $a_j \in A \setminus \{\Pi(v_i)\}$. For a contradiction, suppose v_i prefers $(a_j, \max(1, |\Pi(a_j)|))$ to $(a^*, |\Pi(a^*)|)$, where v_i is assigned to a^*, i.e., $\{v_i, a^*\} \in M$. By the definition of Π, it follows that v_i prefers $(a_j, \max(1, \vec{s}(a_j)))$ to $(a^*, \vec{s}(a^*))$, implying that edge $\{v_i, a^*\}$ does not exist in the constructed bipartite graph and cannot be matched under M, a contradiction.

Since checking the existence of a perfect $\vec{b}$-matching and finding such a matching if it exists can be done in polynomial time by reducing to finding a perfect matching, the whole approach can be done in polynomial time as well. This completes the proof.

For an illustration, consider the first instance in Example 1. Let the congestion vector be $\vec{s} = (2, 1)$ and the number of unsatisfied agents be $t = 0$. The following bipartite graph G has a perfect $\vec{b}$-matching, indicated by the red lines.

$\vec{b}$:	a_0	$\hat{a}_1$	$\hat{a}_2$	$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$
	0	2	1	1	1	1



Indeed, the corresponding $\vec{b}$-matching yields a CP assignment which is Π_2 and it has only satisfied agents. □

[4] A $\vec{b}$-matching M is *perfect* if $\sum_{e \in M\colon u \in e} 1 = \vec{b}[u]$ holds for all vertices u.

B Additional Material for Section 3.1

B.1 Example of Construction 1

Let us consider the four agents with the following preference lists.

$$\begin{array}{ll} v_1\colon & (a_1,1) \succ (a_2,1) \succ (a_3,1) \sim (a_1,2) \sim (a_2,2) \succ \cdots, \\ v_2, v_3\colon & (a_2,1) \succ (a_1,1) \succ (a_3,1) \sim (a_1,2) \sim (a_2,2) \succ \cdots, \\ v_4\colon & (a_1,1) \sim (a_2,1) \succ (a_1,2) \sim (a_2,2) \succ (a_3,1) \succ \cdots, \end{array}$$

One can observe that if every post is non-empty, then a_1 or a_2 will have congestion one. If a_1 has congestion one, then v_1 has to be assigned to a_1 alone and v_4 to a_2 alone, leaving v_2 and v_3 to be envious. If a_2 has congestion one, then v_2 or v_3 will be envious. One can verify that assigning any two agents to a_1 and the remaining two to a_2 is competitive, leaving a_3 empty.

Now, let us "guess" that the number of empty post is $k = 1$. For $k = 1$, we augment the instance with one dummy agent u_1 and two auxiliary agents p_1 and p_2, and two dummy posts b_1 and b_2. Their preference lists are as follows:

$$\begin{array}{ll} u_1\colon & (a_1,1) \sim (a_2,1) \sim (a_3,1) \succ (b_1,1) \succ (b_1,2) \succ (b_1,3); \\ p_1\colon & (b_1,1) \succ (b_2,1) \succ (b_2,2) \succ \cdots \succ (b_2,5); \\ p_2\colon & (b_2,1) \succ (b_1,1) \succ (b_1,2) \succ \cdots \succ (b_1,5). \end{array}$$

One can verify that in the original instance, every CP assignment will leave a_3 empty, and in the augmented instance, every CP assignment will assign the dummy agent u_1 to a_3 alone. The correctness is given by Lemma 3.

B.2 Example of Algorithm 1

Consider the first instance in Example 1 and let us assume that we are in the case with $k = 0$, meaning that we can ignore a_0 and the dummy agents. Initially, $T[a_1] = T[a_2] = 1$, implying that the condition in Line 4 is satisfied, so we can start with the first iteration. In the first iteration, where $T = (1, 1)$, the network and its maximum flow constructed in Phase 1 (Lines 5–6) have been discussed in Example 2 already. Hence, we proceed with line 7. Let the maximum flow f be as indicated in Example 2, i.e., $f(\hat{a}_1, t) = f(\hat{a}_2, t) = 1$ and $f(\hat{a}_3, t) = 0$. Since f does not have value $|V| = 3$, we proceed with Phase 2, where we find an obstruction in Lines 8–14. We start with $V' = \{\hat{v}_3\}$ and $A' = \emptyset$ since $\hat{v}_3$ is the only vertex with $f(\hat{v}_3, t) = 0$. In the while-loop in lines 10–14, we first find $\hat{a}_1$ and compute $A' = \{\hat{a}_1\}$. Then, we compute $V' = \{\hat{v}_3, \hat{v}_1\}$ in line 13. Since no further post $\hat{a}$ exists that has an arc to any agent $\hat{v}$ in V' (see the figure in Example 2), we stop with $A' = \{\hat{a}_1\}$ and $V' = \{\hat{v}_3, \hat{v}_1\}$. One can check that if a_1 would stay with congestion one, then no CP assignment can exist since both v_1 and v_3 would envy the only agent that is assigned to a_1. We will later show that in order to have a CP assignment, it is necessary to increase the congestion of every post in A'. In Phase 3, we increment $T[a_1]$ to 2, while the other post stays with $T[a_2] = 1$. This completes the first iteration.

At line 4, since $T[a_1] + T[a_2] = 3 \leq |V|$, we continue with the second iteration. In the following, the tuples considered in Construction 2(v) are boldfaced.

$$\begin{array}{ll} v_1\colon & (a_1,1) \succ \boldsymbol{(a_1,2)} \sim \boldsymbol{(a_2,1)} \succ (a_2,2) \succ \cdots, \\ v_2\colon & (a_1,1) \sim \boldsymbol{(a_2,1)} \succ (a_1,2) \sim (a_2,2) \succ \cdots, \\ v_3\colon & (a_1,1) \succ \boldsymbol{(a_1,2)} \succ (a_2,1) \succ (a_2,2) \succ \cdots. \end{array}$$

One can verify that among all tuples $(a, T[a])$, $a \in A$, tuple $(a_1, 2)$ is the most preferred tuple of v_1 and v_3, while $(a_2, 1)$ is the most preferred tuple of v_1 and v_2. Hence, the network and maximum flow (highlighted with red lines) constructed in Phase 1 are as given in Figure 2; note that this corresponds to the bipartite graph in Appendix A.2.

In Line 6, we verify that the maximum flow is a perfect flow (i.e., with value $|V|$). Hence, we derive and return an assignment according to Definition 2. This is exactly Π_2 from Example 1.

One can verify that the second instance of Example 1 where every agent has the same preference list as v_2 will lead to the sum of the congestion entries to exceed $|V| = 3$ in the second iteration, certifying that the instance does not have a CP assignment, no matter with or without empty posts.

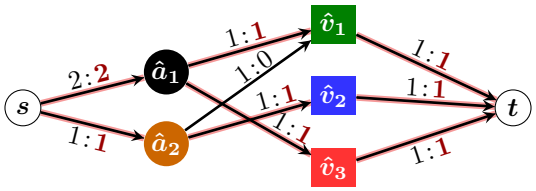


Figure 2: Flow network for iteration 2 where $T = (2, 1)$. For more information, see Appendix B.2.

B.3 Proof of Lemma 4

Lemma 4 ($\star$). *Each (A', V') computed in Phase 2 in lines 8–14 is an obstruction.*

Proof. Let (A', V') be the pair computed in lines 8–14 in some iteration z. Let (G, c) be the network with $G = (\hat{A} \cup \hat{V} \cup \{s, t\}, E)$ and f the maximum flow computed in this iteration. We aim to show that (A', V') satisfies the properties in Definition 3.

By line 7, f fails to have value $|V|$, i.e., $\sum_{\hat{v} \in \hat{V}} f(\hat{v}, t) < |V|$. Hence, there must be a vertex $\hat{v}^* \in \hat{V}$ with $f(\hat{v}^*, t) = 0$. Let $\hat{v}^*$ be such a vertex that is added to V' in line 9. Then, the first part of property (i) is clear since $\hat{v}^* \in V'$ (line 9) and we only add vertices from $\hat{V}$ to V'; see line 13.

Property (ii) is also clear due to line 11.

Let us consider property (iii). Clearly, for every vertex $\hat{a} \in A'$ with $(\hat{a}, \hat{v}^*) \in E(G)$ we must have that $f(s, \hat{a}) = \mathsf{c}(s, \hat{a})$ as otherwise we could increase the flow by one by setting $f(s, \hat{a}) = f(s, \hat{a}) + 1$ and $f(\hat{a}, \hat{v}^*) = f(\hat{v}^*, t) = 1$. By line 13, every out-neighbor $\hat{v}$ of $\hat{a}$ with $f(\hat{a}, \hat{v}) = 1$ is added to V'. Together with $\hat{v}^*$, we obtain that $\mathsf{c}(s, \hat{a}) < |\{\hat{v} \in V' \mid (\hat{a}, \hat{v}) \in E\}|$ since $f(v^*, t) = 0$, as desired.

Consider an arbitrary vertex $\hat{a} \in A'$ with $(\hat{a}, \hat{v}^*) \notin E(G)$. Suppose, towards a contradiction, that $\hat{a}$ does not satisfy property (iii), meaning that $\mathsf{c}(s, \hat{a}) \geq |\{\hat{v'} \in V' \mid (\hat{a}, \hat{v'}) \in E(G)\}|$. We aim to show that there is an "augmenting" path from $\hat{a}$ to $\hat{v}^*$, with arcs having flow values alternating between zero and one, which is a witness for the flow to be not maximum.

