Lower Bounds for Dynamic Distributed Task Allocation*

Hsin-Hao Su
Boston College

Nicole Wein†
MIT

Abstract

We study the problem of distributed task allocation in multi-agent systems. Suppose there is a collection of agents, a collection of tasks, and a *demand vector*, which specifies the number of agents required to perform each task. The goal of the agents is to cooperatively allocate themselves to the tasks to satisfy the demand vector. We study the *dynamic* version of the problem where the demand vector changes over time. Here, the goal is to minimize the *switching cost*, which is the number of agents that change tasks in response to a change in the demand vector. The switching cost is an important metric since changing tasks may incur significant overhead.

We study a mathematical formalization of the above problem introduced by Su, Su, Dornhaus, and Lynch [22], which can be reformulated as a question of finding a low distortion embedding from symmetric difference to Hamming distance. In this model it is trivial to prove that the switching cost is at least 2. We present the first non-trivial lower bounds for the switching cost, by giving lower bounds of 3 and 4 for different ranges of the parameters.

*A preliminary version of this paper appeared in ICALP 2020.
†supported by an NSF Graduate Fellowship and NSF Grant CCF-1514339

1 Introduction

Task allocation in multi-agent systems is a fundamental problem in distributed computing. Given a collection of tasks, a collection of task-performing agents, and a *demand vector* which specifies the number of agents required to perform each task, the agents must collectively allocate themselves to the tasks to satisfy the demand vector. This problem has been studied in a wide variety of settings. For example, agents may be identical or have differing abilities, agents may or may not be permitted to communicate with each other, agents may have limited memory or computational power, agents may be faulty, and agents may or may not have full information about the demand vector. See Georgiou and Shvartsman's book [9] for a survey of the distributed task allocation literature. See also the more recent line of work by Dornhaus, Lynch and others on algorithms for task allocation in ant colonies [5, 22, 6, 19].

We consider the setting where the demand vector *changes dynamically* over time and agents must redistribute themselves among the tasks accordingly. We aim to minimize the *switching cost*, which is the number of agents that change tasks in response to a change in the demand vector. The switching cost is an important metric since changing tasks may incur significant overhead. Dynamic task allocation has been extensively studied in practical, heuristic, and experimental domains. For example, in swarm robotics, there is much experimental work on heuristics for dynamic task allocation (see e.g. [12, 21, 15, 16, 13, 14]). Additionally, in insect biology it has been empirically observed that demands for tasks in ant colonies change over time based on environmental factors such as climate, season, food availability, and predation pressure [17]. Accordingly, there is a large body of biological work on developing hypotheses about how insects collectively perform task allocation in response to a changing environment (see surveys [1, 20]).

Despite the rich experimental literature, to the best of our knowledge there are only two works on dynamic distributed task allocation from a theoretical algorithmic perspective. Su, Su, Dornhaus, and Lynch [22] present and analyze gossip-based algorithms for dynamic task allocation in ant colonies. Radeva, Dornhaus, Lynch, Nagpal, and Su [19] analyze dynamic task allocation in ant colonies when the ants behave randomly and have limited information about the demand vector.

1.1 Problem Statement

We study the formalization of dynamic distributed task allocation introduced by Su, Su, Dornhaus, and Lynch [22].

Objective: Our goal is to minimize the *switching cost*, which is the number of agents that change tasks in response to a change in the demand vector.

Properties of agents:

1. the agents have complete information about the changing demand vector
2. the agents are heterogeneous
3. the agents cannot communicate
4. the agents are memoryless

The first two properties specify *capabilities* of the agents while the third and fourth properties specify *restrictions* on the agents. Although the exclusion of communication and memory may appear overly restrictive, our setting captures well-studied models of both collective insect behavior and swarm robotics, as outlined in Section 1.1.3.

From a mathematical perspective, our model captures the *combinatorial* aspects of dynamic distributed task allocation. In particular, as we show in Section 2, the problem can be reformulated as finding a *low distortion embedding* from symmetric difference to Hamming distance.

1.1.1 Formal statement

Formally, the problem is defined as follows. There are three positive integer parameters: n is the number of agents, k is the number of tasks, and D is the target *maximum switching cost*, which we define later. The goal is to define a set of n deterministic functions $f_1^{n,k}, f_2^{n,k}, \ldots, f_n^{n,k}$, one for each agent, with the following properties.

- **Input:** For each agent a, the function $f_a^{n,k}$ takes as input a *demand vector* $\vec{v} = \{v_1, v_2, \ldots, v_k\}$ where each v_i is a non-negative integer and $\sum_i v_i = n$. Each v_i is the number of agents required for task i, and the total number of agents required for tasks is exactly the total number of agents.
- **Output:** For each agent a, the function $f_a^{n,k}$ outputs some $i \in [k]$. The output of $f_a^{n,k}(\vec{v})$ is the task that agent a is assigned when the demand vector is $\vec{v}$.
- **Demand satisfied:** For all demand vectors $\vec{v}$ and all tasks i, we require that the number of agents a for which $f_a^{n,k}(\vec{v}) = i$ is exactly v_i. That is, the allocation of agents to tasks defined by the set of functions $f_1^{n,k}, f_2^{n,k}, \ldots, f_n^{n,k}$ exactly satisfies the demand vector.
- **Switching cost satisfied:** The *switching cost* of a pair $(\vec{v}, \vec{v'})$ of demand vectors is defined as the number of agents a for which $f_a^{n,k}(\vec{v}) \neq f_a^{n,k}(\vec{v'})$; that is, the number of agents that switch tasks if the demand vector changes from $\vec{v}$ to $\vec{v'}$ (or from $\vec{v'}$ to $\vec{v}$). We say that a pair of demand vectors $\vec{v}$, $\vec{v'}$ are *adjacent* if $|\vec{v} - \vec{v'}|_1 = 2$; that is, if we can get from $\vec{v}$ to $\vec{v'}$ by moving exactly one unit of demand from one task to another. The *maximum switching cost* of a set of functions $f_1^{n,k}, f_2^{n,k}, \ldots, f_n^{n,k}$ is defined as the maximum switching cost over all pairs of adjacent demand vectors; that is, the maximum number of agents that switch tasks in response to the movement of a single unit of demand from one task to another. We require that the maximum switching cost of $f_1^{n,k}, f_2^{n,k}, \ldots, f_n^{n,k}$ is at most D.

Question. *Given n and k, what is the minimum possible maximum switching cost D over all sets of functions $f_1^{n,k}, \ldots, f_n^{n,k}$?*

1.1.2 Remarks

Remark 1. The problem statement only considers the switching cost of pairs of *adjacent* demand vectors. We observe that this also implies a bound on the switching cost of non-adjacent vectors: if every pair of adjacent demand vectors has switching cost at most D, then every pair of demand vectors with ℓ_1 distance d has switching cost at most $D(d/2)$.

Remark 2. The problem statement is consistent with the properties of the agents listed above. In particular, the agents have complete information about the changing demand vector because for each agent, the function $f_a^{n,k}$ takes as input the current demand vector. The agents are heterogeneous because each agent a has a separate function $f_a^{n,k}$. The agents have no communication or memory because the *only* input to each function $f_a^{n,k}$ is the current demand vector.

Remark 3. Forbidding communication among agents is crucial in the formulation of the problem, as otherwise the problem would be trivial. In particular, it would always be possible to achieve maximum switching cost 1: when the current demand vector changes to an adjacent demand vector, the agents simply reach consensus about which single agent will move.

1.1.3 Applications

Collective insect behavior There are a number of hypotheses that attempt to explain the mechanism behind task allocation in ant colonies (see the survey [1]). One such hypothesis is the *response threshold model*, in which ants decide which task to perform based on individual preferences and environmental factors. Specifically, the model postulates that there is an environmental stimulus associated with each task, and each individual ant has an internal threshold for each task, whereby if the stimulus exceeds the threshold, then the ant performs that task. The response threshold model was introduced in the 70s and has been studied extensively since (for comprehensive background on this model see the survey [1] and the introduction of [7]).

Our setting captures the essence of the response threshold model since agents are permitted to behave based on individual preferences (property 2: agents are heterogeneous) and environmental factors (property 1: agents have complete information about the demand vector). We study whether models like the response threshold model can achieve low switching costs.

Inspired by collective insect behavior, researchers have also studied the response threshold model in the context of swarm robotics [2, 11, 24]. Our setting also relates more generally to swarm robotics:

Swarm robotics There is a body of work in swarm robotics specifically concerned with property 3 of our setting: eliminating the need for communication (e.g. [23, 3, 10, 18]). In practice, communication among agents may be unfeasible or costly. In particular, it may be unfeasible to build a fast and reliable network infrastructure capable of dealing with delays and failures, especially in a remote location.

Regarding property 4 of our setting (the agents are memoryless), it may be desirable for robots in a swarm to not rely on memory. For example, if a robot fails and its memory is lost, we may wish to be able to introduce a new robot into the system to replace it.

Concretely, dynamic task allocation in swarm robotics may be applicable to disaster containment [18, 25], agricultural foraging, mining, drone package delivery, and environmental monitoring [21].

1.2 Past Work

Our problem was previously studied only by Su, Su, Dornhaus, and Lynch [22], who presented two upper bounds and a lower bound.

The first upper bound is a very simple set of functions $f_1^{n,k}, \ldots, f_n^{n,k}$ with maximum switching cost $k-1$. Each agent has a unique ID in $[n]$ and the tasks are numbered from 1 to k. The functions $f_1^{n,k}, \ldots, f_n^{n,k}$ are defined so that for all demand vectors, the agents populate the tasks in order from 1 to k in order of increasing agent ID. That is, for each agent a, $f_a^{n,k}$ is defined as the task j such that $\sum_{i=0}^{j-1} d_i < \text{ID}(a)$ and $\sum_{i=0}^{j} d_i \geq \text{ID}(a)$. Starting with any demand vector, if one unit of demand is moved from task i to task j, the switching cost is at most $|i-j|$ because at most one agent from each task numbered between i and j (including i but not including j) shifts to a new task. Thus, the maximum switching cost is $k-1$.

The lower bound of Su et al. is also very simple. It shows that there does not exist a set of functions $f_1^{n,k}, \ldots, f_n^{n,k}$ with maximum switching cost 1 for $n \geq 2$ and $k \geq 3$. Suppose for contradiction that there exists a set of functions $f_1^{n,k}, \ldots, f_n^{n,k}$ with maximum switching cost 1 for $n = 2$ and $k = 3$ (the argument can be easily generalized to higher n and k).

Suppose the current demand vector is $[1, 1, 0]$, that is, one agent is required for each of tasks 1 and 2 while no agent is required for task 3. Suppose agents a and b are assigned to tasks 1 and 2, respectively, which we denote $[a, b, \emptyset]$. Now suppose the demand vector changes from $[1, 1, 0]$ to the adjacent demand vector $[1, 0, 1]$. Since the maximum switching cost is 1, only one agent moves, so agent b moves to task 3, so we have $[a, \emptyset, b]$. Now suppose the demand vector changes from $[1, 0, 1]$ to the adjacent demand vector $[0, 1, 1]$. Again, since the maximum switching cost is 1, agent a moves from task 1 to task 2 resulting in

$[\emptyset, a, b]$. Now suppose the demand vector changes from $[0, 1, 1]$ to the adjacent demand vector $[1, 1, 0]$, which was the initial demand vector. Since the maximum switching cost is 1, agent b moves from task 3 to task 1 resulting in $[b, a, \emptyset]$.

The problem statement requires that the allocation of agents depends *only* on the current demand vector, so the allocation of agents for any given demand vector must be the same regardless of the history of changes to the demand vector. However, we have shown that the allocation of agents for $[1, 1, 0]$ was initially $[a, b, \emptyset]$ and is now $[b, a, \emptyset]$, a contradiction. Thus, the maximum switching cost is at least 2.

The second upper bound of Su et al. states that there exists a set of functions $f_1^{n,k}, \dots, f_n^{n,k}$ with maximum switching cost 2 if $n \leq 6$ and $k = 4$. They prove this result by exhaustively listing all 84 demand vectors along with the allocation of agents for each vector.

1.3 Our results

We initiate the study of non-trivial lower bounds for the switching cost. In particular, with the current results it is completely plausible that the maximum switching cost can always be upper bounded by 2, regardless of the number of tasks and agents. Our results show that this is not true and provide further evidence that the maximum switching cost grows with the number of tasks.

One might expect that the limitations on n and k in the second upper bound of Su et al. is due to the fact the space of demand vectors grows exponentially with n and k so their method of proof by exhaustive listing becomes unfeasible. However, our first result is that the second upper bound of Su et al. is actually *tight* with respect to k. In particular, we show that achieving maximum switching cost 2 is impossible even for $k = 5$ (for *any* $n > 2$).

Theorem 1.1. *For $n \geq 3$, $k \geq 5$, every set of functions $f_1^{n,k}, \dots, f_n^{n,k}$ has maximum switching cost at least 3.*

We then consider the next natural question: *For what values of n and k is it possible to achieve maximum switching cost 3?* Our second result is that maximum switching cost 3 is not always possible:

Theorem 1.2. *There exist n and k such that every set of functions $f_1^{n,k}, \dots, f_n^{n,k}$ has maximum switching cost at least 4.*

The value of k for Theorem 1.2 is an extremely large constant derived from hypergraph Ramsey numbers. Specifically, there exists a constant c so that Theorem 1.2 holds for $n \geq 5$ and $k \geq t_{n-1}(cn)$ where the tower function $t_j(x)$ is defined by $t_1(x) = x$ and $t_{i+1}(x) = 2^{t_i(x)}$.

We remark that while our focus on small constant values of the switching cost may appear restrictive, functions with maximum switching cost 3 already have a highly non-trivial combinatorial structure.

1.4 Our techniques

We introduce two novel techniques, each tailored to a different parameter regime. One parameter regime is when $n \ll k$ and the demand for each task is either 0 or 1. This regime seems to be the most natural for the goal of proving the highest possible lower bounds on the switching cost.

1.4.1 The $n \ll k$ regime

We develop a proof framework for the $n \ll k$ regime and use it to prove Theorem 1.1 for $n = 3$, $k = 5$, and more importantly, to prove Theorem 1.2. We begin by supposing for contradiction that there exists a set of functions $f_1^{n,k}, \dots, f_n^{n,k}$ with switching cost 2 and 3, respectively, and then reason about the structure of these functions. The main challenge in proving Theorem 1.2 as compared to Theorem 1.1 is that functions with switching cost 3 can have a much more involved combinatorial structure than functions with switching cost 2. In principle, our proof framework could also apply to higher switching costs, but at present it is unclear how exactly to implement it for this setting.

The first step in our proofs is to reformulate the problem as that of finding a low distortion embedding from symmetric difference to Hamming distance, which we describe in Section 2. This provides a cleaner way to reason about the problem in the $n \ll k$ parameter regime. Our proofs are written in the language of the problem reformulation, but here we will briefly describe our proof framework in the language of the original problem statement.

The simple upper bound of $k-1$ described in Section 1.2 can be viewed as each agent having a "preference" for certain tasks. The main idea of our lower bound is to show that for *any* set of functions $f_1^{n,k}, \dots, f_n^{n,k}$ with low switching cost, many agents must have a "preference" for certain tasks. More formally, we introduce the idea of a task being *frozen* to an agent. A task t is frozen to agent a if for every demand vector in a particular large set of demand vectors, agent a is assigned to task t. Our framework has three steps:

- In step 1, we show roughly that in total, many tasks are frozen to some agent.
- In step 2, we show roughly that for many agents a, only few tasks are frozen to a.
- In step 3, we use a counting argument to derive a contradiction: we count a particular subset of frozen task/agent pairs in two different ways using steps 1 and 2, respectively.

The proof of Theorem 1.1 for $n = 3$ and $k = 5$ serves as a simple illustrative example of our proof framework, while the proof of Theorem 1.2 is more involved. In particular, in step 1 of the proof of Theorem 1.2, we derive *multiple* possible structures of frozen task/agent pairs. Then, we use Ramsey theory to show that there exists a collection of tasks that all obey only *one* of the possible structures. This allows us to reason about each of the possible structures independently in steps 2 and 3.

1.4.2 The remaining parameter regime

In the remaining parameter regime, we complete the proof of Theorem 1.1. In the previous parameter regime, we only addressed the $n = 3$, $k = 5$ case, and now we need to consider all larger values of n and k. Extending to larger k is trivial (we prove this formally in Section 4). However, it is not at all clear how to extend a lower bound to larger values of n. In particular, our proof framework from the $n \ll k$ regime immediately breaks down as n grows.

The main challenge of handling large n is that having an abundance of agents can actually allow *more* pairs of adjacent demand vectors to have switching cost 2, so it becomes more difficult to find a pair with switching cost greater than 2. To see this, consider the following example.

Consider the subset S_i of demand vectors in which a particular task i has an unconstrained amount of demand and each remaining task has demand at most $n/(k-1)$. We claim that there exists a set of functions $f_1^{n,k}, \dots, f_n^{n,k}$ so that every pair of adjacent demand vectors from S_i has switching cost 2. Divide the agents into $k-1$ groups of $n/(k-1)$ agents each, and associate each task except i to such a group of agents. We define the functions $f_1^{n,k}, \dots, f_n^{n,k}$ so that given any demand vector in S_i, the set of agents assigned to each task except i is simply a subset of the group of agents associated with that task (say, the subset of such agents with smallest ID). This is a valid assignment since the demand of each task except i is at most the size of the group of agents associated with that task. The remaining agents are assigned to task i. Then, given a pair $(\vec{v}, \vec{v'})$ of adjacent demand vectors in S_i, whose demands differ only for tasks s and t, their switching cost is 2 because the only agents assigned to different tasks between $\vec{v}$ and $\vec{v'}$ are: one agent from each of the groups associated with tasks s and t, respectively.

Because it is possible for many pairs of adjacent demand vectors to have switching cost 2, finding a pair of adjacent demand vectors with larger switching cost requires reasoning about a very precise set of demand vectors. To do this, we use roughly the following strategy. We identifying a task that serves the role of i in the above example and then successively move demand out of task i until task i is empty and can thus no longer fill this role. At this point, we argue that we have reached a pair of adjacent demand vectors with switching cost more than 2.

2 Problem reformulation

2.1 Notation

Let A and B be multisets. The intersection of A and B denoted $A \cap B$ is the maximal multiset of elements that appear in both A and B. For example, $\{a, a, b, b\} \cap \{a, b, b, c\} = \{a, b, b\}$. The symmetric difference between A and B, denoted $A \oplus B$, is the multiset of elements in either A or B but not in their intersection. For example, $\{a, a, b, b\} \oplus \{a, b, b, c\} = \{a, c\}$ since we are left with a after removing $\{a, b, b\}$ from $\{a, a, b, b\}$ and we are left with c after removing $\{a, b, b\}$ from a, b, b, c.

A permutation of a multiset A is simply a permutation of the elements of the multiset. For example, one permutation of $\{a, a, b\}$ is aba. We treat permutation as strings and perform string operations on them. For strings X and Y (which may be permutations), let $d(X, Y)$ denote the *Hamming distance* between X and Y. For example, $d(aba, bca) = 2$.

2.2 Problem statement

Given positive integers n, k, and D, the goal is to find a function $\pi_{n,k}$ with the following properties.

- Let $\mathcal{S}_{n,k}$ be the set of all size n multisets of $[k]$. The function $\pi_{n,k}$ takes as input a set $S \in \mathcal{S}_{n,k}$ and outputs a permutation of S.
- We say that a pair $S, S' \in \mathcal{S}_{n,k}$ has *distortion* D' with respect to $\pi_{n,k}$ if $|S \oplus S'| = 2$ and $d(\pi_{n,k}(S), \pi_{n,k}(S')) = D'$. In other words, a pair of multisets has distortion D' if they have the *smallest* possible symmetric distance but *large* Hamming distance (at least D'). We say that $\pi_{n,k}$ has *maximum distortion* D' if the maximum distortion over all pairs $S, S' \in \mathcal{S}_{n,k}$ with $|S \oplus S'| = 2$ is D'. We require that the function $\pi_{n,k}$ has maximum distortion at most D.

We are interested in the question of for which values of the parameters n, k, and D, there exists $\pi_{n,k}$ that satisfies the above properties. In particular, we aim to minimize the maximum distortion:

> **Question.** *Given n and k, what is the minimum possible maximum distortion over all functions $\pi_{n,k}$?*

In other words, the question is whether there exists a function $\pi_{n,k}$ such that *every* pair $S, S' \in \mathcal{S}_{n,k}$ has distortion at least D. Our theorems are lower bounds, so we show that for every function $\pi_{n,k}$ there *exists* a pair $S, S' \in \mathcal{S}_{n,k}$ with distortion at least D.

2.3 Equivalence to original problem statement

We claim that the new problem statement from Section 2.2 is equivalent to the original problem statement from Section 1.1.

Claim 1. *Given parameters n and k (the same for both problem statements) there exists a function $\pi_{n,k}$ with maximum distortion D if and only if there exists a set of functions $f_1^{n,k}, \ldots, f_n^{n,k}$ with maximum switching cost D.*

We describe the correspondence between the two problem statements:

- **Demand vector.** $\mathcal{S}_{n,k}$ is the set of all possible demand vectors since a demand vector is simply a size n multiset of the k tasks. For example, the multiset $S = \{1, 1, 3\}$ is equivalent to the demand vector $\vec{v} = [2, 0, 1]$; both notations indicate that task 1 requires two units of demand, task 2 requires no demand, and task 3 requires one unit of demand.

- **Allocation of agents to tasks.** If $\vec{v}$ is the demand vector representing the multiset $S \in \mathcal{S}_{n,k}$, a permutation $\pi_{n,k}(S)$ is an allocation $f_1^{n,k}(\vec{v}), \ldots, f_n^{n,k}(\vec{v})$ of agents to tasks so that $\pi_{n,k}(S)[i] = f_i^{n,k}(\vec{v})$; that is, agent i performs the task that is the i^{th} element in the permutation $\pi_{n,k}(S)$. For example, $\pi_{3,3}(\{1, 1, 3\}) = 131$ is equivalent to the following: $f_1^{3,3}([2, 0, 1]) = 1$, $f_2^{3,3}([2, 0, 1]) = 3$, and $f_3^{3,3}([2, 0, 1]) = 1$; both notations indicate that agents 1 and 3 both performs task 1, while agent 2 performs task 2.

- **Switching cost.** If $\vec{v}, \vec{v'}$ are the demand vectors representing the multisets $S, S' \in \mathcal{S}_{n,k}$ respectively, the value $d(\pi_{n,k}(S), \pi_{n,k}(S'))$ is the switching cost because from the previous bullet point, $\pi_{n,k}(S)[i] \neq \pi_{n,k}(S')[i]$ if and only if $f_a^{n,k}(\vec{v}) \neq f_a^{n,k}(\vec{v'})$.

- **Adjacent demand vectors.** The set of all pairs $S, S' \in \mathcal{S}_{n,k}$ such that $|S \oplus S'| = 2$ is the set of all pairs of adjacent demand vectors. This is because $|S \oplus S'| = 2$ means that starting from S, one can reach S' by changing exactly one element in S from some $i \in [k]$ to some $j \in [k]$. Equivalently, starting from the demand vector represented by S and moving one unit of demand from task i to task j results in the demand vector represented by S'.

- **Maximum switching cost.** If $f_1^{n,k}, \ldots, f_n^{n,k}$ is the set of functions representing $\pi_{n,k}$, then $\pi_{n,k}$ has maximum distortion D if and only if $f_1^{n,k}, \ldots, f_n^{n,k}$ has maximum switching cost D. This is because $S, S' \in \mathcal{S}_{n,k}$ has distortion D if and only if $|S \oplus S'| = 2$ and $d(\pi_{n,k}(S), \pi_{n,k}(S')) = D$ which is equivalent to saying that the demand vectors $\vec{v}$ and $\vec{v'}$ that represent S and S' are adjacent and have switching cost D.

2.4 Restatement of results

We restate Theorems 1.1 and 1.2 in the language of the problem restatement.

Theorem 2.1 (Restatement of Theorem 1.1)**.** *Let $n \geq 3$ and $k \geq 5$. Every function $\pi_{n,k}$ has maximum distortion at least 3.*

Theorem 2.2 (Restatement of Theorem 1.2)**.** *There exist n and k so that every function $\pi_{n,k}$ has maximum distortion at least 4.*

2.5 Example instance

To build intuition about the problem restatement, we provide a concrete example of a small instance of the problem. Suppose $n = 3$ and $k = 2$. For notational clarity, instead of denoting $[k] = \{0, 1\}$ we denote $[k] = \{a, b\}$. Then $\mathcal{S}_{3,2}$ is the set of all size 3 multisets of $\{a, b\}$; that is, $\mathcal{S}_{3,2} = \{\{a, a, a\}, \{a, a, b\}, \{a, b, b\}, \{b, b, b\}\}$.

$\pi_{3,2}$ is a function that maps each element of $\mathcal{S}_{3,2}$ to a permutation of itself. For example, $\pi_{3,2}$ could be defined as follows:

$$\pi_{3,2}(\{a,a,a\}) = aaa, \quad \pi_{3,2}(\{a,a,b\}) = aba \quad \pi_{3,2}(\{a,b,b\}) = bab, \quad \pi_{3,2}(\{b,b,b\}) = bbb.$$

We are concerned with all pairs $S, S' \in \mathcal{S}_{3,2}$ such that $|S \oplus S'| = 2$ (since the maximum distortion of $\pi_{3,2}$ is defined in terms of only these pairs). In this example, the only such pairs are as follows:

$$\{a,a,a\} \oplus \{a,a,b\} = 2, \quad \{a,a,b\} \oplus \{a,b,b\} = 2, \quad \{a,b,b\} \oplus \{b,b,b\} = 2.$$

For each such pair, we consider $d(\pi_{3,2}(S), \pi_{3,2}(S'))$:

$$d(aaa, aba) = 1, \quad d(aba, bab) = 3, \quad d(bab, bbb) = 1.$$

This particular choice of $\pi_{3,2}$ has maximum distortion 3 (since the largest value in the above row is 3), however we could have chosen $\pi_{3,2}$ with maximum distortion 1 (for example if $\pi_{3,2}(\{a,b,b\}) = bba$ instead of bab).

3 The $n \ll k$ regime

In this section we will prove Theorem 2.1 for $n = 3$, $k = 5$, and Theorem 2.2. The proofs are written in the language of the problem reformulation from Section 2. For these proofs it will suffice to consider only the elements of $\mathcal{S}_{n,k}$ that are *subsets* of $[k]$, rather than multisets. This corresponds to the set of demand vectors where each task has demand either 0 or 1. *For the rest of this section we consider only subsets of* $[k]$*, rather than multisets.*

We call each element of $[k]$ a *character* (e.g. in the above example instance, a and b are characters).