Let us go through the **repeat**-loop in lines 10–14. Observe that in each round of this loop, we aim at finding a vertex $\hat{a}$ not already in A' that has an out-arc to some agent-vertex from V'; V' is initialized with $V' = \{\hat{v}^*\}$. This implies that we can find a vertex in $\hat{v}_x \in V' \setminus \{\hat{v}^*\}$ due to which we add $\hat{a}$ in line 11. Further, for each vertex $\hat{v'}$ in $V' \setminus \{\hat{v}^*\}$, we can also find a vertex $\hat{a'}$ in a previous round such that $f(\hat{a'}, \hat{v'}) = 1$ in line 13. Let $\hat{a}_x$ be the vertex from A' due to which we add $\hat{v}_x$, i.e., $f(\hat{a}_x, \hat{v}_x) = 1$. Since each vertex in V' has only one out-arc with capacity one, due to the conservation constraint of the flow f, we infer that $f(\hat{a}, \hat{v}_x) = 0$; recall that $f(\hat{a}_x, \hat{v}_x) = 1$. Repeating the above reasoning, there must be a vertex $\hat{v}_{x-1}$ from $V' \setminus \{\hat{v}_x\}$ due to which we add $\hat{a}_x$. Then, either $\hat{v}_{x-1} = \hat{v}^*$ or $\hat{v}_{x-1} \neq \hat{v}^*$.

In the former case, we infer that $(\hat{a}, \hat{v}_x, \hat{a}_x, \hat{v}^*)$ is an augmenting path since by assumption $\hat{a}$ has enough capacity to accommodate all incident agents, including $\hat{v}_x$. Thus, flipping the flow values along the path would increase the value of the flow:

$$f(s, \hat{a}) = f(s, \hat{a}) + 1, \quad f(\hat{a}, \hat{v}_x) = 1, \quad f(\hat{a}_x, \hat{v}_x) = 0, \quad f(\hat{a}_x, \hat{v}^*) = f(\hat{v}^*, t) = 1, \text{ a contradiction.}$$

In the latter case, since V' is finite and no vertex from $\hat{V}$ can obtain more than one positive flow, by repeating the above reasoning, we must end up with an arc to $\hat{v}^*$ with zero flow; recall that $\hat{v}^* \in V'$. Then, we again obtain an augmenting path $P = (\hat{a}, \hat{v}_x, \hat{a}_x, \ldots, \hat{a}_0, \hat{v}_0 = \hat{v}^*)$. Analogously, we can increase the total flow by flipping the flow values along this path, a contradiction.

It remains to show property (iv). This is clear since otherwise for each vertex $\hat{v} \in V'$ we could find a vertex $\hat{a} \in A'$ and set $f(\hat{a}, \hat{v}) = f(\hat{v}, t) = 1$. In particular, the starting vertex $\hat{v}^*$ would have positive flow going through it, a contradiction. □

B.4 Proof of Lemma 5

Lemma 5 ($\star$)**.** *Assume that I admits a CP assignment Π with no posts being empty. Then, for each iteration $z \geq 1$, each obstruction (A', V') found in iteration z, and each post $a \in A$, the following holds. If $\hat{a} \in A'$, then $|\Pi(a)| \geq T_z[a] + 1$; otherwise $|\Pi(a)| \geq T_z[a]$.*

Proof. Let us consider the first iteration and let (A', V') be the found obstruction. Since Π does not have empty posts, the statement clearly holds for all posts $a \in A$ with $\hat{a} \notin A'$. Let $N = (G, \mathsf{c})$ denote the network and f the maximum flow of N computed in the first phase. Let $P = \{v \in V \mid \hat{v} \in V'\}$ and $Q = \{a \in A \mid \hat{a} \in A'\}$ be the set of agents and posts that correspond to the vertices in V' and A', respectively.

Suppose, for the sake of contradiction, that there exists a post $a \in Q$ with $|\Pi(a)| \leq T_1[a] = 1$. By Definition 3(iii), more than $\mathsf{c}(s, \hat{a}) = T_1[a] = 1$ vertex from V' is incident to $\hat{a}$ in G. By Construction 2(v), at least two agents from P consider $(a, 1)$ as one of the most preferred tuples.

Since $|\Pi(a)| \leq 1$, at least one agent from P is not assigned to a but considers $(a, 1)$ as one of the most preferred tuples. Let $v_0 \in P$ be such an agent. Then, he must be assigned to some other post a_0 such that $(a_0, |\Pi(a_0)|)$ is one of the most preferred tuples for v_0 as well. This implies that $|\Pi(a_0)| = 1$ since we are in the first iteration. By Construction 2(v), we infer that $(\hat{a}_0, \hat{v}_0) \in E(G)$, and by line 11, that $\hat{a}_0 \in A'$.

By Definition 3(iii), more than $\mathsf{c}(s, \hat{a}_0) = T_1[a_0] = 1$ vertex from V' is incident to $\hat{a}_0$ in G, and we can find another agent $v_1 \in P$ that is not assigned to a_0 but considers $(a_0, 1)$ as one of the most preferred tuples. Again, this agent v_1 will be assigned to some post a_1 with $(a_1, 1)$ being one of the most preferred tuples of v_1. By Construction 2(v), we infer that $(\hat{a}_1, \hat{v}_1) \in E(G)$, and by line 11 that $\hat{a}_1 \in A'$. Repeating the above reasoning, we will be able to find a distinct vertex $\hat{a}_i \in A'$ for each vertex $\hat{v}_i \in V'$ such that $\Pi(a_i) = \{v_i\}$. That is, $|A'| \geq |V'|$, a contradiction to Definition 3(iv) since $|A'| = \sum_{\hat{a} \in A'} \mathsf{c}(s, \hat{a}) < |V'|$ in this case.

Now, let us consider other iterations. For each z, let (A'_z, V'_z) be the obstruction found in iteration z. Note that the table entries never decrease. Hence, if the statement were incorrect, there must be an iteration $z \geq 2$ where the statement holds in all iterations $z' \leq z - 1$ but not in iteration z.

Suppose, for the sake of contradiction, that the statement is incorrect and let z be the index of the first such iteration where the statement is incorrect. That is, in all iterations $z' \leq z - 1$, we have that for all $a' \in A$,

$$\text{if } \hat{a}' \in A'_z, \text{ then } |\Pi(a')| \geq T_{z'}[a'] + 1; \text{otherwise } |\Pi(a')| \geq T_{z'}[a'] \tag{1}$$

but there exists a post a such that

$$\text{if } \hat{a} \in A'_z, |\Pi(a)| \leq T_z[a]; \text{ if } \hat{a} \notin A'_z, \text{ then } |\Pi(a)| < T_z[a]. \tag{2}$$

First, observe that $\hat{a} \in A'_z$ since otherwise $\Pi(a) \geq T_{z-1}[a] = T_z[a]$ by line 15, a contradiction to the assumption.

Next, we claim that $T_z[a] = |\Pi(a)|$. If $\hat{a} \in A'_{z-1}$ (i.e., $\hat{a}$ was in the obstruction found in iteration $z - 1$), then by assumption (1)–(2) and by line 15, we infer $|\Pi(a)| \geq T_{z-1}[a] + 1 = T_z[a] \geq |\Pi(a)|$, as desired. If $\hat{a} \notin A'_{z-1}$, then again by assumption (1)–(2) and by line 15, we infer $|\Pi(a)| \geq T_{z-1}[a] = T_z[a] \geq |\Pi(a)|$, as desired as well.

Recall that we inferred that $\hat{a} \in A'_z$. Let (G, c) denote the network constructed in iteration z. By Lemma 4, (A'_z, V'_z) is an obstruction for (G, c). Let $\hat{v}^* \in V'$ be the starting vertex with $f(\hat{v}^*, t) = 0$. By Definition 3(iii), more than $\mathsf{c}(s, \hat{a}) = T_z[a]$ vertices from V'_z exist that have a as an in-neighbor. By Construction 2 (v), more than $T_z[a] = |\Pi(a)|$ agents from V'_z consider $(a, T_z[a])$ as one of the most-preferred tuples among all $(a', T_z[a'])$. Hence, at least one of such agents is not assigned to a by Π.

Let $\hat{v} \in V'_z$ be a vertex whose corresponding agent v considers $(a, |\Pi(a)|)$ as one of the most preferred tuples but is assigned to some other post $a' \neq a$. Then, $(\hat{a}, \hat{v}) \in E(G)$. To prevent v from being envious (recall that no post is empty), we must have that $(a', |\Pi(a')|) \succeq_v (a, |\Pi(a)|)$.

We claim that $\hat{a}' \in A'_z$ as well. Since $(a, |\Pi(a)|)$ is one of the most-preferred tuples of v among all $(a', T_z[a'])$, by previous paragraph and by congestion aversion, we infer that $T_z[a'] \geq |\Pi(a')|$. If $T_z[a'] > |\Pi(a')|$, then by line 15, there must exists an iteration $z' \in [z - 1]$ where $\hat{a}' \in A'_{z'}$

and $T_{z'}[a'] = |\Pi(a')|$, a contradiction to (1). Hence, $T_z[a'] = |\Pi(a')|$, implying that $(a', T_z[a'])$ is also one of the most preferred tuples of v among all $(a'', T_z[a''])$. By Construction 2(v), we have $(\hat{a}', \hat{v}') \in E(G)$, and by line 11, we have $\hat{a}' \in A'_z$, as desired.