3.1 Proof framework

As described in Section 1.4, we develop a three-step proof framework for the $n \ll k$ regime. Suppose we are trying to prove that every function $\pi_{n,k}$ has maximum distortion at least D for a particular n and k. We begin by supposing for contradiction that there exists $\pi_{n,k}$ with maximum distortion less than D. That is, we suppose that every pair $S, S' \in \mathcal{S}_{n,k}$ with $|S \oplus S'| = 2$ has $d(\pi_{n,k}(S), \pi_{n,k}(S')) < D$. Under the assumption that such a $\pi_{n,k}$ exists, steps 1 and 2 of the framework show that $\pi_{n,k}$ must obey a particular structure. For the remainder of this section, we drop the subscript of π since n and k are fixed.

Notation. For any set $R \subseteq [k]$, let $\mathcal{U}_R$ be the set of all sets $S \subseteq [k]$ such that $R \subset S$ and $|S| = |R| + 1$.

Step 1: Structure of size $n-1$ sets. We begin by fixing a size $n-1$ set $R \subseteq [k]$. Now, consider $\mathcal{U}_R$ (defined above). We note that all pairs $S, S' \in \mathcal{U}_R$ are by definition such that $|S \oplus S'| = 2$. Because we initially supposed that π has maximum distortion less than D, we know that for all pairs $S, S' \in \mathcal{U}_R$, we have $d(\pi(S), \pi(S')) < D$.

Then we prove a structural lemma which roughly says that many characters $r \in R$ have a "preference" to be in a particular position in the permutations $\pi(S)$ for $S \in \mathcal{U}_R$. We say that R *i-freezes* the character r if $\pi(S)[i] = r$ for many $S \in \mathcal{U}_R$. Our structural lemma roughly says that for many characters $r \in R$, there exists an index $i \in [n]$ such that R i-freezes r. In other words, for many $S \in \mathcal{U}_R$, the $\pi(S)$s agree on the position of many characters in the permutation.

Step 2: Structure of size $n-2$ sets. We begin by fixing a size $n-2$ set $Q \subseteq [k]$. Now, consider $\mathcal{U}_Q$. We note that each $R \in \mathcal{U}_Q$ obeys the structural lemma from step 1; that is, for many characters $r \in R$, there exists an index $i \in [n]$ such that R i-freezes r.

We prove a structural lemma which roughly says that the sets $P \in \mathcal{U}_Q$ are for the most part *consistent* about which characters they freeze to which index of the permutation. More specifically, for many characters $q \in Q$, for all pairs $P, P' \in \mathcal{U}_Q$, if R i-freezes r and R' j-freezes r, then $i = j$.

Step 3: Counting argument. In step 3, we use a counting argument to derive a contradiction. For the proof of Theorem 2.1, a simple argument suffices. The idea is that step 1 shows that many characters are frozen overall while step 2 shows that each character can only be frozen to a single index. Then, the pigeonhole principle implies that more than one character is frozen to a single index, which helps to derive a contradiction.

For the proof of Theorem 2.2, it no longer suffices to just show that more than one character is frozen to a single index. Instead, we require a more sophisticated counting argument and a careful choice of what quantity to count. We end up counting the number of pairs (Q, a) such that $R \in \mathcal{U}_Q$, where $Q \subset [k]$ is a size $n-2$ set and $a \in [n] \setminus Q$. To reach a contradiction, we count this quantity in two different ways, using steps 1 and 2 respectively.

Having reached a contradiction, we conclude that π has maximum distortion at least D.

3.2 Proof of Theorem 2.1 for $n = 3$, $k = 5$

In this section, we prove Theorem 2.1 for $n = 3$, $k = 5$, which serves as a simple illustrative example of our proof framework from Section 3.1.

Theorem 3.1 (Special case of Theorem 2.1)**.** *Every function $\pi_{3,5}$ has maximum distortion at least 3.*

Proof. Suppose by way of contradiction that there is a function $\pi_{3,5}$ with maximum distortion at most 2. For the remainder of this section we omit the subscript of π since $n = 3$, $k = 5$ are fixed. For clarity of notation, we let $\{a, b, c, d, e\}$ be the characters in $[k]$ for $k = 5$. Thus, we are considering the set of all $\binom{5}{3} = 10$ size 3 subsets of $\{a, b, c, d, e\}$. (Recall that we are only concerned with subsets, not multisets.)

Step 1: Structure of size $n-1$ sets

We begin by fixing a set $\{x, y\} \subseteq \{a, b, c, d, e\}$ of size $n - 1 = 2$. Recall that $\mathcal{U}_{\{x,y\}}$ is the set of all size 3 sets S such that $\{x, y\} \subseteq S \subseteq \{a, b, c, d, e\}$. For example, $\mathcal{U}_{\{a,b\}} = \{\{a, b, c\}, \{a, b, d\}, \{a, b, e\}\}$. We note that by definition all pairs $S, S' \in \mathcal{U}_{\{x,y\}}$ have $|S \oplus S'| = 2$. Thus, to find a pair with distortion 3 and thereby obtain a contradiction, it suffices to find a pair $S, S' \in \mathcal{U}_{\{x,y\}}$ with Hamming distance $d(\pi(S), \pi(S')) = 3$. Since $n = 3$, this means we are looking for permutations $\pi(S), \pi(S')$ that disagree about the position of *all* elements.

The following lemma says that π places one of x or y at the *same* position for *all* $\pi(S)$ with $S \in \mathcal{U}_{\{x,y\}}$. For ease of notation, we give this phenomenon a name:

Definition 3.1 (freeze)**.** We say that a pair $\{x, y\} \subseteq \{a, b, c, d, e\}$ *i-freezes* a character $p \in \{x, y\}$ if for all $S \in \mathcal{U}_{\{x,y\}}$, we have $\pi(S)[i] = p$. We simply say that $\{x, y\}$ *freezes* p if i is unspecified. Equivalently, we say that a character p is *i-frozen* (or just *frozen*) by a pair.

Lemma 3.1. *For every $\{x, y\} \subseteq \{a, b, c, d, e\}$, there exists i so that $\{x, y\}$ i-freezes either x or y.*

For example, one way that the pair $\{a, b\}$ could satisfy Lemma 3.1 is if the permutations $\pi(\{a, b, c\})$, $\pi(\{a, b, d\})$, and $\pi(\{a, b, e\})$ *all* place the character a in the 0^{th} position. In this case, we would say that the pair $\{a, b\}$ 0-freezes a.

Proof of Lemma 3.1. Without loss of generality, consider $\{x, y\} = \{a, b\}$. In this case, $\mathcal{U}_{\{x,y\}} = \mathcal{U}_{\{a,b\}} = \{\{a, b, c\}, \{a, b, d\}, \{a, b, e\}\}$. Thus, we are trying to show that $\{a, b, c\}$, $\{a, b, d\}$, and $\{a, b, e\}$ *all* agree on the position of either a or b.

Suppose without loss of generality that $\pi(\{a, b, c\}) = abc$. We first note that $\pi(\{a, b, c\})$ and $\pi(\{a, b, d\})$ must agree on the position of either a or b because otherwise we would have $d(\pi(\{a, b, c\}), \pi(\{a, b, d\})) = 3$ which would mean that $\pi(\{a, b, c\})$ and $\pi(\{a, b, d\})$ would have distortion 3, and we would have proved Theorem 3.1. Without loss of generality, suppose $\pi(\{a, b, c\})$ and $\pi(\{a, b, d\})$ agree on the position of a; that is, $\pi(\{a, b, d\})$ is either abd or adb.

By the same reasoning, $\pi(\{a, b, c\})$ and $\pi(\{a, b, e\})$ agree on the position of either a or b, and $\pi(\{a, b, d\})$ and $\pi(\{a, b, e\})$ agree on the position of either a or b. If $\pi(\{a, b, e\})$ agrees with *either* $\pi(\{a, b, c\})$ or $\pi(\{a, b, d\})$ on the position of a, then it agrees with *both* (in which case we are done) since $\pi(\{a, b, c\})$ and $\pi(\{a, b, d\})$ agree on the position of a, by the previous paragraph. Thus, the only option is that $\pi(\{a, b, e\})$ agrees with *both* $\pi(\{a, b, c\})$ and $\pi(\{a, b, d\})$ on the position of b. This completes the proof. □

Step 2: Structure of size $n - 2$ sets

Since $n - 2 = 1$, we begin by fixing a single element $x \in \{a, b, c, d, e\}$. In the following lemma we prove that x cannot be frozen to two different indices.

Lemma 3.2. *If a pair $\{x, y\} \subseteq \{a, b, c, d, e\}$ i-freezes x and a pair $\{x, z\} \subseteq \{a, b, c, d, e\}$ j-freezes x then $i = j$.*

Proof. Since $\{x, y\}$ i-freezes x, then in particular, $\pi(\{x, y, z\})[i] = x$. Since $\{x, z\}$ j-freezes x, then in particular, $\pi(\{x, y, z\})[j] = x$. A single character cannot be in multiple positions of the permutation $\pi(\{x, y, z\})$ so $i = j$. □

Step 3: Counting argument

Lemma 3.1 implies that for each character $x \in \{a, b, c, d, e\}$ except for at most one, *some* pair $\{x, y\}$ freezes x. That is, at least 4 characters are frozen by some pair. However $n = 3$ so by the pigeonhole principle, two characters $x, y \in \{a, b, c, d, e\}$ are frozen to the same index i.

Fix x, y, and i, and suppose x and y are each i-frozen. By Lemma 3.1, the pair $\{x, y\}$ freezes either x or y. Without loss of generality, say $\{x, y\}$ freezes x. By Lemma 3.2, since x is i-frozen by some pair, *all* pairs that freeze x must i-freeze x. Thus, the pair $\{x, y\}$ i-freezes x.

Let $\{y, z\} \subseteq \{a, b, c, d, e\}$ be a pair that i-freezes y. Thus we have $\pi(\{x, y, z\})[i] = y$. However, since $\{x, y\}$ i-freezes x, we also have $\pi(\{x, y, z\})[i] = x$. This is a contradiction since $\pi(\{x, y, z\})[i]$ cannot take on two different values. □

3.3 Proof of Theorem 2.2

Theorem 3.2 (Restatement of Theorem 2.2)**.** *There exist n and k so that every function $\pi_{n,k}$ has maximum distortion at least 4.*

More specifically, we will show that there exists a constant c so that Theorem 1.2 holds for $n \geq 5$ and $k \geq t_{n-1}(cn)$ where the tower function $t_j(x)$ is defined by $t_1(x) = x$ and $t_{i+1}(x) = 2^{t_i(x)}$.

Proof. Suppose by way of contradiction that there is a function $\pi_{n,k}$ with maximum distortion at most 3, for n and k to be set later.

For the remainder of this section we omit the subscript of π since n and k are fixed. As a convention, we will generally use the variables P, Q, R, and S to refer to subsets of $[k]$ of size $n - 3$, $n - 2$, $n - 1$, and n, respectively.

3.3.1 Step 1: Structure of size $n-1$ sets.

Let $R \subset [k]$ be a size $n-1$ set. Recall from Section 3.1 that $\mathcal{U}_R$ is the set of all size n sets S such that $R \subset S \subset [k]$. We note that all pairs $S, S' \in \mathcal{U}_R$ are by definition such that $|S \oplus S'| = 2$. Because we initially supposed that π has maximum distortion at most 3, we know that all pairs $S, S' \in \mathcal{U}_R$ have Hamming distance $d(\pi(S), \pi(S')) \leq 3$.

We begin by generalizing the notion of *freezing* a character from Definition 3.1. Instead of freezing a single character, our new definition will concern freezing a *set* of characters. Freezing a set of characters essentially means that every character in the set is frozen to a different index.

Definition 3.2 (freeze)**.** Let $R \subseteq [k]$ be a size $n-1$ set and let $A_R \subseteq R$. We say that R *freezes* A_R with *freezing function* g_R if g_R is a one-to-one mapping from A_R to $[n]$ such that for all $a \in A_R$ and all $S \in \mathcal{U}_R$, we have $\pi(S)[g_R(a)] = a$.

Unlike in the proof of Theorem 3.1, it is not true that for any size $n-1$ set $R \subset [k]$, some subset of R must be frozen. Instead, to capture the full structure of permutations with Hamming distance 3, we will need another notion of freezing, which we call *semi-freezing*. In this definition, each character is restricted to *two* indices instead of just one.

Definition 3.3 (semi-freeze)**.** Let $R \subset [k]$ be a size $n-1$ set. We say that R is *semi-frozen* with *semi-freezing function* h_R and *wildcard index* w_R if h_R is a one-to-one mapping from R to $[n]$ such that for all $r \in R$, we have that for all $S \in \mathcal{U}_R$ either $\pi(S)[h_R(r)] = r$ or $\pi(S)[w_R] = r$.

We note that since g_R is a one-to-one mapping and R is of size $n-1$, the *only* index in $[n]$ not mapped to by g_R is the wildcard index w_R. We call w_R the wildcard index because $\pi(S)$ could place *any* character from R at index w_R. In contrast, for every other index i, $\pi(S)$ can only place a single character from R at index i, namely the character mapped to i by the function g_R.

Our structural lemma for step 1 says that either R freezes a large subset $A \subset R$, or R is semi-frozen.

Lemma 3.3. *Every set $R \subset [k]$ of size $n-1$, obeys one of the following two configurations:*

1. *there exists a set $A_R \subset R$ of size $n-3$ such that R freezes A_R, or*

2. *R is semi-frozen.*

Figure 1 shows the structure of permutations that obey each of the two configurations in Lemma 3.3. In configuration 1, each character in a *large subset* of R is always mapped to a *single* index. In configuration 2, each character in R is always mapped to one of *two* possible choices. In other words, both configurations enforce a rigid structure but each of them are flexible in a different way. Configuration 1 is flexible in that it does not impose structure on characters not in A_R, and rigid in that the characters in A_R are *always* mapped to the same position. On the other hand, configuration 2 is flexible in that it allows each character to map to a choice of two positions, but rigid in that the structure is imposed on *every* character in R.

To prove Lemma 3.3, we would like to initially fix a pair $S, S' \in \mathcal{U}_R$ with $d(\pi(S), \pi(S')) = 3$. The following lemma proves that we can assume that such a pair exists, because if not, then configuration 1 of Lemma 3.3 already holds. The proof of the following lemma is nearly identical to the proof of Lemma 3.1.

Lemma 3.4. *If all pairs $S, S' \in \mathcal{U}_R$ have $d(\pi(S), \pi(S')) \leq 2$, then for every size $n-1$ set $R \subset [k]$, there exists a subset A_R of size $n-2$ such that R freezes A_R.*

Proof. Any pair $S, S' \in \mathcal{U}_R$ must agree on the position of at least $n-2$ characters in R because otherwise we would have $d(\pi(S), \pi(S')) > 2$. Also, there must exist a pair $S, S' \in \mathcal{U}_R$ with $d(\pi(S), \pi(S')) = 2$ because if all pairs had $d(\pi(S), \pi(S')) = 1$ then all $S \in \mathcal{U}_R$ would agree on the position of all $n-1$ characters in R and we would be done. Thus, let $S, S' \in \mathcal{U}_R$ be such that $d(\pi(S), \pi(S')) = 2$.

a	b	c	d	e
a	b	d	c	f
a	b	g	c	d
a	b	c	d	h
a	b	i	c	d
a	b	d	j	c

a	b	c	d	e
a	b	c	f	d
a	b	g	d	c
a	b	h	d	c
a	i	c	d	b
j	b	c	d	a

Figure 1: Examples of the configurations from Lemma 3.3. Each of the two subfigures shows the the set of permutations $\pi(S)$ for each $S \in \mathcal{U}_R$ where $R = \{a, b, c, d\}$, $n = 5$, and $k = 10$. The left subfigure shows configuration 1 of Lemma 3.3: the frozen set is $A = \{a, b\}$ since a and b each only appear at a fixed index, as marked by the gray box. The right subfigure shows configuration 2 of Lemma 3.3: R is semi-frozen with wildcard index indicated by the gray box since each element of R only appears at the wildcard index and one other index.

Let $R = \{a_1, a_2, \ldots, a_{n-1}\}$ and without loss of generality suppose $\pi(S)$ and $\pi(S')$ agree on the position of the characters $a_1, a_2, \ldots, a_{n-2}$ and disagree on the position of a_{n-1}. We wish to show that for all $S'' \in \mathcal{U}_R$, $\pi(S'')$ also agrees with $\pi(S)$ and $\pi(S')$ on the position of the characters $a_1, a_2, \ldots, a_{n-2}$. Suppose for contradiction that there exists $S'' \in \mathcal{U}_R$ such that $\pi(S'')$ disagrees with $\pi(S)$ and $\pi(S')$ on the position of some character in $\{a_1, a_2, \ldots, a_{n-2}\}$. Then since $\pi(S'')$ must agree with both $\pi(S)$ and $\pi(S')$ on the position of at least $n-2$ characters in R, $\pi(S'')$ must agree with *both* $\pi(S)$ and $\pi(S')$ on the position of a_{n-1}. But, this is a contradiction because $\pi(S)$ and $\pi(S')$ disagree on the position of a_{n-1}. □

Proof of Lemma 3.3. Let $S, S' \in \mathcal{U}_R$ be such that $d(\pi(S), \pi(S')) = 3$. Such S, S' exist by Lemma 3.4. Fix $S, S' \in \mathcal{U}_R$. Let $R = \{a_1, a_2, \ldots, a_{n-1}\}$ and without loss of generality suppose $\pi(S)$ and $\pi(S')$ agree on the position of the $n-3$ characters $a_3, a_4, \ldots, a_{n-1}$. If every $S'' \in \mathcal{U}_R$ is such that $\pi(S'')$ also agrees with $\pi(S)$ and $\pi(S')$ on the positions of the characters $a_3, a_4, \ldots, a_{n-1}$, then we are done because in this case R freezes the set $\{a_3, a_4, \ldots, a_{n-1}\}$. So suppose otherwise; that is, let $S'' \in \mathcal{U}_R$ be such that $\pi(S'')$ disagrees with $\pi(S)$ and $\pi(S')$ on the position of a_3 (without loss of generality).

Since $S, S', S'' \in \mathcal{U}_R$, each of S, S', and S'' have one additional character besides those in R. Let s, s', and s'' be these characters respectively. In the following we will analyze $\pi(S)$, $\pi(S')$ and $\pi(S'')$. Since $a_4, a_5, \ldots a_{n-1}$ are all in the same position with respect to all three permutations, we will ignore these characters. That is, letting $Z = \{a_4, a_5, \ldots a_{n-1}\}$, we consider $S \setminus Z = \{a_1, a_2, a_3, s\}$, $S' \setminus Z = \{a_1, a_2, a_3, s'\}$, and $S'' \setminus Z = \{a_1, a_2, a_3, s''\}$. We will abuse notation and let $\pi(S \setminus Z)$ be the subpermutation of $\pi(S)$ containing only the elements of $S \setminus Z$, and similarly for $S' \setminus Z$ and $S'' \setminus Z$.

Suppose without loss of generality that $\pi(S \setminus Z) = a_1 a_2 s a_3$. Then since $\pi(S)$ and $\pi(S')$ agree on the position of a_3 but disagree on the positions of a_1 and a_2, we have that either

1. $\pi(S')$ places s' in the position that $\pi(S)$ places s, so $\pi(S' \setminus Z) = a_2 a_1 s' a_3$, or
2. $\pi(S')$ places s' in the position that $\pi(S)$ places a_1 or a_2, so without loss of generality $\pi(S' \setminus Z) = s' a_1 a_2 a_3$.

Recall that $\pi(S'')$ disagrees with $\pi(S)$ on the position of a_3. Since $d(\pi(S''), \pi(S)) \leq 3$ and the positions of a_3 and s'' each account for one unit of difference between $\pi(S'')$ and $\pi(S)$, we know that $\pi(S'')$ agrees with $\pi(S)$ on the position of at least one of a_1 or a_2. Similarly, since $\pi(S'')$ disagrees with $\pi(S')$ on the position of a_3, we have that $\pi(S'')$ agrees with $\pi(S')$ on the position of at least one of a_1 or a_2. If $\pi(S' \setminus Z) = a_2 a_1 s' a_3$ (case 1 above), then $\pi(S'')$ cannot possibly agree with both $\pi(S)$ and $\pi(S')$ on the position of at least one of a_1 or a_2 because the positions of a_1 and a_2 are swapped in $\pi(S)$ as compared to $\pi(S')$. Thus, it must be the case that $\pi(S' \setminus Z) = s' a_1 a_2 a_3$ (case 2 above).

Now, given that $\pi(S \setminus Z) = a_1a_2sa_3$ and $\pi(S' \setminus Z) = s'a_1a_2a_3$, there is only one possibility for $\pi(S'' \setminus Z)$ that satisfies the criteria that $\pi(S'')$ disagrees with $\pi(S)$ and $\pi(S')$ on the position of a_3 and agrees with each of $\pi(S)$ and $\pi(S')$ on the position of at least one of a_1 or a_2. The only possibility is that $\pi(S'' \setminus Z) = a_1a_3a_2s''$.

The above argument applies for any $S'' \in \mathcal{U}_R$ such that $\pi(S'')$ disagrees with $\pi(S)$ and $\pi(S')$ on the position of some a_i with $3 \leq i \leq n-1$. That is, letting $Z' = \{a_3, a_4, \ldots a_{n-1}\} \setminus \{a_i\}$ and letting $s'' = S'' \setminus R$, we have that without loss of generality, $\pi(S \setminus Z') = a_1a_2sa_i$, $\pi(S' \setminus Z') = s'a_1a_2a_i$, and $\pi(S'' \setminus Z) = a_1a_ia_2s''$.

We claim that the structure we have derived implies that R is semi-frozen. To see this, consider the following semi-freezing function h_R:

$$h_R(a_1) = \pi(S)[a_1],$$
$$h_R(a_2) = \pi(S')[a_2],$$
$$\text{for each element } a_i \text{ for } 3 \leq i \leq n-1,\ h_R(a_i) = \pi(S)[a_i] = \pi(S')[a_i],$$
$$\text{and the wildcard index } w_R = \pi(S)[a_2] = \pi(S')[a_1].$$

□

3.3.2 Treating configurations 1 and 2 independently

Before moving to step 2 of the proof framework, we will show using Ramsey theory that it suffices to consider each of the two configurations from Lemma 3.3 *independently*. Lemma 3.3 shows that every size $n-1$ subset of $[k]$ obeys one of two configurations. Using Ramsey theory, we will show that there must exist a subset $K' \subseteq [k]$ such that either *all* size $n-1$ subsets of K' obey configuration 1 or *all* size $n-1$ subsets of K' obey configuration 2. This will allow us to avoid reasoning about the complicated interactions between the two configurations.

The required size $k' = |K'|$ can be expressed as a *hypergraph Ramsey number*. The hypergraph Ramsey number $r_j(t,t)$ is the minimum value m such that every red-blue coloring of the j-tuples of an m-element set contains either a red set or a blue set of size t, where a set is called red (blue) if all j-tuples from this set are red (blue). Thus, it suffices to let k' satisfy $r_{n-1}(k', k') = k$.

Erdős and Rado [8] give the following bound on $r_j(t,t)$, as stated in [4]. There exists a constant c such that:

$$r_{n-1}(k', k') \leq t_{n-1}(ck')$$

where the tower function $t_j(x)$ is defined by $t_1(x) = x$ and $t_{i+1}(x) = 2^{t_i(x)}$.

In the following we will show that it suffices to let $n \geq 5$ and $k' \geq n+7$, so it suffices to set $n \geq 5$ and $k \geq t_{n-1}(cn)$.

3.3.3 Step 2a: Structure of size $n-2$ sets for configuration 1

From the previous section, there exists a size k' set K' of tasks such that either all size $n-1$ subsets of K' obey configuration 1 or all size $n-1$ subsets of K' obey configuration 2. In this section we will assume that all size $n-1$ subsets of K' obey configuration 1, and later we will independently consider configuration 2.

Recall that configuration 1 says that for every set $R \subset [k']$ of size $n-1$, there exists a set $A_R \subset R$ of size $n-3$ such that R freezes A_R. Recall that g_R is the freezing function.

Let $Q \subset [k']$ be a size $n-2$ set. Recall that $\mathcal{U}_Q$ is the set of all size $n-1$ sets R such that $Q \subset R \subset [k']$. We will prove the following simple structural lemma, analogous to Lemma 3.2, which says that the freezing functions for any two sets in $\mathcal{U}_Q$ are *consistent*.

Lemma 3.5. *For every size $n-2$ set $Q \subset [k']$, for any pair $R, R' \in \mathcal{U}_Q$, for any $a \in A_R \cap A_{R'}$, $g_R(a) = g_{R'}(a)$.*

Proof. Fix $R, R' \in \mathcal{U}_Q$. Since R and R' are each composed by adding a single character to Q, we have that $|R \cup R'| = n$ and $R \cup R' \in \mathcal{U}_R \cap \mathcal{U}_{R'}$.

Since $R \cup R' \in \mathcal{U}_R$, we know that a is at position $g_R(a)$ in $\pi(R \cup R')$ and since $R \cup R' \in \mathcal{U}_{R'}$, we know that a is at position $g_{R'}(a)$ in $\pi(R \cup R')$. Then, since a can only occupy a single position in the permutation $\pi(R \cup R')$, we have that $g_R(a) = g_{R'}(a)$. □

3.3.4 Step 3a: Counting argument for configuration 1

Like the previous section, in this section we will assume that all size $n-1$ subsets of K' obey configuration 1. Unlike step 3 of Theorem 3.1, it does not suffice to simply show that two characters are frozen to the same index. Instead, we apply a more sophisticated counting argument.

By Lemma 3.5, for every size $n-2$ set $Q \subset [k']$, we have that all $R \in \mathcal{U}_Q$ agree on the value of $g_R(a)$ if it exists. Thus, we can define G_Q as the union of g_Rs over all $R \in \mathcal{U}_Q$. Formally, for any $a \in [k']$, $G_Q(a) = i$ if for every $R \in \mathcal{U}_Q$ with $a \in A_R$, we have $g_R(a) = i$. We note that $G_Q(a)$ *exists* if for *some* $R \in \mathcal{U}_Q$, $a \in A_R$.

Since $|Q| = n-2$ and there are n indices total, $G_Q(a)$ can exist for at most $n-2$ characters $a \in Q$ and at most 2 characters $a \notin Q$. We say that the pair (Q, a) is *irregular* if $G_Q(a)$ exists and $a \notin Q$. The quantity that we will count is the total number of irregular pairs (Q, a) over all size $n-2$ sets $Q \subset [k']$ and all $a \in [k']$.

On one hand, as previously mentioned, each set Q can only be in at most 2 irregular pairs. Then since there are $\binom{k'}{n-2}$ sets $Q \subset [k']$ of size $n-2$, the total number of irregular pairs is at most $2\binom{k'}{n-2}$.