By Definition 3(iii), we infer that more than $\mathsf{c}(a') = T_z[a'] = |\Pi(a')|$ vertices from V'_z have in-arcs from $\hat{a'}$. By Construction 2(v), more than $\mathsf{c}(a') = T_z[a'] = |\Pi(a')|$ agents consider $(a', |\Pi(a')|)$ as one of the most preferred tuples among all $(p, T[p]), p \in A$.

Analogously, we can again find another vertex $\hat{v'} \in V'_z$ such that v' considers $(a', |\Pi(a')|)$ as one of the most preferred tuples among all $(p, T[p]), p \in A$, but is assigned to some other post $a'' \neq a'$ with $\hat{a''} \in A'_z$ and $T_z[a''] = |\Pi(a'')|$. By repeating this argument, we infer that every vertex $\hat{\alpha} \in A'_z$ has $T_z[\alpha] = |\Pi(\alpha)|$. By Definition 3(iv), we have that $|V'_z| > \sum_{\hat{\alpha} \in A'_z} \mathsf{c}(s, \hat{\alpha}) = \sum_{\hat{\alpha} \in A'_z} T_z[\alpha] = \sum_{\hat{\alpha} \in A'_z} |\Pi(\alpha)|$. So there must be a vertex $\hat{\mu} \in V'_z$ such that μ is assigned to a post b with $\hat{b} \notin A'_z$. By line 11 and Construction 2(v), let $\hat{\alpha} \in A'_z$ with $(\hat{\alpha}, \hat{\mu}) \in E(G)$ such that $(\alpha, T_z[\alpha])$ is a most preferred tuple of μ among all tuples $(p, T_z[p])$, $p \in A$.

By our previous argument, we have that $T_z[\alpha] = |\Pi(\alpha)|$. By CP, we have that $(b, |\Pi(b)|) \succeq_v (\alpha, |\Pi(\alpha)|)$. Since $(\alpha, T_z[\alpha])$ is a most preferred tuple of μ among all tuples $(p, T_z[p])$, $p \in A$, we further infer that $T_z[b] \geq |\Pi(b)|$.

Since $b \notin A'_z$, meaning by line 11 that $(\hat{b}, \hat{\mu}) \notin E(G)$, by Construction 2(v), we further infer that $|\Pi(b)| < T_z[b]$. By line 15, there must exist an iteration $z' \in [z-1]$ with $T_{z'}[b] = \Pi(b)$ and $T_{z'}[b]$ was incremented. This is a contradiction to (1) however. □

B.5 Proof of Lemma 6

Lemma 6 (⋆). *If* Π *is an assignment returned in line 7, then* Π *is CP and has no empty post.*

Proof. Let z be the integration and f be the perfect flow based on which Π is computed in line 7. By the definition of perfectness (see Definition 2), the value of f equals the number $|V|$ of agents. This means that $\sum_{a \in A} |\Pi(a)| = |V|$. By the capacity constraints, we obtain that $|V| = \sum_{a \in A} |\Pi(a)| \leq \sum_{a \in A} T_z[a] \leq |V|$, the last inequality holds due to the while-loop-condition in line 4. Hence, for each post $a \in A$ we must have that $|\Pi(a)| = T_z[a]$ since $|\Pi(a)| \leq T_z[a]$ holds by the capacity constraints in Construction 2(iii).

This implies that $\Pi(a) \neq \emptyset$ since $T_z[a] \geq 1$. Hence, to show that Π is CP, it suffices to show that for each agent v that is assigned to a post a and for each post a' with $a' \neq a$ it holds that $(a, |\Pi(a)|) \succeq_v (a', |\Pi(a')|)$. Towards a contradiction, suppose that $(a', |\Pi(a')|) \succ_v (a, |\Pi(a)|)$. By the reasoning above, it follows that $(a', T_z[a']) \succ_v (a, T_z[a])$, a contradiction to Construction 2(v). □

B.6 Continuation of the proof of Theorem 1

Theorem 1 (⋆). *Algorithm 1 correctly decides whether an instance has a CP assignment in* $O(m^2 \cdot (n+m)^2)$ *time, where* m *and* n *denote the number of posts and agents, respectively.*

It remains to analyze the running time. The main body of the algorithm is a for-loop (line 1) and has at most m iterations. In each iteration k, the algorithm constructs a new instance I according to Construction 1. Note that I has $O(n+m)$ agents and $O(m)$ posts, and it can be constructed in $O((n+m)^2)$ time since each agent has $O(n+m)$ tuples in his preference list. Then, we continue with the big **while**-loop in lines 4–16. If we can show the **while**-loop run in $O(m \cdot (n+m)^2)$ time, we obtain our desired running time of $O(m^2 \cdot (n+m)^2)$.

So, it remains to analyze the **while**-loop. In line 3, initializing the table T needs $O(m)$ time. The while-loop (lines 4–15) runs at most n times since no table entries are ever decreased and in each iteration at least one table entry is increased by one. For each iteration, we first construct a network $N = (G, \mathsf{c})$ based on (I, T); see Construction 2. The directed graph G has $O(n+m)$ vertices and $O(m \cdot n)$ arcs, and each capacity value is in $O(n)$. Hence, constructing the network needs $O(m \cdot n)$ time.

Afterwards, there are three phases. The first phase (lines 5–7) finds a maximum flow for N and checks whether its value is $|V|$. Computing a maximum flow can be done in $O(m \cdot n)$ time and comparing two values needs constant time. Hence, the first phase needs $O(m \cdot n)$ time.

The second phase (lines 8–14) finds an obstruction (A', V') by first finding a vertex $\hat{v}^*$ with $f(\hat{v}^*, t) = 0$. This can be done in $O(1)$ time if we store such information when we compare the value of the flow with $|V|$ in the first phase. Hence, the initialization of V' and A' needs $O(1)$ time. Then, the algorithm goes to the repeat-loop in lines 10–14. To analyze the running time of this loop, we observe that there are $O(m \cdot n)$ arcs between A' and V' and each arc only needs to be checked at most once during the whole loop (line 11). Adding new vertices to V' can be done in $O(m \cdot n)$ time as well since for each newly added alternative $\hat{a}$ there are at most n vertices $\hat{v}$ from $\hat{V}$ with positive flow from $\hat{a}$ to $\hat{v}$. Hence, the repeat-loop needs $O(m \cdot n)$ time.

It is straightforward that the last phase (lines 15–15) runs in $O(m)$ time. Summarizing, we obtain that the desired $O(m \cdot n^2)$ time for the **while**-loop.

C Additional Material for Section 4

C.1 Correctness of the Construction in the Proof of Theorem 2

Theorem 2 ($\star$)**.** EF+TG *is NP-complete; hardness holds even if there are no ties.*

Proof of the correctness of the construction. **Correctness.** It remains to show the correctness, i.e., I has an exact cover if and only if I' admits an EF and TG assignment.

For the "only if" part, let $J \subseteq [m]$ denote an exact cover for I. Then, we claim that the following assignment Π is EF+TG:

- For each $j \in J$, let $\Pi(a_j) = \{v_i \mid u_i \in C_j\}$.
- For each $j \in [m] \setminus J$, let $\Pi(a_j) = \emptyset$.
- Let $\Pi(b_1) = \{p_j \mid j \in [2m]\}$ and $\Pi(b_2) = \{q_j \mid j \in [2m]\}$.

Since each set-post contains either zero or three agents, no dummy agent envies any element-agent. The dummy agents also do not envy each other due to their preferences. Similarly, no two element-agents envy each other and no element-agent envies any dummy agent since he does not like b_1 or b_2 more.

For "if" part, let Π be an EF and TG assignment for the constructed instance. We aim at showing that the set-posts that are assigned element-agents constitute an exact cover. To this end, let $J = \{j \mid \exists v_i \text{ with } v_i \in \Pi(a_j)\}$. We first show two claims.

Claim C.1.1. *For each set-post a_j it holds that $|\Pi(a_j)| \in \{0, 3\}$.*

Proof. Since there are $2m$ dummy agents $\{p_1, p_2, \ldots, p_{2m}\}$, but there are only m set-posts, at least one dummy agent, say p_z, is not assigned to a set-post alone. Hence, for every set-post a_j, it holds that $|\Pi(a_j)| \neq 1$ since otherwise p_z would envy the agent that is assigned to a_j. Since the maximum congestion for every set-post is 3, we further infer that $|\Pi(a_j)| \in \{0, 2, 3\}$ holds for every set-post a_j. In particular, this implies that no dummy agent p_z with $1 \leq z \leq 2m$ is assigned to a set-post alone.