On the other hand, the definition of configuration 1 implies a lower bound on the number of irregular pairs. Recall that configuration 1 says that for every size $n-1$ set $R \subset [k']$, there exists a set $A_R \subset R$ of size $n-3$ such that R freezes A_R. Fix sets R and A_R. We claim that for each $a \in A_R$, the pair $(R \setminus \{a\}, a)$ is an irregular pair. Firstly, is clear that $a \notin R \setminus \{a\}$. Secondly, $G_{R\setminus\{a\}}(a)$ exists because $R \in \mathcal{U}_{R\setminus\{a\}}$ and $a \in A_R$. Thus, for each $a \in A_R$, the pair $(R \setminus \{a\}, a)$ is an irregular pair.

Thus, every size $n-1$ set $R \subset [k']$ *produces* $n-3$ irregular pairs $(R \setminus \{a\}, a)$. Furthermore, given an irregular pair (Q, a), there is only one set that could produce it, namely $Q \cup \{a\}$. Then since there are $\binom{k'}{n-1}$ sets $R \subset [k']$ of size $n-1$, we have that the total number of irregular pairs is at least $(n-3)\binom{k'}{n-1}$.

Thus, we have shown that the total number of irregular pairs is at most $2\binom{k'}{n-2}$ and at least $(n-3)\binom{k'}{n-1}$. Therefore, we have reached a contradiction if $2\binom{k'}{n-2} < (n-3)\binom{k'}{n-1}$ which is true if $n \geq 4$ and $k' > \frac{n^2-3n+4}{n-3}$. In particular, $n \geq 5$, $k' \geq n+7$ satisfy these bounds.

3.3.5 Step 2b: Structure of size $n-2$ sets for configuration 2

From Section 3.3.2, there exists a size k' set K' of tasks such that either all size $n-1$ subsets of K' obey configuration 1 or all size $n-1$ subsets of K' obey configuration 2. We have already considered the configuration 1 case and now we will assume that all size $n-1$ subsets of K' obey configuration 2. Recall that configuration 2 says that R is semi-frozen. Recall that h_R is the semi-freezing function and w_R is the wildcard index.

Let $Q \subset [k']$ be a size $n-2$ set. Recall that $\mathcal{U}_Q$ is the set of all size $n-1$ sets R such that $Q \subset R \subset [k']$. We will prove the following structural lemma, which says that the semi-freezing functions for two sets in $\mathcal{U}_Q$ are in some sense *consistent*.

Lemma 3.6. *For every size $n-2$ set $Q \subset [k']$, there exists a size $n-4$ set ${}_QT \subset Q$ such that for all $R, R' \in \mathcal{U}_Q$ and all $t \in T_Q$, $h_R(t) = h_{R'}(t)$.*

We will prove Lemma 3.6 through a series of lemmas. In the following lemma, we consider the characters that are *not* in the set T_Q, that is, the characters $q \in Q$ for which $h_R(q) \neq h_{R'}(q)$.

Lemma 3.7. *For all size $n-2$ sets $Q \subset [k']$ and all $R, R' \in \mathcal{U}_Q$, if $q \in Q$ is such that $h_R(q) \neq h_{R'}(q)$, then either $\pi(R \cup R')[w_R] = q$ or $\pi(R \cup R')[w_{R'}] = q$.*

Proof. Since R and R' are each composed by adding a single character to Q, we have $R \cup R' \in \mathcal{U}_R \cap \mathcal{U}_{R'}$. Since $R \cup R' \in \mathcal{U}_R$, we know that the position of q in $\pi(R \cup R')$ is either $h_R(q)$ or w_R. Since $R \cup R' \in \mathcal{U}_{R'}$, we know that the position of q in $\pi(R \cup R')$ is either $h_{R'}(q)$ or $w_{R'}$. Thus, the position of q in $\pi(R \cup R')$ must be either w_R or $w_{R'}$, because otherwise its position would have to be both $h_R(q)$ and $h_{R'}(q)$, which cannot happen since $h_R(q) \neq h_{R'}(q)$. □

Before proving Lemma 3.6, we prove the pairwise version of Lemma 3.6.

Lemma 3.8 (pairwise version of Lemma 3.6)**.** *For every size $n-2$ set $Q \subset [k']$ of characters, for every pair $R, R' \in \mathcal{U}_Q$, there exists a size $n-4$ subset $T \subset Q$ such that every character $t \in T$, $h_R(t) = h_{R'}(t)$.*

Proof. Suppose by way of contradiction that there exist $R, R' \in \mathcal{U}_Q$ such that there is a set of 3 characters $Q' \subset Q$ so that for each $q \in Q'$, $h_R(q) \neq h_{R'}(q)$. By Lemma 3.7, for each $q \in Q'$ the position of q in $\pi(R \cup R')$ is either w_R or $w_{R'}$. That is, all 3 characters in Q' must occupy a total of 2 positions in $\pi(R \cup R')$, which is impossible. □

We have just shown in Lemma 3.8 that given a size $n-2$ set $Q \subset [k']$, for every pair $R, R' \in \mathcal{U}_Q$ there are at most two characters q in Q for which $h_R(q) \neq h_{R'}(q)$. Thus, we have two cases: 1) the uninteresting case where every pair R, R' has only one such character q, and 2) the interesting case where there exist R, R' so that there are two such characters q. The following lemma handles the uninteresting case by showing that in this case Lemma 3.6 already holds.

Lemma 3.9. *Suppose $Q \subset [k']$ is a size $n-2$ set of characters and for every pair $R, R' \in \mathcal{U}_Q$ there is at most one character $q \in Q$, with $h_R(q) \neq h_{R'}(q)$. Then there exists a size $n-3$ subset $T \subset Q$ such that for every pair $R, R' \in \mathcal{U}_Q$ and every character $t \in T$, $h_R(t) = h_{R'}(t)$.*

Proof. Let $R, R' \in \mathcal{U}_Q$ and $q \in Q$ be such that $h_R(q) \neq h_{R'}(q)$. Then by assumption, for all $q' \in Q$ with $q' \neq q$, $h_R(q') = h_{R'}(q')$. Consider $R'' \in \mathcal{U}_Q$. It suffices to show that for all $q' \in Q$ with $q' \neq q$, we have $h_{R''}(q') = h_R(q')$. Suppose for contradiction that there exists $q' \in Q$ with $q' \neq q$ such that $h_{R''}(q') \neq h_R(q')$. Then, since $h_R(q') = h_{R'}(q')$, we have $h_{R''}(q') \neq h_{R'}(q')$. From the precondition of the lemma statement, q' is the only character with $h_{R''}(q') \neq h_R(q')$ and q' is the only character with $h_{R''}(q') \neq h_{R'}(q')$. Thus, $h_{R''}(q) = h_R(q)$ and $h_{R''}(q) = h_{R'}(q)$. So, $h_R(q) = h_{R'}(q)$, a contradiction. □

We have handled the uninteresting case from above and now we handle the interesting case in which there exist $R, R' \in \mathcal{U}_Q$ so that there are two characters q in Q for which $h_R(q) \neq h_{R'}(q)$. The following lemma shows that in this case we can completely characterize the structure of h_R and $h_{R'}$. Table 1 depicts the structure.

Lemma 3.10. *For every size $n-2$ set $Q \subset [k']$ of characters, if $R, R' \in \mathcal{U}_Q$ are such that there exist $q, q' \in Q$ with $h_R(q) \neq h_{R'}(q)$ and $h_R(q') \neq h_{R'}(q')$, then (modulo switching q and q'):*

1. *Let r be the single character in $R \backslash Q$ and let r' be the single character in $R' \backslash Q$. Then $h_R(q) = h_{R'}(r')$ and $h_R(r) = h_{R'}(q')$.*

2. *$h_R(q') = w_{R'}$ and $h_{R'}(q) = w_R$.*

3. *for all $q'' \in Q$ not equal to q or q', $h_R(q'') = h_{R'}(q'')$.*

			$\boldsymbol{w_R}$	$\boldsymbol{w_{R'}}$	
$\boldsymbol{\pi(R \cup R')}$	r'	r	q	q'	$\{q'' \in Q\}$
$\boldsymbol{h_R}$	q	r	N/A	q'	$\{q'' \in Q\}$
$\boldsymbol{h_{R'}}$	r'	q'	q	N/A	$\{q'' \in Q\}$

Table 1: The structure imposed by Lemma 3.10. Each column indicates an index in $[n]$. For example, the first column indicates that $\pi(R \cup R')[r'] = h_R(q) = h_{R'}(r')$. Some entries are not applicable (N/A) because by definition h_R does not map anything to the index w_R.

Proof. We begin with item 3. By Lemma 3.7, without loss of generality $\pi(R \cup R')[w_R] = q$ and $\pi(R \cup R')[w_{R'}] = q'$. Since the position in $\pi(R \cup R')$ of each remaining character $q'' \in Q$ is either $h_R(q'')$ or w_R but the position w_R is taken by q, it must be that $\pi(R \cup R')[h_R(q'')] = q''$. Similarly, we have $\pi(R \cup R')[h_{R'}(q'')] = q''$. Thus, $h_R(q'') = h_{R'}(q'')$ for all $q'' \in Q$ with $q'' \neq q, q'$.

We now move to item 2. Since the position of q in $\pi(R \cup R')$ is either $w_{R'}$ or $h_{R'}(q)$ but $w_{R'}$ is taken by q', we have that $\pi(R \cup R')[h_{R'}(q)] = q$. We already know that $\pi(R \cup R')[w_R] = q$, so $h_{R'}(q) = w_R$. By a symmetric argument, $h_R(q') = w_{R'}$.

We now move to item 1. Since the position in $\pi(R \cup R')$ of r is either $h_R(r)$ or w_R, but the position w_R is taken by q, it must be that $\pi(R \cup R')[h_R(r)] = r$. By a symmetric argument, $\pi(R \cup R')[h_{R'}(r')] = r'$. Combining these two facts, since r and r' cannot occupy the same index in $\pi(R \cup R')$, we have $h_R(r) \neq h_{R'}(r')$. We proceed by process of elimination.

The set of indices which are mapped to by h_R is $[n] \setminus \{w_R\}$ and the indices which have so far been mapped to by items 2 and 3 are $w_{R'}$ and $h_R(q'') = h_{R'}(q'')$ for all $q'' \in Q$ not equal to q or q'. The set of indices which are mapped to by $h_{R'}$ is $[n] \setminus \{w_{R'}\}$ and the indices which have so far been mapped to by items 2 and 3 are w_R and $h_{R'}(q'') = h_R(q'')$ for all $q'' \in Q$ with $q'' \neq q, q'$. Thus, the set of indices which have not yet been mapped to is the same for h_R and $h_{R'}$: $[n] \setminus \{w_R, w_{R'}, h_R(q'') = h_{R'}(q'')\}$. The characters for which h_R has not yet been determined are q and r and the characters for which $h_{R'}$ has not yet been determined are q' and r'. From the previous paragraph, we know that $h_R(r) \neq h_{R'}(r')$. Thus, we have $h_R(q) = h_{R'}(r')$ and $h_R(r) = h_{R'}(q')$. ☐

We are now ready to prove Lemma 3.6. We have just shown in Lemma 3.10 that individual pairs $R, R' \in \mathcal{U}_Q$ obey a particular structure, and in Lemma 3.6 we will derive structure among *all* $R \in \mathcal{U}_Q$.

Lemma 3.11 (Restatement of Lemma 3.6). *For every size $n-2$ set $Q \subset [k']$, there exists a size $n-4$ set $T_Q \subset Q$ such that for all $R, R' \in \mathcal{U}_Q$ and all $t \in T_Q$, $h_R(t) = h_{R'}(t)$.*

Proof. By Lemma 3.9 we can assume that there exists a pair $R, R' \in \mathcal{U}_Q$ so that there exist $q, q' \in Q$ with $h_R(q) \neq h_{R'}(q)$ and $h_R(q') \neq h_{R'}(q')$. Fix R, R', q, and q'. R and R' obey the structure specified by Lemma 3.10. Consider $R'' \in \mathcal{U}_Q$ with $R'' \neq R, R'$. Suppose by way of contradiction that there exists $q'' \in Q$ with $q'' \neq q, q'$ such that $h_{R''}(q'') \neq h_R(q'')$ (and thus also $h_{R''}(q'') \neq h_{R'}(q'')$ since $h_R(q'') = h_{R'}(q'')$ by Lemma 3.10).

We first note that it cannot be the case that both $h_{R''}(q) = h_R(q)$ and $h_{R''}(q') = h_R(q')$ because then we would have $h_{R''}(q) \neq h_{R'}(q)$ and $h_{R''}(q') \neq h_{R'}(q')$, in which case $h_{R''}$ and $h_{R'}$ would differ on inputs q, q', and q'' which contradicts Lemma 3.8. Thus, $h_{R''}$ must differ from each of h_R and $h_{R'}$ and on exactly one of q or q'. Without loss of generality, suppose $h_{R''}(q) \neq h_R(q)$ and $h_{R''}(q') \neq h_{R'}(q')$.

Since $h_{R''}$ also differs from each of h_R and $h_{R'}$ on input q'', $h_{R''}$ differs from each of h_R and $h_{R'}$ on exactly two inputs. Thus, the pair R'', R and the pair R'', R' both obey the structure specified by Lemma 3.10. We claim that it is impossible to reconcile these pairwise structural constraints.

R, R', and R'' are each composed by adding a single character to Q. Let r, r', and r'' be these characters respectively. Applying item 1 of Lemma 3.10 to the pair R, R' we have the following two cases:

Case 1: $h_R(q) = h_{R'}(r')$ **and** $h_R(r) = h_{R'}(q')$**.** Item 1 of Lemma 3.10 presents two options for the pair R, R'': either $h_{R''}(r'') = h_R(q)$ or $h_{R''}(r'') = h_R(q'')$. If $h_{R''}(r'') = h_R(q)$, then from the definition of case 1, $h_{R'}(r') = h_{R''}(r'')$, but this is not true by item 1 of Lemma 3.10. Thus, $h_{R''}(r'') = h_R(q'')$ and $h_R(r) = h_{R''}(q)$. Since $h_{R''}$ and $h_{R'}$ differ only on inputs q' and q'', we have $h_{R''}(q) = h_{R'}(q)$. Thus, we have shown that $h_R(r) = h_{R'}(q)$. However, by item 2 of Lemma 3.10, we have $h_{R'}(q) = w_R$, which is a contradiction since $h_R(r) \neq w_R$.

Case 2: $h_R(q') = h_{R'}(r')$ **and** $h_R(r) = h_{R'}(q)$**.** Since $h_{R''}$ and h_R differ only on inputs q and q'', we have $h_{R''}(q') = h_R(q')$. Thus, $h_{R''}(q') = h_{R'}(r')$. Then by item 2 of Lemma 3.10, we have $h_{R''}(q'') = w_{R'}$. Since $h_{R''}$ and $h_{R'}$ differ only on inputs q' and q'', we have $h_{R''}(q) = h_{R'}(q)$. Then since we are in case 2, we have $h_R(r) = h_{R''}(q)$. Then by item 2 of Lemma 3.10, we have $h_{R''}(q'') = w_R$. Thus, we have shown that $h_{R''}(q'')$ is equal to both $w_{R'}$ and w_R, which is not true by Lemma 3.10. ☐

By Lemma 3.11, we can define a function h'_Q that takes as input any element $t \in T_Q$ and outputs the value $h_R(t)$, which is the same for all $R \in \mathcal{U}_Q$.

Let $P \subset [k']$ be a size $n-3$ set. Recall that $\mathcal{U}_P$ is the set of all size $n-2$ sets Q such that $P \subset Q \subset [k']$. We conclude this section by proving a lemma similar to Lemma 3.5, which says that the functions h' for any two sets in $\mathcal{U}_P$ are *consistent*.

Lemma 3.12. *For every size $n-3$ set $P \subset [k']$, for any pair $Q, Q' \in \mathcal{U}_P$, for any character $t \in T_Q \cap T_{Q'}$, $h'_Q(t) = h'_{Q'}(t)$.*

Proof. Since Q and Q' are each composed by adding a single character to P, we have $Q \cup Q' \in \mathcal{U}_Q \cap \mathcal{U}_{Q'}$. Since $Q \cup Q' \in \mathcal{U}_Q$, we know that $h'_Q(t) = h_{Q\cup Q'}(t)$ and since $Q \cup Q' \in \mathcal{U}_{Q'}$, we know that $h'_{Q'}(t) = h_{Q\cup Q'}(t)$. Thus, $h'_Q(t) = h'_{Q'}(t)$. ☐

3.3.6 Step 3b: Counting argument for configuration 2

Like the previous section, in this section we will assume that all size $n-1$ subsets of K' obey configuration 2. The counting argument similar to that from step 3a.

By Lemma 3.12, for every size $n-3$ set $P \subset [k']$, we have that all $Q \in \mathcal{U}_P$ agree on the value of $h'_Q(t)$ if it exists. Thus, we can define H_P as the union of h'_Qs over all $Q \in \mathcal{U}_P$. Formally, $H_P(t) = i$ if for every $Q \in \mathcal{U}_P$ with $t \in T_Q$, we have $h'_Q(t) = i$. We note that $H_P(t)$ *exists* if for *some* $Q \in \mathcal{U}_P$, t is in the set T_Q.

Since $|P| = n-3$ and there are n indices total, $H_P(t)$ can exist for at most $n-3$ characters $t \in P$ and at most 3 characters $t \notin P$. We say that the pair (P, t) is *irregular* if $H_P(t)$ exists and $t \notin P$. The quantity that we will count is the total number of irregular pairs (P, t) over all size $n-3$ sets $P \subset [k']$ and all $t \in [k']$.

On one hand, as previously mentioned, each set P can only be in at most 3 irregular pairs. Then since there are $\binom{k'}{n-3}$ sets $P \subset [k']$ of size $n-3$, the total number of irregular pairs is at most $3\binom{k'}{n-3}$.

On the other hand, Lemma 3.6 implies a lower bound on the number of irregular pairs. By Lemma 3.6, for *every* size $n-2$ set $Q \subset [k']$, the set $T_Q \subset Q$ is of size $n-4$. Fix sets Q and T_Q. We claim that for each $t \in T_Q$, the pair $(Q \setminus \{t\}, t)$ is an irregular pair. Firstly, it is clear that $t \notin Q \setminus \{t\}$. Secondly, $H_{Q\setminus\{t\}}(t)$ exists because $Q \in \mathcal{U}_{Q\setminus\{q\}}$ and $t \in T_Q$. Thus, for each $t \in T_Q$, the pair $(Q \setminus \{t\}, t)$ is an irregular pair.

Thus, every size $n-2$ set $Q \subset [k']$ *produces* $n-4$ irregular pairs $(Q \setminus \{t\}, t)$. Furthermore, given an irregular pair (P, t), there is only one set that could produce it, namely $P \cup \{t\}$. Then since there are $\binom{k'}{n-2}$ sets $Q \subset [k']$ of size $n-2$, we have that the total number of irregular pairs is at least $(n-4)\binom{k'}{n-2}$.

Thus, we have shown that the total number of irregular pairs is at most $3\binom{k'}{n-3}$ and at least $(n-4)\binom{k'}{n-2}$. Therefore, we have reached a contradiction if $3\binom{k'}{n-3} < (n-4)\binom{k'}{n-2}$ which is true if $n \geq 5$ and $k' > \frac{n^2-4n+6}{n-4}$. In particular, $n \geq 5$, $k' \geq n+7$ satisfy these bounds. $\square$

4 The remaining parameter regime

Theorem 4.1 (restatement of Theorem 1.1)**.** *For $n \geq 3$, $k \geq 5$, every set of functions $f_1^{n,k}, \ldots, f_n^{n,k}$ has maximum switching cost at least 3.*

Remark. We note that the proof framework from Section 3 immediately breaks down if we try to apply it to Theorem 4.1 for all n, k. For example, when $n > k$, there are no size n subsets of $[k]$ so we must instead consider size n *multisets* of $[k]$. Even if we have the same setting of parameters as Theorem 2.1 but we are considering multisets, in step 1 of the proof framework Lemma 3.1 is no longer true. That is, it is not true that for all size 2 multisets $\{x, y\}$ of $[k]$, we have that $\{x, y\}$ i-freezes either x or y for some i. In particular, suppose $\{x, y\} = \{a, a\}$. Then if is possible that $\pi(\{a, a, b\}) = aab$, $\pi(\{a, a, c\}) = aca$, and $\pi(\{a, a, d\}) = daa$, in which case a is not frozen to any index. Since the proof framework from Section 3 no longer applies, we develop entirely new techniques in this section.

For the rest of this section we will use the language of the original problem statement rather than that of the problem reformulation.

4.1 Preliminaries

To prove the Theorem 4.1, we need to show that Theorem 3.1 extends to larger k and n. As noted in Section 1.4.2, extending to larger n is challenging, while extending to larger k is trivial, as shown in the following lemma.

Lemma 4.1. *Fix n and k. If there exists a set of functions $f_1^{n,k}, \ldots, f_n^{n,k}$ with maximum switching cost D, then for all $k' < k$, there exists a set of functions $g_1^{n,k'}, \ldots, g_n^{n,k'}$ with maximum switching cost D.*

Proof. For each demand vector $\vec{v}$ with n agents and k tasks such that only the first k' entries of $\vec{v}$ are non-zero, let $\vec{v'}$ be the length k' vector consisting of only the first k' entries of $\vec{v}$. We note that the set of all such vectors $\vec{v'}$ is the set of all demand vectors for n agents and k' tasks. Set each $g_i^{n,k'}(\vec{v'}) = f_i^{n,k}(\vec{v})$. Then the switching cost for any adjacent pair $(\vec{v'_1}, \vec{v'_2})$ with respect to $g_1^{n,k'}, \ldots, g_n^{n,k'}$ is equal to the switching cost of the corresponding adjacent pair $(\vec{v_1}, \vec{v_2})$ with respect to $f_1^{n,k}, \ldots, f_n^{n,k}$. Thus, the maximum switching cost of $g_1^{n,k'}, \ldots, g_n^{n,k'}$ is equal to the maximum switching cost of $f_1^{n,k}, \ldots, f_n^{n,k}$. $\square$

Notation. We say that an ordered pair of adjacent demand vectors $(\vec{v_1}, \vec{v_2})$ is (s, t)-adjacent if starting with $\vec{v_1}$ and moving exactly one unit of demand from task s to task t results in $\vec{v_2}$. We say that an agent a is (i, j)-*mobile* with respect to an ordered pair of adjacent demand vectors $(\vec{v_1}, \vec{v_2})$ if $f_a^{n,k}(\vec{v_1}) = i$, $f_a^{n,k}(\vec{v_2}) = j$, and $i \neq j$.

We note that if $(\vec{v_1}, \vec{v_2})$ is (s, t)-adjacent and has switching cost 2, then for some task i, some agent a must be (s, i)-mobile and another agent b must be (i, t)-mobile. We say that i is the *intermediate* task with respect to $(\vec{v_1}, \vec{v_2})$.

4.2 Proof overview

We begin by supposing for contradiction that there exists a set of functions $f_1^{n,k}, \ldots, f_n^{n,k}$ with maximum switching cost 2, and then we prove a series of structural lemmas about such functions.

As previously mentioned, the main challenge of proving Lemma 4.1 is handling large n. To illustrate this challenge, we repeat the example from Section 1.4.2. This example shows that having large n can allow more pairs of adjacent demand vectors to have switching cost 2, making it more difficult to find a pair with switching cost greater than 2.

Consider the subset S_i of demand vectors in which a particular task i has an unconstrained amount of demand and each remaining task has demand at most $n/(k-1)$. We claim that there exists a set of functions $f_1^{n,k}, \ldots, f_n^{n,k}$ so that every pair of adjacent demand vectors from S_i has switching cost 2. Divide the agents into $k-1$ groups of $n/(k-1)$ agents each, and associate each task except i to such a group of agents. We define the functions $f_1^{n,k}, \ldots, f_n^{n,k}$ so that given any demand vector in S_i, the set of agents assigned to each task except i is simply a subset of the group of agents associated with that task (say, the subset of such agents with smallest ID). This is a valid assignment since the demand of each task except i is at most the size of the group of agents associated with that task. The remaining agents are assigned to task i. Then, given a pair $(\vec{v}, \vec{v'})$ of adjacent demand vectors in S_i, whose demands differ only for tasks s and t, their switching cost is 2 because the only agents assigned to different tasks between $\vec{v}$ and $\vec{v'}$ are: one agent from each of the groups associated with tasks s and t, respectively.

To overcome the challenge illustrated by the above example, our general method is to identify a task that serves the role of task i and then successively move demand out of task i until task i is empty, and thus can no longer serve its original role. We note that in the above example, the task i serves as the intermediate task for all pairs of adjacent demand vectors from S_i. Thus, we will choose i to be an intermediate task.

In particular, we show that there is a demand vector $\vec{v}$ so that we can identify tasks i and t with the following important property: if we start with $\vec{v}$ and move a unit of demand to task t from *any* other task except i, the switching cost is 2 and the intermediate task is i.

Furthermore, we prove that if we start with demand vector $\vec{v}$ and move a unit of demand from task i to task t resulting in demand vector $\vec{v_1}$, then t and i have the important property from the previous paragraph with respect to $\vec{v_1}$. Applying this argument inductively, we show that no matter how many units of demand we successively move from i to t, i and t still satisfy the important property with respect to the current demand vector.

We move demand from i to t until task i is empty. Then, the final contradiction comes from the fact that if we now move a unit of demand from any non-i task to t, then the important property implies that the switching cost is 2 and the intermediate task is i; however, i is empty and an empty task cannot serve as an intermediate task.

4.3 Proof of Theorem 4.1

Theorem 3.1 proves Theorem 4.1 for the case of $n = 3$ and $k = 5$. Lemma 4.1 implies that Theorem 4.1 also holds for $n = 3$ and any $k \geq 5$. Thus, it remains to prove Theorem 4.1 for $n \geq 4$ and $k \geq 5$. Suppose by way of contradiction that $n \geq 4$, $k \geq 5$, and $f_1^{n,k}, f_2^{n,k}, \ldots, f_n^{n,k}$ is a set of functions with switching cost 2.

As motivated in the algorithm overview, our first structural lemma concerns tasks i and t such that if we move a unit of demand to task t from any other task except i, the switching cost is 2 and the intermediate task is i.