Towards a contradiction, suppose that there exists a set-post a_j with $|\Pi(a_j)| \notin \{0, 3\}$. This implies that $|\Pi(a_j)| = 2$. Then, every dummy agent p_z with $1 \leq z \leq 2m$ is to be assigned a set-post since otherwise he would envy the two agents that are assigned to a_j. Since there are exactly $2m$ dummy agents$p_1, \ldots, p_{2m}$, this means that every set-post a_x, $x \in [m]$, must have $|\Pi(x)| = 2$. Then, no other agent can be assigned to the set-post. However, all element-agents will envy all p_i's, a contradiction. This concludes the proof. (end of the proof of Claim C.1.1 $\diamond$)

By Claim C.1.1, we know that each set-post is assigned either zero or three agents. Next, we show that every element-agent v_i is assigned to an *acceptable* set-post.

Claim C.1.2. *For each element-agent v_i it holds that $\Pi(v_i) \in \{a_j \mid j \in [m]$ and $u_i \in C_j\}$.*

Proof. Suppose this is not true, and by TG let v_i denote an element-agent that is assigned to b_2; note that v_i does not find b_1 acceptable. Since there are $2m$ dummy agents q_z each with congestion $2m$ for b_2, at least one of them is *not* assigned to b_2. This agent envies v_i, a contradiction. (end of the proof of Claim C.1.2 $\diamond$)

Claim C.1.2 implies that J is a set cover, while Claim C.1.1 implies that $|J| \leq n$. Altogether we conclude that J is an exact cover. □

C.2 Proof of Theorem 3

Theorem 3 ($\star$). EF+TG *is FPT with respect to the number n of agents and the number m of posts, respectively.*

Proof. We first consider the parameter n. Let $I = (A, V, (\succeq_v)_{v \in V})$ be an instance of CONGESTED ASSIGNMENT. Due to TG, each agent is assigned to one of his first n tuples. Hence, for each agent $v_i \in V$, we guess (by brute-force searching) which of his first n tuples that v_i is "assigned" to, i.e., (a, d). After assigning all the agents, we check in linear time whether this results in a valid assignment, i.e., if we v_i "assign" (a, d), then there must be exactly d agents that are guessed to be "assigned" to (a, d). This check can be done in $O(n^2 + m)$ time. We abandon the current guess if it does not give a valid assignment; otherwise we proceed to check EF in $O(n^2)$ time.

Since there are n agents, each with n choices, the whole procedure can be done in $O(n^n \cdot (n^2 + m))$ time, which is an FPT time with respect to n.

Now, we consider the parameter m. Let $I = (A, V, (\succeq_v)_{v \in V})$ be an instance of CONGESTED ASSIGNMENT. We guess (by brute-force searching) the set of empty posts $A' \subseteq A$ in the sought solution. Then, we modify the preference list $\succeq_v$ of each agent v as follows: First, truncate $\succeq_v$ by removing all tuples (a, d) with ranks are higher than n; then, remove all tuples (a', d') with a' in A'. Denote the new preference list as $\succeq'_v$. Let $I' = (V, A \setminus A', (\succeq'_v)_{v \in V})$ denote the modified instance.

Since for assignments with all posts being non-empty, CP and EF are equivalent, we infer that I admits an EF and TG assignment where all posts in A' are empty and the rest is non-empty if and only if I' admits CP assignment where all posts are non-empty. The latter problem can be checked in polynomial time via lines 3–16 in Algorithm 1. The running time depends on the running time of the **while**-loop, which is $m(n + m)^2$ time. See the proof in Appendix B.6 for more details. Since there are 2^m subsets of empty posts to check, the overall running time is $2^m \cdot m \cdot (n + m)^2$, which is an FPT time with respect to m. □

Clique. The following graph problem is W[1]-complete with respect to the clique size h [20]. We will use it to show W[1]-hardness for finding a a maximally competitive assignment.

> CLIQUE
> **Input:** An undirected graph $G{=}(U, E)$, an integer $h \geq 0$.
> **Question:** Does G admit a *clique* of size h, i.e., a size-h subset $U' \subseteq U$ which induces a complete subgraph?

C.3 Proof of Theorem 4

Theorem 4 ($\star$). MAXCP+TG *is W[1]-hard and in XP with respect to the number t of unsatisfied agents. The W[1]-hardness holds even if there are no ties.*

Proof. We first show the W[1]-hardness by providing a parameterized reduction from the CLIQUE problem.

Let $I = (G = (U, E), h)$ denote an instance of CLIQUE with $U = \{u_1, \ldots, u_{|U|}\}$ and $E = \{e_1, \ldots, e_{|E|}\}$. We create a MAXCP+TG instance $I' = (A, V, (\succeq_v)_{v \in V}, t)$ as follows. Let $t = h + h(h-1)$. We will show that the agents corresponding to the vertices and edges of a size-h clique are the only unsatisfied agents.

– For each vertex $u_i \in U$, create a *vertex-post* a_i, a *vertex-agent* w_i, and $h - 1$ copies of w_i, denoted as $\tilde{w}_i^z$ with $z \in [h-1]$.

– For each edge $e_\ell \in E$ with $e_\ell = \{u_i, u_j\}$, create an *edge-post* b_ℓ and three *edge-agents* e^*_ℓ, e^i_ℓ, and e^j_ℓ.
– Create L dummy agents $x_1, \cdots, x_L$ with $L = |U|(h-2) + (|U| - h - 1) + (h(h-1) - 1) + (|E| - \binom{h}{2} - 1) + (t+1)$.
– Create five auxiliary posts $a_0, \tilde{a}_0, b_0, y, c_0$. Post y shall accommodate all L dummy agents, while c_0 is a "blocker" making sure that agents are assigned to the desired posts.

Let $V = W \cup \bigcup_{u_i \in U} \tilde{W}_i \cup \{e^*_\ell, e^i_\ell, e^j_\ell \mid e_\ell \in E,\ e_\ell = \{u_i, u_j\}\} \cup \{x_i \mid i \in [L]\}$, and $\mathcal{A} = \{a_i \mid u_i \in U\} \cup \{b_\ell \mid e_\ell \in E\} \cup \{a_0, \tilde{a}_0, b_0, y, c_0\}$, where $W = \{w_i \mid u_i \in U\}$ and $\tilde{W}_i = \{\tilde{w}^z_i \mid z \in [h]\}$. Let $n = |V|$.

Preferences. We state the preferences of the agents, restricted to the first n tuples. Here, $\langle \alpha, s, t\rangle = (\alpha, s) \succ (\alpha, s+1) \succ \cdots \succ (\alpha, t)$ depicts the preference list on tuples for post α and congestions ranging between s and t. Note that we also briefly explain the main purpose of these preferences in italicized text.

– The dummy agent x_i with $i \in [L]$ has the following preference list:

$$x_i\colon \langle a_1, 1, h-2\rangle \succ \langle a_2, 1, h-2\rangle \succ \cdots \succ \langle a_{|U|}, 1, h-2\rangle \succ \langle a_0, 1, |U| - h - 1\rangle \succ$$
$$\langle \tilde{a}_0, 1, h(h-1) - 1)\rangle \succ \langle b_0, 1, |E| - \binom{h}{2} - 1\rangle \succ \langle y, 1, L\rangle \succ \langle c_0, 1, n + (t+1) - 2L\rangle.$$

The dummy agents shall ensure some minimum number of agents assigned to each post (except b_ℓ and y): At least $h-1$, $|U| - h$, $h(h-1)$, and $|E| - \binom{h}{2}$ agents are to be assigned to $a_i (i \in [|U|])$, a_0, $\tilde{a}_0$, and b_0, respectively. The reason is that since at least $t+1$ dummy agents are to be assigned to y or c_0, they would envy the agents assigned to a post if its congestion is less than or equal to the maximum congestion of x_i to that post, which is not possible for a yes instance. Indeed, the dummy agents can only be assigned to y.

– The vertex-agent w_i with $i \in [|U|]$ has the following preference list:

$$w_i\colon \langle a_0, 1, |U| - h - 1\rangle \succ \langle a_i, 1, h-2\rangle \succ (a_0, |U| - h) \succ (a_i, h-1) \succ (a_i, h) \succ \langle c_0, 1, n - |U|\rangle.$$

We will show that exactly $|U| - h$ vertex-agents w_i are assigned to a_0. Consequently, there remain h vertex-agents v_i that are assigned to a_i. They shall correspond to the clique-vertices if G admit a size-h clique.

– For each $i \in [|U|]$, all copy-agents $\tilde{w}^z_i$ with $z \in [h-1]$ of the vertex-agent w_i have the same preference list:

$$\tilde{w}^z_i\colon \langle \tilde{a}_0, 1, h(h-1) - 1\rangle \succ \langle a_i, 1, h-1\rangle \succ (\tilde{a}_0, h(h-1)) \succ \langle c_0, 1, n - (h+1)(h-1)\rangle.$$

The copy-agents shall ensure that all $\tilde{w}^z_i$, $z \in [h-1]$, are jointly assigned to either $\tilde{a}_0$ or a_i. If they are assigned to a_i, then no other agent (including w_i) can be assigned to a_i. This corresponds to the case that the vertex u_i is not in the clique.