Lemma 4.2. *Let $(\vec{v}, \vec{v_1})$ be a pair of (s_1, t)-adjacent demand vectors with switching cost 2 and intermediate task i. Then, for all $\vec{v_2}$ such that $(\vec{v}, \vec{v_2})$ are (s_2, t)-adjacent for $s_2 \neq i$, the pair $(\vec{v}, \vec{v_2})$ has switching cost 2,* *intermediate task i, and the same (i,t)-mobile agent as $(\vec{v},\vec{v_1})$.*

Proof. Table 2 depicts the proof.

	s_1	i	t	s_2
$\vec{v}$	a	b		c
$\vec{v_1}$		a	b	c
$\vec{v_2}$ **(case 1)**	a	b	c	
$\vec{v_2}$ **(case 2)**		b	d	not c

Table 2: Demand vectors and the corresponding assignment of agents. For example, the row labeled $\vec{v}$ indicates that for the demand vector $\vec{v}$, agent a is assigned to task s_2, agent b is assigned to task i, and agent c is assigned to task s_2. There could also be other agents in the system that are not shown in the table.

With respect to $(\vec{v},\vec{v_1})$, let a be the (s_1,i)-mobile agent and let b be the (i,t)-mobile agent. Then a and b behave according to rows $\vec{v}$ and $\vec{v_1}$ of Table 2.

Suppose by way of contradiction that $(\vec{v},\vec{v_2})$ is *not* as in the lemma statement. That is, either $(\vec{v},\vec{v_2})$ has switching cost 1 or $(\vec{v},\vec{v_2})$ has switching cost 2 and either a different intermediate task from $(\vec{v},\vec{v_1})$ or a different (i,t)-mobile agent.

Case 1. $(\vec{v},\vec{v_2})$ **has switching cost 1.** Let c be the mobile agent with respect to $(\vec{v},\vec{v_2})$. Then, $\vec{v_2}$ is as in row $\vec{v_2}$ (case 1) of Table 2. Also, since c is not mobile with respect to $(\vec{v},\vec{v_1})$, c is assigned to s_2 for both $\vec{v}$ and $\vec{v_1}$ as shown in Table 2. Comparing rows $\vec{v_1}$ and $\vec{v_2}$ (case 1) of Table 2, it is clear that $(\vec{v_1},\vec{v_2})$ are adjacent and have switching cost 3, since a, b, and c each switch tasks. This is a contradiction.

Case 2. $(\vec{v},\vec{v_2})$ **has switching cost 2.** Let i_2 be the intermediate task of $(\vec{v},\vec{v_2})$ and let c be the (s_2,i_2)-mobile agent for $(\vec{v},\vec{v_2})$. Then, for $\vec{v_2}$, c is not assigned to s_2, as shown in Table 2. Let d be the (i_2,t)-mobile agent for $(\vec{v},\vec{v_2})$. We note that it is possible that $d=a$, however $d \neq b$ since d is assigned to i_2 for $\vec{v}$ while b is assigned to i, and $i \neq i_2$. Table 2 shows the positions of b and d (but not a) in $\vec{v_2}$. Comparing rows $\vec{v_1}$ and $\vec{v_2}$ (case 2) of Table 2, it is clear that $(\vec{v_1},\vec{v_2})$ has switching cost 3, since b, d, and c each switch tasks. Since $(\vec{v_1},\vec{v_2})$ are adjacent, this is a contradiction. □

We have just shown in Lemma 4.2 that with respect to any demand vector $\vec{v}$, the set of tasks can be split into two distinct types such that every task is of exactly one type.

Definition 4.1 (type 1 task)**.** A task t is of *type 1* with respect to a demand vector $\vec{v}$ if when we start with $\vec{v}$ and move a unit of demand from any task to task t, the switching cost is 1.

Definition 4.2 (type 2 task)**.** A task t is of *type 2* with respect to a demand vector $\vec{v}$ if there exists a task i and an agent a such that when we start with $\vec{v}$ and move a unit of demand from any task except i to task t, the switching cost is 2, the intermediate task is i, and the (i,t)-mobile agent is a. We say that a is the *intermediate agent* of t with respect to $\vec{v}$.

Remark. We note that if $\vec{v}$ only has two non-empty tasks besides t, then it is possible that the identity of task i is ambiguous. However, every time we reference a type 2 task we always have the condition that there are at least three non-empty tasks besides t so there will be no ambiguity.

As mentioned in the proof overview we wish to successively move demand out of an intermediate task until it is empty. The bulk of the remainder of the proof is to prove the following lemma (Lemma 4.3), which roughly says that if t is a type 2 task and i is t's intermediate task, then after we move a unit of demand from task i to task t, task t remains a type 2 task with intermediate task i. Then, by iterating Lemma 4.3, we show that after moving any amount of demand from task i to task t, task t still remains a type 2 task with intermediate task i.

Lemma 4.3. *Let $\vec{v}$ be a demand vector with at least four non-zero entries. Then there exists a task t such that t is of type 2 with respect to $\vec{v}$ and $\vec{v}$ has at least four non-empty tasks distinct from t. Let i be the intermediate task of t with respect to $\vec{v}$. Let $\vec{v'}$ be such that $(\vec{v}, \vec{v'})$ are (i, t)-adjacent. Then, t is a type 2 task with intermediate task i with respect to $\vec{v'}$.*

Lemma 4.3 implies Theorem 4.1. Let $\vec{v}$, t, and i be as in Lemma 4.3. We claim that if task i is non-empty in $\vec{v'}$ then the triple ($\vec{v'}$, t, i) also satisfies the precondition of Lemma 4.3. This is because if task i is non-empty in $\vec{v'}$ then the set of non-empty tasks in $\vec{v'}$ is a superset of the set of non-empty tasks in $\vec{v'}$. Then since $\vec{v}$ has at least four non-empty tasks distinct from t, $\vec{v'}$ also has at least four non-empty tasks distinct from t. Also, by Lemma 4.3, t is a type 2 task with intermediate task i with respect to $\vec{v'}$. Thus, we have shown that if task i is non-empty for $\vec{v'}$ then ($\vec{v'}$, t, i) satisfy the precondition of Lemma 4.3. Thus, we can iterate Lemma 4.3: if we start with $\vec{v}$ and successively move demand from task i to task t until task i is empty, the resulting demand vector $\vec{v''}$ is such that t is a type 2 task with intermediate task i. However, it is impossible for i to be an intermediate task with respect to $\vec{v''}$ since i is empty. It remains to prove Lemma 4.3.

4.3.1 Proof of Lemma 4.3

The following lemma shows that the pair $(\vec{v}, \vec{v'})$ from the statement of Lemma 4.3 has switching cost 1.

Lemma 4.4. *Let t be a type 2 task with intermediate task i with respect to a demand vector $\vec{v}$. Suppose $\vec{v}$ has at least one unit of demand in each of two tasks s_1 and s_2, both distinct from t and i. Let $\vec{v'}$ be the demand vector such that $(\vec{v}, \vec{v'})$ is (i, t)-adjacent. Then $(\vec{v}, \vec{v'})$ has switching cost 1.*

Proof. Table 3 depicts the proof.

	s_1	i	t	s_2
$\vec{v}$	a	b	not d	c
$\vec{v_1}$		a	b	c
$\vec{v_2}$	a	c	b, not d	
$\vec{v'}$ **(case 1)**			a, not b	c
$\vec{v'}$ **(case 2)**	a		d, not b	c

Table 3: Demand vectors and the corresponding assignment of agents.

Let $\vec{v_1}$ be such that $(\vec{v}, \vec{v_1})$ is (s_1, t)-adjacent and let $\vec{v_2}$ be such that $(\vec{v}, \vec{v_2})$ is (s_2, t)-adjacent. With respect to $(\vec{v}, \vec{v_1})$, let a be the (s_1, i)-mobile agent and let b be the (i, t)-mobile agent. Then a and b behave according to rows $\vec{v}$ and $\vec{v_1}$ of Table 2.

With respect to $(\vec{v}, \vec{v_2})$, let c be the (s_2, i)-mobile agent. From Lemma 4.2 we know that b is the (i, t)-mobile agent for $(\vec{v}, \vec{v_2})$. Thus b and c behave according to rows $\vec{v}$ and $\vec{v_2}$ of Table 2.

Suppose by way of contradiction that $(\vec{v}, \vec{v'})$ has switching cost 2. Let $i' \neq i, t$ be the intermediate task. We condition on the (i', t)-mobile agent. We already know that it is not b since b is assigned to i for $\vec{v}$.

Case 1. the (i', t)-mobile agent for $(\vec{v}, \vec{v'})$ is a or c. Suppose the (i', t)-mobile agent for $(\vec{v}, \vec{v'})$ is a, as shown in row $\vec{v'}$ (case 1) of Table 3. If the (i', t)-mobile agent is c, the argument is identical. Since we are assuming c is not a mobile-agent, c is assigned to s_2 in $\vec{v'}$ as shown in Table 3. Also, since a is the only

(i', t)-mobile agent, we know that b is not assigned to t in $\vec{v'}$ as shown in Table 3. Comparing rows $\vec{v_2}$ and $\vec{v'}$ (case 1) of Table 3, it is clear that $(\vec{v_2}, \vec{v'})$ have switching cost 3, since a, b, and c all switch tasks. Also, $\vec{v_2}$ and $\vec{v'}$ are adjacent since both are the result of starting with $\vec{v}$ and moving one unit of demand from some task to task t. This is a contradiction.

Case 2. the (i', t)-mobile agent for $(\vec{v}, \vec{v'})$ is neither a nor c. Let d be the (i', t)-mobile agent for $(\vec{v}, \vec{v_3})$. Then d, a, and c are assigned as in row $\vec{v'}$ (case 2) of Table 3. Also, since d is the only (i', t)-mobile agent and $d \neq b$, we know that b is not assigned to t in $\vec{v'}$, as shown in Table 3.

Also, since d is the (i', t)-mobile agent for $(\vec{v}, \vec{v'})$, we know that d is not assigned to t in $\vec{v}$, as shown in Table 3. Then, since b is the only (i, t)-mobile agent for $(\vec{v}, \vec{v_2})$, we know that d is also not assigned to t for $\vec{v_2}$, as shown in Table 3.

Comparing rows $\vec{v_2}$ and $\vec{v'}$ (case 2) of Table 3, it is clear that $(\vec{v_2}, \vec{v_3})$ have switching cost 3, since d, b, and c all switch tasks. This is a contradiction since $\vec{v_2}$ and $\vec{v'}$ are adjacent. □

Next, we prove another structural lemma concerning the intermediate task i, which says that task i is of type 1.

Lemma 4.5. *Let $\vec{v}$ be a demand vector with at least four non-zero entries and let t be a type 2 task with intermediate task i. Then, task i is of type 1 with respect to $\vec{v}$.*

Proof. Table 4 depicts the proof.

	s	i	t
$\vec{v}$	a	b, not c	
$\vec{v_1}$		a, not c	b
$\vec{v_2}$		b, c, not a	

Table 4: Demand vectors and the corresponding assignment of agents.

Suppose for contradiction that task i is of type 2 with respect to $\vec{v}$, and let i' be the intermediate task. Since $\vec{v}$ has at least four non-zero entries, there exists a task $s \neq i'$ that is non-empty for $\vec{v}$. Let $\vec{v_1}$ be such that $(\vec{v}, \vec{v_1})$ are (s, t)-adjacent. For $(\vec{v}, \vec{v_1})$, let a be the (s, i)-mobile agent and let b be the (i, t)-mobile agent, as shown in Table 4.

Letting $\vec{v_2}$ be such that $(\vec{v}, \vec{v_2})$ are (s, i)-adjacent, $(\vec{v}, \vec{v_2})$ has switching cost 2 since $s \neq i'$. Thus, a does *not* switch to task i for $(\vec{v}, \vec{v_2})$, as shown in Table 4. Also, b remains in task i for $(\vec{v}, \vec{v_2})$, as shown in Table 4. Let c be the mobile agent for $(\vec{v}, \vec{v_2})$ that switches to task i. Then agent c is not assigned to task i with respect to $\vec{v}$ or $\vec{v_1}$ and is assigned to task i with respect to $\vec{v_2}$, as shown in Table 4.

We note that $(\vec{v_1}, \vec{v_2})$ are (t, i)-adjacent while they differ on the assignment of agents a, b, and c, a contradiction. □

Later, we will prove the following lemma (Lemma 4.6), which says that (under certain conditions) there is *at most one* task of type 1. Combining this with Lemma 4.5 allows us to say that every task except for intermediate task i is of type 2, which will be a useful structural property.

Lemma 4.6. *For any demand vector $\vec{v}$ with at least four non-zero entries, there is at most one task of type 1.*

In order to prove Lemma 4.6, we will prove two structural lemmas. The following simple lemma is useful (but may at first appear unrelated).

Lemma 4.7. *Let $(\vec{v}, \vec{v_1})$ be a pair of (s, t_1)-adjacent demand vectors with switching cost 2 and intermediate task i_1. Let $(\vec{v}, \vec{v_2})$ be a pair of (s, t_2)-adjacent demand vectors with switching cost 2 and intermediate task i_2. Then it is* not *the case that s, i_1, t_1, i_2, and t_2 are all distinct.*

Proof. Suppose by way of contradiction that s, i_1, t_1, i_2, and t_2 are all distinct. Table 5 depicts the proof.

	s	i_1	t_1	i_2	t_2
$\vec{v}$	a	b		c	
$\vec{v_1}$		a	b	c	
$\vec{v_2}$		b, not a			c

Table 5: Demand vectors and the corresponding assignment of agents.

With respect to $(\vec{v}, \vec{v_1})$, let a be the (s, i_1)-mobile agent and let b be the (i_1, t_1)-mobile agent. Then a and b behave according to rows $\vec{v}$ and $\vec{v_1}$ of Table 5.