– The edge-agents e^*_ℓ, e^i_ℓ and e^j_ℓ with $e_\ell = \{u_i, u_j\}$ have the following preference lists:

$$e^*_\ell\colon \langle b_0, 1, |E| - \binom{h}{2} - 1\rangle \succ (b_\ell, 1) \succ (b_0, |E| - \binom{h}{2}) \succ (b_\ell, 2) \succ \langle c_0, 1, n - (|E| - \binom{h}{2}) - 2\rangle.$$
$$e^i_\ell\colon (b_\ell, 1) \succ (b_\ell, 2) \succ \langle a_i, 1, h\rangle \succ \langle c_0, 1, n - h - 2\rangle.$$
$$e^j_\ell\colon (b_\ell, 1) \succ (b_\ell, 2) \succ \langle a_j, 1, h\rangle \succ \langle c_0, 1, n - h - 2\rangle.$$

Note that e^*_ℓ can only be assigned to b_ℓ or b_0, and e^i_ℓ (resp. e^j_ℓ) only to a_i (resp. a_j) or b_ℓ. If e^*_ℓ is assigned to b_ℓ and does not envy other agents, then no other agent can be assigned to b_ℓ, as otherwise at least $|E| - \binom{h}{2} + 1$ agents must be assigned to b_0, which is impossible due to top-guarantees. Therefore, if e^*_ℓ is assigned to b_ℓ, then e^i_ℓ and e^j_ℓ have to be assigned to a_i and a_j, respectively. We will show that e^*_ℓ cannot be unsatisfied, and having e^i_ℓ and e^j_ℓ assigned to a_i and a_j, respectively, corresponds to having the edge e_ℓ in a size-h clique.

The maximum congestions of the agents are depicted in Table 1.

Correctness. Clearly, the construction can be done in polynomial time and no agent has ties in his preference list. It remains to show the correctness, i.e., I has a clique of size h if and only if I' admits a TG assignment with $t = h + h(h-1)$ agents being unsatisfied.

	a_1	$\dots$	$a_{\lvert U\rvert}$	a_0	$\tilde{a}_0$	b_ℓ	b_0	y	c_0
w_1	h	0	0	$\lvert U\rvert - h$	0	0	0	0	$n - \lvert U\rvert$
$\vdots$	0	h	0	$\lvert U\rvert - h$	0	0	0	0	$n - \lvert U\rvert$
$w_{\lvert U\rvert}$	0	0	h	$\lvert U\rvert - h$	0	0	0	0	$n - \lvert U\rvert$
$\tilde{w}_1^z$	$h-1$	0	0	0	$h(h-1)$	0	0	0	$n - h^2 + 1$
$\vdots$	0	$h-1$	0	0	$h(h-1)$	0	0	0	$n - h^2 + 1$
$\tilde{w}_{\lvert U\rvert}^z$	0	0	$h-1$	0	$h(h-1)$	0	0	0	$n - h^2 + 1$
e_ℓ^*	0	0	0	0	0	2	$\lvert E\rvert - \binom{h}{2}$	0	$n - (\lvert E\rvert - \binom{h}{2}) - 2$
e_ℓ^i	h	0	0	0	0	2	0	0	$n - h - 2$
e_ℓ^j	0	0	h	0	0	2	0	0	$n - h - 2$
x_z	$h-2$	$h-2$	$h-2$	$\lvert U\rvert - h - 1$	$h(h-1) - 1$	0	$\lvert E\rvert - \binom{h}{2} - 1$	L	$n + (t+1) - 2L$

Table 1: Maximum congestions of the agents constructed for Theorem 4. For an illustration, we assume that $e_\ell = \{u_1, u_{|U|}\}$.

The "only if" part. Let $\mathcal{C} \subseteq U$ denote an h-clique for I. Let $E^{\mathcal{C}} \subseteq E$ denote the edge set associated with $\mathcal{C}$, i.e., $E^{\mathcal{C}} = \{e_\ell = \{u_i, u_j\} \mid u_i, u_j \in \mathcal{C}\}$. Then, we claim that the following assignment Π is a TG assignment with t unsatisfied agents.

- For each $u_i \in \mathcal{C}$, assign w_i to a_i, and assign $\tilde{w}_i^z$ with $z \in [h-1]$ to $\tilde{a}_0$.
- For each $u_i \notin \mathcal{C}$, assign $\tilde{w}_i^z$ with $z \in [h-1]$ to a_i, and assign w_i to a_0.
- For each $e_\ell = \{u_i, u_j\} \in E^{\mathcal{C}}$, assign e_ℓ^i to a_i, e_ℓ^j to a_j, and e_ℓ^* to b_ℓ.
- For each $e_\ell = \{u_i, u_j\} \notin E^{\mathcal{C}}$, assign e_ℓ^i and e_ℓ^j to b_ℓ, and e_ℓ^* to b_0.
- Assign x_z to y with $z \in [L]$.

Clearly, Π is TG with the following congestion vector.

Observation 3. *Π is TG and satisfies the following.*

(i) $|\Pi(a_0)| = |U| - h$, $|\Pi(\tilde{a}_0)| = h(h-1)$, $|\Pi(b_0)| = |E| - \binom{h}{2}$, *and* $|\Pi(y)| = L$.
(ii) For each $u_i \in \mathcal{C}$, it holds that $\Pi(a_i) = \{w_i, \tilde{w}_i^z \mid z \in [h-1]\}$.
For each $u_i \in U \setminus \mathcal{C}$, it holds that $|\Pi(a_i)| = \{\tilde{w}_i^z \mid z \in [h-1]\}$.
(iii) For each $e_\ell \in E$, if $e_\ell \in E^{\mathcal{C}}$, then $|\Pi(b_\ell)| = 2$*; otherwise* $|\Pi(b_\ell)| = 1$.

Let $V' = \{w_i \mid u_i \in \mathcal{C}\} \cup \{e_\ell^i, e_\ell^j \mid e_\ell \in E^{\mathcal{C}} \text{ with } e_\ell = \{u_i, u_j\}\}$. Note that $|V'| = t$. We aim to show that all agents except those from V' are satisfied. By the above observation, it is straightforward that every dummy agent x_z is satisfied, every agent that *does not* correspond to the clique vertices is satisfied, and the copies $\tilde{w}_i^z$ of all vertex-agents are also satisfied. It remains to consider the edge-agents that are not in V'. Let $e_\ell \in E$ with $e_\ell = \{u_i, u_j\}$. Clearly, if $e_\ell \notin E^{\mathcal{C}}$, then the two edge-agents e_ℓ^i and e_ℓ^j are satisfied since they are assigned to their most preferred post. Agent e_ℓ^* with $e_\ell \notin E^{\mathcal{C}}$ is also satisfied since he is assigned to b_0 with congestion $|E| - \binom{h}{2}$ which is better than $(b_\ell, 2)$. If $e_\ell \in E^{\mathcal{C}}$, then agent e_ℓ^* is satisfied since he is assigned to b_ℓ alone which is better than $(b_0, |E| - \binom{h}{2})$. Hence, only the agents in V' are unsatisfied. Since $|V'| = t$, this concludes the proof for the "only if" direction.

The "if" part. Let Π be a TG assignment with at most t unsatisfied agents. We aim to show that the following vertex subset $\mathcal{C}$ is a size-h clique: $\mathcal{C} = \{u_i \mid |\Pi(a_i)| \geq h\}$. Before we show this, we observe the following regarding the congestions and assignments of the posts.

Claim C.3.1. *(1)* $|\Pi(a_0)| = |U| - h$, $|\Pi(\tilde{a}_0)| = h(h-1)$, *and* $|\Pi(b_0)| = |E| - \binom{h}{2}$.
(2) For each $u_i \in U$, it holds that $|\Pi(a_i)| \in \{h-1, h\}$.
(3) For each $e_\ell \in E$, it holds that $|\Pi(b_\ell)| \leq 2$.
(4) $\Pi(a_0) \subseteq \{w_i \mid u_i \in U\}$, $\Pi(\tilde{a}_0) \subseteq \{\tilde{w}_i^z \mid i \in [|U|], z \in [h-1]\}$, *and* $\Pi(b_0) \subseteq \{e_\ell^* \mid e_\ell \in E\}$.
(5) All edge-agents $E^* = \{e_\ell^* \mid e_\ell \in E\}$ are satisfied.

Proof. We show the first two statements together by considering the dummy agents.

Since Π is TG and the maximum congestion of dummy x_z for $a_0, \tilde{a}_0, b_0$, and a_i with $i \in [|U|]$ are $|U|-h-1$, $h(h-1)-1$, $|E|-\binom{h}{2}-1$, and $h-2$, respectively, we infer by simple calculation that there

are more than t dummy agents who are assigned to y or c_0. Since Π does not have more than t unsatisfied agents, this further implies that there is at least one satisfied dummy agent x_z who is assigned to y or c_0. By his preferences, every tuple that he prefers to $(y, t+1)$ must have congestion that exceeds his maximum durable congestion. This implies that $|\Pi(a_0)| \geq |U| - h$, $|\Pi(\tilde{a}_0)| \geq h(h-1)$, and $|\Pi(b_0)| \geq |E| - \binom{h}{2}$, for $|\Pi(a_i)| \geq h-1$. Since no agent allows more than the aforementioned congestions (except for a_i), we further infer that $|\Pi(a_0)| = |U| - h$, $|\Pi(\tilde{a}_0)| = h(h-1)$, and $|\Pi(b_0)| = |E| - \binom{h}{2}$. For a_i, since the maximum congestion of any agent for a_i is h, we infer that $h-1 \leq |\Pi(a_i)| \leq h$.

Statement (3) is straightforward by observing the maximum congestion of any agent towards b_ℓ is two.

The first part of statement (4) follows from the fact that the only agents that have $(a_0, |U| - h)$ in their top n choices are the vertex-agents. Similarly, we can show that the other parts of statement (4) are also correct.

To show statement (5), let us analyze which agents are unsatisfied. To this end, define $W' = \{w_i \in W \mid w_i \notin \Pi(a_0)\}$. By statements (1) and (4), we infer that $|W'| = h$.

Further, every agent w_i in W' is unsatisfied since by statement (2) that $|\Pi(a_i)| \geq h-1$, any agent not assigned to a_0 will envy those that are assigned to a_0. This implies that at most $2\binom{h}{2}$ agents other than W' can be unsatisfied.

By statement (2), partition the posts $\{a_i \mid u_i \in U\}$ into A_1 and A_2 with $A_1 = \{a_i \mid u_i \in U \wedge |\Pi(a_i)| = h-1\}$ and $A_2 = \{a_i \mid u_i \in U \wedge |\Pi(a_i)| = h\}$. Note that by the top-guarantees, every post a_i from A_2 can only be assigned vertex-agents w_i or edge-agent e^i_ℓ for some edge $e_\ell \in E$ with $u_i \in e_\ell$. However, this implies that every agent assigned to post $a_i \in A_2$ is unsatisfied since w_i prefers $(a_0, |U| - h)$ to $(a_i, h-1)$ and every edge-agent e^i_ℓ (with $u_i \in e_\ell$) prefers $(b_\ell, 2)$ to (a_i, h); recall by statements (1) and (3) that $|\Pi(a_0)| = |U| - h$ and $|\Pi(b_\ell)| \leq 2$. This further implies that $|A_2| \leq h$ since $t = h + h(h-1) = h^2$ can be unsatisfied.

By statement (1), we have that $|\Pi(\tilde{a}_0)| = h(h-1)$. Since every vertex-agent has $h-1$ copies, there are at least h vertices each of which has a copy-agent assigned to $\tilde{a}_0$. Since every copy-agent e^z_ℓ corresponding to vertex u_i prefers $(a_i, h-1)$ to $(\tilde{a}_0, h(h-1))$, it follows that at least $h - |A_2|$ copy-agents will be unsatisfied, namely those whose corresponding vertex-post has congestion $h-1$.

Since at most h vertex-agents and at most $(|U| - h)(h-1)$ copy-agents can be assigned to any vertex-post a_i, the number of edge-agents e^i_ℓ that have to be assigned to some vertex-post a_i is at least

$$|A_1| \cdot (h-1) + |A_2| \cdot h - \big(h + (|U| - h)(h-1)\big) = h(h-1) - (h - |A_2|).$$

Observe that each edge-agent e^i_ℓ that is assigned to some vertex-post a_i is unsatisfied. This implies that at least $h(h-1) - (h - |A_2|)$ edge-agents are unsatisfied. Together with the $h - |A_2|$ unsatisfied copy-agents, no more other agent can be unsatisfied. In other words, every edge-agent e^*_ℓ must be satisfied, as desired. (end of the proof of Claim C.3.1 $\diamond$)

Now, we are ready to show that $\mathcal{C}$ is a clique of size h. We first show that $\mathcal{C}$ has size h. Define $W' = \{w_i \mid \Pi(w_i) \neq a_0\}$. By the preferences of the vertex-agents and by Claim C.3.1(1), $|W'| = h$ and every vertex-agent in W' is unsatisfied. By Claim C.3.1(1) and by the maximum congestions of the agents towards b_0, we infer that $\Pi(b_0)$ consists of exactly $\binom{|E| - \{h}{2\}}$ edge-agents e^*_ℓ, $\ell \in [|E|]$. By Claim C.3.1(5), every remaining edge-agent e^*_ℓ that is *not* assigned to b_0 must be assigned to the corresponding edge-post b_ℓ alone. This implies that the remaining two edge-agents e^i_ℓ and e^j_ℓ with $e_\ell = \{u_i, u_j\}$ are not assigned to b_ℓ and hence unsatisfied; they both envy e^*_ℓ. Define $E' = \{e^i_\ell, e^j_\ell \mid \Pi(e^*_\ell) = b_\ell\}$. Then, $|E'| = h(h-1)$ and it yields $h(h-1)$ unsatisfied edge-agents by Claim C.3.1(1). Together with the h unsatisfied vertex-agents in W', we infer that every copy-agent $\tilde{w}^z_i$ is satisfied. In particular, it means that for each copy-agent $\tilde{w}^z_i$ that is assigned to $\tilde{a}_0$ it must hold that $|\Pi(a_i)| = h$. Recall that there are at least h vertices u_i each of which has a copy-agent assigned to $\tilde{a}_0$. This further implies that there are at least h vertex-posts that each have congestion h, that is $|\mathcal{C}| = h$.

It remains to show that $\mathcal{C}$ is a clique. Let $E'' = \{e_\ell \mid e^i_\ell \in \Pi(a_i) \text{ for some } u_i \in \mathcal{C}\}$ be the set consisting of all edges whose corresponding edge-agents are assigned to some vertex-post. Clearly, $|E''| \geq \binom{h}{2}$ since $\mathcal{C} = h$; note that the equality holds only if $\mathcal{C}$ induces a clique. Towards a contradiction, suppose that $\mathcal{C}$ contains two vertices u_i and u_j that are not adjacent with each other. This implies that $|E''| > \binom{h}{2}$. Let us consider each edge $e_\ell \in E''$ and let $e_\ell = \{u_i, u_j\}$ with $i < j$.

By definition, at least one of the two edge-agents e_ℓ^i and e_ℓ^j is assigned to a_i and he is unsatisfied. We claim that both edge-agents are unsatisfied. Without loss of generality, assume that $\Pi(e_\ell^i) = a_i$ and is unsatisfied. If e_ℓ^j is assigned to vertex-post a_j or c_0, then he is unsatisfied as well. Otherwise, e_ℓ^j is assigned to post b_ℓ alone, making e_ℓ^* unsatisfied which is not possible according to Claim C.3.1(5). Hence, both e_ℓ^i and e_ℓ^j are satisfied. Since there can be only $h(h-1)$ unsatisfied edge-agents, we conclude that $|E''| = \binom{h}{2}$, as desired.

Now, we turn to the XP result. The ideas is to guess the unsatisfied agents and the posts that they are assigned to, and replace them with dummies and run Algorithm 1 for the reduced instance. More precisely, we guess who are the unsatisfied agents in $O(n^t)$ time; denoting the set of unsatisfied agents as $V^* = \{v_1^*, \dots, v_t^*\}$. For each V^*, we further guess which posts they are assigned to in $O(m^t)$ time, denoted as $A^* = \{a_1^*, \dots, a_t^*\}$ with a_z^* being the post that v_z^* will assigned to and $z \in [t]$.

Then we create t dummy agents $P = \{p_z \mid z \in [t]\}$ and set their preference list as $p_z \colon (a_z^*, 1) \succ \cdots \succ (a_z^*, n) \succ \cdots$. We replace the agents V^* with the dummies and use Algorithm 1 to solve the resulting instance. If Algorithm 1 returns no on the current guess, we proceed with the next guess; otherwise, let Π be CP assignment returned by Algorithm 1. It is straightforward that replacing each dummy P_z with v_z^* in the assignment yields a TG assignment with at most t unsatisfied agents.

The overall running time is $O(n^t m^t)$. □

C.4 Proof of Theorem 5

Theorem 5 (⋆)**.** MAXCP+TG *is FPT with respect to n, and in XP with respect to m, where n and m denote the number of agents and the number of posts, respectively.*

Proof sketch. **Parameter n:** We first guess a subset V' of unsatisfied agents. Afterwards, similarly to Theorem 3, we guess for each satisfied agent $V \setminus V'$ one of his first n tuples and check whether the $|V \setminus V'|$ guesses yield a valid assignment $\Pi_{V'}$ and store the number of unsatisfied agents. Finally, we select one valid $\Pi_{V'}$ with fewest unsatisfied agents. The whole approach can be done in FPT time wrt. n.

Parameter m: For the XP-algorithm, we guess the congestion vector $\vec{s}$ with $\vec{s}[j] \in \{0, \dots, n\}$ and $\Sigma\vec{s} = n$ and use the algorithm behind Lemma 2 to determine the minimum number of unsatisfied agents. The overall running time is $n^m \cdot (m+n)^{O(1)}$, which is XP wrt. m. □

C.5 Proof of Theorem 6

Theorem 6 (⋆)**.** *Deciding whether an instance of* CONGESTED ASSIGNMENT *has an assignment with at most t unsatisfied agents is W[1]-hard with respect to t.*

Proof. We reduce from the W[1]-complete problem CLIQUE; the definition can be found ahead of Appendix C.3.

Let $I = (G = (U, E), h)$ denote an instance of CLIQUE with $U = \{u_1, \dots, u_{\hat{n}}\}$ and $E = \{e_1, \dots, e_{\hat{m}}\}$ being the vertex set and edge set, respectively. Without loss of generality, we assume that $\hat{n} > 3h + \binom{h}{2}$ and $\hat{m} > 2h + 2\binom{h}{2}$ as the problem remains W[1]-hard in this case.

The idea is to construct an instance $I' = (\mathcal{A}, V, (\succeq_v)_{v \in V})$ of CONGESTED ASSIGNMENT such that the unsatisfied agents correspond to the vertices and edges of a size-h clique. We set the number of unsatisfied agents to $t = 2h + \binom{h}{2}$, and let L and R be two very large numbers such that $L > 2t$ and $R > (L+2) \cdot (\hat{n} + \hat{m}) + h$. For the sake of brevity, let $N = (L+2) \cdot (\hat{n} + \hat{m}) + 2R$, and we will create exactly N agents.

Posts and agents.

- For each vertex $u_i \in U$, create one *vertex-post* a_i and $L+2$ *vertex-agents* w_i, p_i, p_i^z, $z \in [L]$.
- For each edge $e_\ell \in E$, create one *edge-post* b_ℓ and $L+2$ *edge-agents* e_ℓ, f_ℓ^z, $z \in [L+1]$.
- Create $2R$ dummy agents x_z, y_z, $z \in [R]$.
- Create 3 auxiliary posts a_0, b_0, and c_0.

Let $A = \{a_i \mid i \in [\hat{n}]\}$, $B = \{b_\ell \mid \ell \in [\hat{m}]\}$, $W = \{w_i \mid i \in [\hat{n}]\}$, $P = \{p_i \mid i \in [\hat{n}]\}$, $P_i = \{p_i^z \mid z \in [L], i \in [\hat{n}]\}$, $F_\ell = \{f_\ell^z \mid z \in [L+1]\}$, $X = \{x_z \mid z \in [R]\}$, and $Y = \{y_z \mid z \in [R]\}$. Then, we set $\mathcal{A} = A \cup B \cup \{a_0, b_0, c_0\}$, and $V = W \cup P \cup \bigcup_{i \in [\hat{n}]} P_i \cup E \cup \bigcup_{\ell \in [\hat{m}]} F_\ell \cup X \cup Y$. In total, we have created $\hat{n} + \hat{m} + 3$ posts and N agents.

Preferences. For two numbers $s, t \in [N]$ and post $\alpha \in \mathcal{A}$, let $\langle \alpha, s, t \rangle = (\alpha, s) \succ (\alpha, s+1) \succ \cdots \succ (\alpha, t)$ depict the preference list on tuples for post α and congestions ranging between s and t. The notation "$***$" refers to an arbitrary but congestion-averse preferences of the tuples that are not explicitly mentioned.

(i) For each vertex $u_i \in U$, the vertex-agents $w_i \in W$, $p_i \in P$, and $p_i^z \in P_i$, $z \in [L]$, have the following preference lists:

$$w_i\colon \langle a_0, 1, \hat{n} - h \rangle \succ \langle a_i, 1, L+2 \rangle \succ \langle c_0, 1, N \rangle \succ *** \succ \langle b_0, 1, N \rangle \succ \langle a_0, \hat{n} - h + 1, N \rangle,$$
$$p_i\colon \langle a_i, 1, L+1 \rangle \succ \langle a_0, 1, \hat{n} - h \rangle \succ (a_i, L+2) \succ \langle c_0, 1, N \rangle \succ *** \succ \langle b_0, 1, N \rangle \succ \langle a_0, \hat{n} - h + 1, N \rangle,$$
$$p_i^z\colon \langle a_i, 1, L+2 \rangle \succ \langle c_0, 1, N \rangle \succ *** \succ \langle b_0, 1, N \rangle \succ \langle a_0, 1, N \rangle.$$

(ii) For each edge $e_\ell \in E$, the edge-agents e_ℓ, f_ℓ^z, $z \in [L+1]$, have the following preference lists, where we assume $e_\ell = \{u_i, u_j\}$:

$$e_\ell\colon \langle b_0, 1, \hat{m} - \binom{h}{2} \rangle \succ \langle b_\ell, 1, L+2 \rangle \succ \langle c_0, 1, N \rangle \succ *** \succ \langle b_0, \hat{m} - \binom{h}{2} + 1, N \rangle \succ \langle a_0, 1, N \rangle.$$
$$f_\ell^z\colon \langle b_\ell, 1, L+1 \rangle \succ \langle a_i, 1, L+1 \rangle \succ \langle a_j, 1, L+1 \rangle \succ (b_\ell, L+2) \succ \langle c_0, 1, N \rangle \succ *** \succ$$
$$\langle b_0, 1, N \rangle \succ \langle a_0, 1, N \rangle.$$

(iii) The preference lists of the dummy agent $x_z \in X$ and $y_z \in Y$ are as follows:

$$x_z\colon \langle a_0, 1, \hat{n} - h - 1 \rangle \succ \langle a_1, 1, L \rangle \succ \cdots \succ \langle a_{\hat{n}}, 1, L \rangle \succ \langle c_0, 1, 2R \rangle \succ \langle a_1, L+1, N \rangle \succ \cdots \succ$$
$$\langle a_{\hat{n}}, L+1, N \rangle \succ \langle c_0, 2R+1, N \rangle \succ *** \succ \langle b_0, 1, N \rangle \succ \langle a_0, \hat{n} - h, N \rangle \succ \langle a_0, 1, N \rangle.$$
$$y_z\colon \langle b_0, 1, \hat{m} - \binom{h}{2} - 1 \rangle \succ \langle b_1, 1, L \rangle \succ \cdots \succ \langle b_{\hat{m}}, 1, L \rangle \succ \langle c_0, 1, N \rangle \succ *** \succ$$
$$\langle b_0, \hat{m} - \binom{h}{2}, N \rangle \succ \langle a_0, 1, N \rangle.$$

This completes the construction of the instance I', which can clearly be done in polynomial time. Note that it is also a parameterized reduction since the parameter $t = 2h + \binom{h}{2}$ is a polynomial function in h. It remains to show the correctness, i.e., I has a size-h clique if and only if I' has an assignment with at most t unsatisfied agents.

For the **"only if"** part, let U' be a clique of size h. We construct the following assignment Π and show that it has at most t unsatisfied agents.

(1) For each vertex $u_i \in U$, assign all agents from $P_i \cup \{p_i\}$ to a_i. Additionally assign w_i to a_i if $u_i \in U'$; otherwise assign w_i to a_0.
(2) For each edge $e_\ell \in E$, assign all agents from F_ℓ to b_ℓ. Additionally assign e_ℓ to b_ℓ if $e_\ell \subseteq U'$, i.e., both its endpoints are in U'; otherwise assign e_ℓ to b_0.
(3) Assign all agents from $X \cup Y$ to c_0.

To see who is unsatisfied, let $W' = \{w_i \in W \mid w_i \in \Pi(a_i)\}$, $P' = \{p_i \in P \mid p_i \in \Pi(a_i)\}$, and $E' = \{e_\ell \in E \mid e_\ell \in \Pi(b_\ell)\}$. We claim that all agents but those from $W' \cup P' \cup E'$ are satisfied.

In the following, we say that a post α is the *most preferred post* for agent q if every tuple that is contains a post other than α is less preferred than $(\alpha, |\Pi(\alpha)|)$. Further, a tuple (α, d) is a *most preferred feasbile* tuple for agent q if every tuple (α', d') that is preferred to (α, d) has congestion $|\Pi(\alpha')| > d'$.

Clearly, every agent in $X \cup Y$ is satisfied since $(c_0, 2R)$ is his most preferred *feasible* tuple. Every agent in $(\bigcup_{i \in [\hat{n}]} P_i) \cup (W \setminus W') \cup (P \setminus P') \cup (E \setminus E')$ is satisfied since he is assigned to his most preferred post. Every agent $f_\ell^z \in F_\ell$ is also satisfied since either he is assigned to his most preferred

post (if e_ℓ is not a "clique" edge) or $|\Pi(a_i)| = |\Pi(a_j)| = L+2$ so $(b_\ell, L+2)$ remains his most preferred feasible tuple. This concludes the proof for the "only if" direction.

For the **"if"** direction, let Π denote an assignment with at most t unsatisfied agents. Before we construct a clique, let us analyze the preferences and Π would look like.

Claim C.5.1. Π *satisfies the following.*

(1) $|\Pi(a_0)| = \hat{n} - h$ *and* $|\Pi(b_0)| = \hat{m} - \binom{h}{2}$.
(2) Every agent from W *that is* not *assigned to* a_0 *is unsatisfied and every agent from* E *that is* not *assigned to* b_0 *is unsatisfied.*
(3) For each $a_i \in A$ *we have that* $|\Pi(a_i)| \geq L+1$ *and for each* $b_\ell \in B$ *we have that* $|\Pi(b_\ell)| \geq L+1$.
(4) It holds that $|\Pi(c_0)| \leq 2R$.
(5) For each $a_i \in A$ *we have that* $|\Pi(a_i)| \leq L+2$ *and for each* $b_\ell \in B$ *we have that* $|\Pi(b_\ell)| \leq L+2$.

Proof. Statement (1)*:* The lower bounds are straightforward since all agents from $W \cup X$ prefer $(a_0, \hat{n} - h - 1)$ to any other tuple that does not contain a_0: If $|\Pi(a_0)| < \hat{n} - h$ would hold, then more than $|W \cup X| - (\hat{n} - h) > R$ agents will be unsatisfied, which is not possible since $R > (L+2) \cdot (\hat{n} + \hat{m}) > t$. Similar reasoning shows that $|\Pi(b_0)| \geq \hat{m} - \binom{h}{2}$ by considering the preferences of $E \cup Y$. Now, we show the upper bounds. Suppose, for the sake of contradiction, that $|\Pi(a_0)| > \hat{n} - h$. Then, since no agent considers $(a_0, \hat{n} - h + 1)$ more valuable than any other tuple that does not contain a_0, all agents assigned to a_0 are unsatisfied. Since we can assume that $\hat{n} > 3h + \binom{h}{2}$, it follows that more than t agents will be unsatisfied, a contradiction. Similarly, since we have just shown that $|\Pi(a_0)| \leq \hat{n} - h$, from the remaining possible tuples, no agent considers $(b_0, \hat{m} - \binom{h}{2})$ more valuable than any tuple that does not contain b_0 (except $(a_0, \hat{n} - h + z)$, $z \geq 1$, which is excluded). Consequently, by the fact that $\hat{m} > 2h + 2\binom{h}{2}$, we infer that $|\Pi(b_0)| \leq \hat{m} - \binom{h}{2}$ as otherwise all agents assigned to b_0 are unsatisfied the number of which exceeds t.

Statement (2)*:* This statement follows directly from the previous statement and from the preferences of the agents in $W \cup E$.

Statement (3): We show the lower bound by iterating through all $i \in [\hat{n}]$. By Statement (1), every agent in $X \cup P_1 \cup \{p_1\}$ prefers (a_1, L) to every other tuple that does not contain a_1 (excluding a_0). Hence, $|\Pi(a_1)| \geq L+1$ as otherwise more than $R - L > t$ agents from X are not assigned to a_1 and will be unsatisfied. By applying the above reasoning for the next $i \geq 2$, we infer that $|\Pi(a_i)| \geq L+1$ holds for all $i \in [\hat{n}]$. Similarly, we infer that $|\Pi(b_\ell)| \geq L+1$ holds for every $\ell \in [\hat{m}]$.

Statement (4)*:* Suppose this is not true, i.e., $|\Pi(c_0)| \geq 2R+1$. Then, by Statements (1)–(2) and by the bound $t = 2h + \binom{h}{2}$, at most h agents from $V \setminus (W \cup E)$ can be unsatisfied. By construction, every agent in X that is assigned to c_0 will be unsatisfied since he prefers $(a_1, L+1)$ to $(c_0, 2R+1)$. Hence, at most h agents from X can be assigned to c_0. This means that at least $2R + 1 - h$ agents from $W \cup P \cup \bigcup_{i \in [\hat{n}]} P_i \cup E \cup \bigcup_{\ell \in [\hat{m}]} F_\ell \cup Y$ need to be assigned to c_0. This is not possible however since $R > (L+2) \cdot (\hat{n} + \hat{m}) + h$ and $|Y| = R$.

Statement (5)*:* Let $i \in [\hat{n}]$. The statement follows directly from the fact that every agent prefers $(c_0, 2R)$ to $(a_i, L+3)$ and $|\Pi(c_0)| \leq 2R$ (see Claim C.5.1(1)): $|\Pi(a_i)| > L+2$ would hold, then all agents assigned to a_i are unsatisfied, the number of which exceed t since $L > 2t$. (end of the proof of Claim C.5.1 $\diamond$)

The next statement is about the structure of the agents assigned to $A \cup B$.

Claim C.5.2. *Let* $A' = \{a_i \in A \colon |\Pi(a_i)| = L+2\}$ *and* $B' = \{b_\ell \in B \colon |\Pi(b_\ell)| = L+2\}$. *Then,* Π *satisfies the following.*

(1) $|A'| + |B'| \geq h + \binom{h}{2}$.
(2) For each post $a_i \in A'$*, at least two agents in* $\Pi(a_i)$ *are unsatisfied; for each post* $b_\ell \in B'$*, at least one agent in* $\Pi(b_\ell)$ *is unsatisfied.*
(3) $|A'| \leq h$ *and* $|B'| \geq \binom{h}{2}$.
(4) For each post $b_\ell \in B'$ *it holds that* $|\Pi(a_i)| = |\Pi(a_j)| = L+2$ *where* $e_\ell = \{u_i, u_j\}$.

Proof. Statement (1)*:* This can be shown by simple calculation. By Claim C.5.1(1) and (4), at least $(L+2)\cdot(\hat{n}+\hat{m})-(\hat{n}-h)-(\hat{m}-\binom{h}{2})=(L+1)\cdot(\hat{n}+\hat{m})+h+\binom{h}{2}$ agents are assigned to the posts of $A\cup B$. By Claim C.5.1(3) and (5), each post in $A\cup B$ is assigned either $L+1$ or $L+2$ agents. That is, at least $h+\binom{h}{2}$ of the post in $A\cup B$ are each assigned $L+2$ agents, confirming that $|A'|+|B'|\geq h+\binom{h}{2}$.

Statement (2)*:* For each post $a_i\in A'$, since $|\Pi(a_i)|=L+2$ and $|P_i|=L$, at least two agents in $\Pi(a_i)$ are not from P_i, i.e., $|\Pi(a_i)\setminus P_i|\geq 2$ We claim that the agents in $\Pi(a_i)\setminus P_i$ are unsatisfied by considering the preferences of all agents except P_i: Every agent from $W\cup P$ prefers $(a_0,\hat{n}-h)$ to $(a_i,L+2)$. Every agent from E prefers $(b_0,\hat{m}-\binom{h}{2})$ to $(a_i,L+2)$. Every agent from $(\bigcup_{i'\in[\hat{n}]\setminus\{i\}}P_i)\cup(\bigcup_{\ell\in[\hat{m}]}F_\ell)\cup X\cup Y$ prefers $(c_0,2R)$ to $(a_i,L+2)$. Since $|\Pi(a_0)|=\hat{n}-h$, $|\Pi(b_0)|=\hat{m}-\binom{h}{2}$, and $|\Pi(c_0)|\leq 2R$ (see Claim C.5.1s(1) and (4)), we infer that every agent in $\Pi(a_i)\setminus P_i$ is unsatisfied.

Similarly, for each post $b_\ell\in B'$, since $|\Pi(b_\ell)|=L+2$ and $|F_\ell|=L+1$, at least one agent in $\Pi(b_\ell)$ is not from F_ℓ. We claim that the agents in $\Pi(b_\ell)\setminus F_\ell$ are unsatisfied by considering the preferences of all agents except P_i: Every agent from $W\cup P\cup(\bigcup_{i\in[\hat{n}]}P_i)\cup(\bigcup_{\ell'\in[\hat{m}]\setminus\{\ell\}}F_\ell)\cup X\cup Y$ prefers $(c_0,2R)$ to $(b_\ell,L+2)$. Every agent from E prefers $(b_0,\hat{m}-\binom{h}{2})$ to $(b_\ell,L+2)$. Again, since $|\Pi(b_0)|=\hat{m}-\binom{h}{2}$ and $|\Pi(c_0)|\leq 2R$ (see Claim C.5.1(1) and (4)), we infer that every agent in $\Pi(b_\ell)\setminus F_\ell$ is unsatisfied.

Statement (3)*:* Statement (2) implies that at least $2|A'|+|B'|$ agents are unsatisfied. By the upper bound that $t\leq 2h+\binom{h}{2}$ and by Statement (1), we infer that $|A'|\leq h$, and hence $|B'|\geq\binom{h}{2}$.

Statement (4)*:* Suppose, towards a contradiction, that $|\Pi(a_i)|\neq L+2$. Then, by Claim C.5.1(3) and (5), it follows that $|\Pi(a_i)|=L+1$. We claim that every agent assigned to b_ℓ is unsatisfied. Let us consider an arbitrary agent $q\in\Pi(b_\ell)$. Clearly, if $q\in W\cup P\cup\bigcup_{\ell\in[\hat{m}]}P_\ell\cup X\cup Y\cup E\cup F\setminus(\{e_\ell\}\cup F_\ell)$, then he is unsatisfied since he prefers $(c_0,2R)$ to $(b_\ell,L+2)$. If $q=e_\ell$, then he is unsatisfied since he prefers $(b_0,\hat{m}-\binom{h}{2})$ to $(b_\ell,L+2)$, while if $q\in F_\ell$, then he is unsatisfied since he prefers $(a_i,L+1)$ to $(b_\ell,L+2)$ as well. This concludes the proof that every agent in $\Pi(b_\ell)$ is unsatisfied, implying that more than $L>2t$ agents is unsatisfied, a contradiction.

By an analogous reasoning, we can show that $|\Pi(a_j)|=L+2$. (end of the proof of Claim C.5.2 $\diamond$)

Now, we are ready to show the existence of a size-h clique. By Claim C.5.2(3), B' corresponds to at least $\binom{h}{2}$ edges. Hence, there are at least h vertices incident to any edge corresponding to B'. For each vertex a_i that is "incident" to any edge-post in B', we know by Claim C.5.2(4) that its corresponding vertex-post a_i must be assigned $L+2$ posts. By Claim C.5.2(3), there are at most h such vertex-posts. Hence, there are exactly h vertex-posts that are each assigned $L+2$ agents, and this is possible if and only if they form a size-h clique. □