With respect to $(\vec{v}, \vec{v_2})$, let c be the (i_2, t_2)-mobile agent. Then c behaves according to Table 5. Since $i_1 \neq s, i_2, t_2$, we know that b is in the same position in $\vec{v}$ and $\vec{v_2}$. For the same reason, a does not move to i_1 with respect to $(\vec{v}, \vec{v_2})$, as shown in Table 5.

Comparing rows $\vec{v_1}$ and $\vec{v_2}$ in Table 5, it is clear that $(\vec{v_1}, \vec{v_2})$ have switching cost 3. Since $\vec{v_1}$ and $\vec{v_2}$ are adjacent, this is a contradiction. □

The following lemma is a weaker version of Lemma 4.6, which says that there is at least one task of type 2.

Lemma 4.8. *For any demand vector $\vec{v}$ with at least two non-zero entries, there is at least one task of type 2.*

Proof. Table 6 depicts the proof.

	s_1	s_2	s_4	s_5
$\vec{v}$	a	b		
$\vec{v_1}$		b	a	
$\vec{v_2}$	a			b
$\vec{v_3}$	a		b	
$\vec{v_4}$		b		a
$\vec{v_5}$ (case 1)			a	b
$\vec{v_5}$ (case 2)			b	a

Table 6: Demand vectors and the corresponding assignment of agents.

Suppose by way of contradiction that every task is of type 1 with respect to $\vec{v}$. That is, for all $\vec{v'}$ adjacent to $\vec{v}$, the pair $(\vec{v}, \vec{v'})$ has switching cost 1. Let s_1, s_2, and s_3 be non-empty tasks with respect to $\vec{v}$ (s_3 is not shown in Table 6). Let s_4 and s_5 be additional tasks. Let $\vec{v_1}$ be such that $(\vec{v}, \vec{v_1})$ are (s_1, s_4)-adjacent with mobile agent a. Let $\vec{v_2}$ be such that $(\vec{v}, \vec{v_2})$ are (s_2, s_5)-adjacent with mobile agent b. The assignments of agents a, b, and c for vectors $\vec{v}$, $\vec{v_1}$, and $\vec{v_2}$ are shown in Table 6.

Let $\vec{v_3}$ be such that $(\vec{v}, \vec{v_3})$ are (s_2, s_4)-adjacent. We know that $(\vec{v}, \vec{v_3})$ has switching cost 1 but we do not know whether the mobile agent is b or some other agent. Similarly, let $\vec{v_4}$ be such that $(\vec{v}, \vec{v_4})$ are (s_1, s_5)-adjacent. We know that $(\vec{v}, \vec{v_4})$ has switching cost 1 but we do not know whether the mobile agent is a or some other agent. We claim that the mobile agent for $(\vec{v}, \vec{v_4})$ is a, and symmetrically the mobile agent for $(\vec{v}, \vec{v_3})$ is b, as shown in Table 6.

Suppose for contradiction that the mobile agent for $(\vec{v}, \vec{v_4})$ is some agent $d \neq a$. Let $\vec{v_5}$ be such that $(\vec{v_1}, \vec{v_5})$ are (s_2, s_5)-adjacent. Note that $\vec{v_5}$ is adjacent to both $\vec{v_4}$ and $\vec{v_2}$. $\vec{v_2}$ places agent b (and not d) in task s_5, and $\vec{v_4}$ places agent d (and not b) in task s_5. Then, since $\vec{v_5}$ can only place at most one of b or d in task s_5, $\vec{v_5}$ must disagree with either $\vec{v_2}$ or $\vec{v_4}$ on the assignment of *both* b and d. Also, $\vec{v_1}$ places agent a in task s_4 while both $\vec{v_2}$ and $\vec{v_4}$ place agent a in task s_1. Since $(\vec{v_1}, \vec{v_5})$ are (s_2, s_5)-adjacent and have switching cost at most 2, agent a cannot switch from task s_4 to task s_1 when the demand vector changes from $\vec{v_1}$ to $\vec{v_5}$. Thus, $\vec{v_5}$ disagrees with both $\vec{v_2}$ and $\vec{v_4}$ on the assignment of agent a. Thus, we have shown that $\vec{v_5}$ disagrees with either $\vec{v_2}$ or $\vec{v_4}$ on the assignment of a, b, and d, a contradiction. Therefore, the mobile agent for $(\vec{v}, \vec{v_4})$ is a, as shown in Table 6. By the same argument, the mobile agent for $(\vec{v}, \vec{v_3})$ is b, as shown in Table 6.

Now, consider $(\vec{v_1}, \vec{v_5})$, which are (s_2, s_5)-adjacent. We first claim that no agents besides a and b are mobile for $(\vec{v_1}, \vec{v_5})$. Like the previous paragraph, Note that $\vec{v_5}$ is adjacent to both $\vec{v_4}$ and $\vec{v_2}$. Again, $\vec{v_2}$ places agent b (and not a) in task s_5 and $\vec{v_4}$ places agent a (and not b) in task s_5. Then, since $\vec{v_5}$ can only place at most one of a or b in task s_5, $\vec{v_5}$ must disagree with either $\vec{v_2}$ or $\vec{v_4}$ on the assignment of *both* a and b. Then since $\vec{v_1}$, $\vec{v_2}$, and $\vec{v_4}$ all agree on the assignment of every agent except a and b, no other agent c can be mobile for $(\vec{v_1}, \vec{v_5})$, because then $\vec{v_5}$ would disagree with either $\vec{v_2}$ or $\vec{v_4}$ on the placement of agents a, b, and c. Thus, no agents besides a and b are mobile for $(\vec{v_1}, \vec{v_5})$.

Therefore, we have two cases:

Case 1: $(\vec{v_1}, \vec{v_5})$ **has switching cost 1.** In this case the only mobile agent for $(\vec{v_1}, \vec{v_5})$ is b as indicated in row $\vec{v_5}$ (case 1) of Table 6.

Table 6 shows that $(\vec{v_5}, \vec{v_4})$ has switching cost 2 where a is (s_4, s_5)-mobile and b is (s_5, s_2)-mobile. Thus, with respect to $\vec{v_5}$, task s_2 is of type 2 with intermediate task s_5.

Similarly, Table 6 shows that $(\vec{v_5}, \vec{v_3})$ has switching cost 2 where b is (s_5, s_4)-mobile and a is (s_4, s_1)-mobile. Thus, with respect to $\vec{v_5}$, task s_1 is of type 2 with intermediate task s_4.

We observe that the combination of the previous two paragraphs violates Lemma 4.7. Recall that s_3 is a task that is non-empty for $\vec{v}$, and thus also $\vec{v_5}$. We apply Lemma 4.7 with parameters $(\vec{v}, s, t_1, t_2) = (\vec{v_5}, s_3, s_2, s_1)$. Then, from the previous two paragraphs, the intermediate task for task s_2 is task s_5 and the intermediate task for task s_1 is task s_4, so i_1 and i_2 are tasks s_5 and s_4, respectively. Then s, t_1, t_2, i_1, and i_2 are all distinct, which violates Lemma 4.7.

Case 2: $(\vec{v_1}, \vec{v_5})$ **has switching cost 2.** In this case for $(\vec{v_1}, \vec{v_5})$, b is (s_2, s_4)-mobile and a is (s_4, s_5)-mobile, as indicated in row $\vec{v_5}$ (case 2) of Table 6.

Taking the reverse, $(\vec{v_5}, \vec{v_1})$ has switching cost 2 where a is (s_5, s_4)-mobile and b is (s_4, s_2)-mobile. Thus, with respect to $\vec{v_5}$, task s_2 is of type 2 with intermediate task s_4.

Similarly, Table 6 shows that $(\vec{v_5}, \vec{v_2})$ has switching cost 2 where b is (s_4, s_5)-mobile and a is (s_5, s_1)-mobile. Thus, with respect to $\vec{v_5}$, task s_1 is of type 2 with intermediate task s_5.

We observe that the combination of the previous two paragraphs violates Lemma 4.7. We apply Lemma 4.7 with parameters $(\vec{v}, s, t_1, t_2) = (\vec{v_5}, s_3, s_2, s_1)$. Then, from the previous two paragraphs, the intermediate task for task s_2 is task s_4 and the intermediate task for task s_1 is task s_5, so i_1 and i_2 are tasks s_4 and s_5 respectively. Then s, t_1, t_2, i_1, and i_2 are all distinct, which violates Lemma 4.7. □

We are now ready to prove Lemma 4.6.

Lemma 4.9 (Restatement of Lemma 4.6)**.** *For any demand vector $\vec{v}$ with at least four non-zero entries, there is at most one task of type 1.*

Proof. Suppose by way of contradiction that there are two type 1 tasks t_1, t_1' with respect to $\vec{v}$. By Lemma 4.8, there must be at least one type 2 task with respect to $\vec{v}$. Let t_2 be a type 2 task with respect to $\vec{v}$, choosing an empty task if possible. Let i be the intermediate task for t_2 with respect to $\vec{v}$. Either $i \neq t_1$ or $i \neq t'_1$. Without loss of generality, suppose $i \neq t_1$. Let s_1 be a non-empty task with $s_1 \neq i, t, t_2$.

For the construction, we require an additional non-empty task s_2 with $s_2 \neq s_1, i, t, t_2$. However, such a task might not exist since we are only assuming that $\vec{v}$ has at least four non-zero entries. We note that we cannot assume that $\vec{v}$ has at least five non-zero entries because Theorem 4.1 only assumes that the total number of agents is at least four. Thus, the existence of s_2 is a technicality that is only important when the number of agents is exactly four. We will first assume that such a task s_2 exists and later we will show that s_2 must indeed exist. Table 7 depicts the proof.

	s_1	i	t_1	t_2	s_2
$\vec{v}$	a	b			c
$\vec{v_1}$		a		b	c
$\vec{v_2}$		b	a		c
$\vec{v_3}$		c	a	b	
$\vec{v_4}$	a	b, not c			

Table 7: Demand vectors and the corresponding assignment of agents.

Let $\vec{v_1}$ be such that $(\vec{v}, \vec{v_1})$ are (s_1, t_2)-adjacent. Since t_2 is of type 2 with intermediate task i with respect to $\vec{v}$, $(\vec{v}, \vec{v_1})$ has switching cost 2 and intermediate task i. Let a be the (s, i)-mobile agent and let b be the (i, t_2)-mobile agent, as shown in Table 7.

Let $\vec{v_2}$ be such that $(\vec{v}, \vec{v_2})$ are (s_1, t_1)-adjacent. Since t_1 is of type 1 with respect to $\vec{v}$, $(\vec{v}, \vec{v_2})$ has switching cost 1. Thus, agents b and c are not mobile for $(\vec{v}, \vec{v_3})$ as shown in Table 7. If a is the mobile agent for $(\vec{v}, \vec{v_2})$ then a is assigned to task t_1 for $\vec{v_2}$ and otherwise a is assigned to task s. In either case, the assignment of both a and b differs between $\vec{v_2}$ and $\vec{v_1}$. Since $(\vec{v_1}, \vec{v_2})$ are adjacent, they cannot differ on the assignment of any other agents besides a and b. Thus, $(\vec{v_1}, \vec{v_2})$ has b as its (t_2, i)-mobile agent and a as its (i, t_1) mobile agent, as shown in Table 7. Thus, with respect to $\vec{v_1}$, task t_1 is of type 2 with intermediate task i and intermediate agent a. We have also shown that the mobile agent for $(\vec{v}, \vec{v_2})$ is a.

Let $\vec{v_3}$ be such that $(\vec{v_1}, \vec{v_3})$ are (s_2, t_1)-adjacent. Since t_1 is of type 2 with intermediate task i and intermediate agent a, $(\vec{v_1}, \vec{v_3})$ has switching cost 2 and a is the (i, t_1)-mobile agent. Let c be the (s_2, i)-mobile agent for $(\vec{v_1}, \vec{v_3})$. Then, $\vec{v_3}$ is as is in Table 7.

Let $\vec{v_4}$ be such that $(\vec{v}, \vec{v_4})$ are (s_2, t_1)-adjacent. Since t_1 is of type 1 with respect to $\vec{v}$, $(\vec{v}, \vec{v_4})$ has switching cost 1. Thus, neither a nor b are mobile agents for $(\vec{v}, \vec{v_4})$, as shown in Table 7. If c is the mobile agent for $(\vec{v}, \vec{v_4})$ then $\vec{v_4}$ assigns c to task t_1 and otherwise $\vec{v_4}$ assigns c to s_2. In either case, $\vec{v_4}$ does not assign c to task i, as shown in Table 7. We note that $(\vec{v_3}, \vec{v_4})$ are (t_2, s_1)-adjacent, however according to Table 7, $\vec{v_3}$ and $\vec{v_4}$ disagree on the assignment of a, b, and c, a contradiction.

In the above construction, we assumed the existence of task s_2. It remains to show that there indeed exists a non-empty task s_2 with $s_2 \neq s, i, t, t_2$. As mentioned previously, the existence of s_2 is a technicality that is only important when the number of agents is exactly four. Thus, the remainder of the proof merely addresses a technicality. Table 8 depicts the remainder of the proof.

Suppose by way of contradiction that s_1, i, t, and t_2 are the only non-empty tasks in $\vec{v}$. Let s' be an empty task. The task t_2 was initially chosen to be an *empty* type 2 task if one exists. Since t_2 is non-empty, we know that s' is of type 1.

Let $\vec{v_1}$ be such that $(\vec{v}, \vec{v_1})$ are (s_1, t_2)-adjacent. Since t_2 is of type 2 with intermediate task i with respect to $\vec{v}$, $(\vec{v}, \vec{v_1})$ has switching cost 2 and intermediate task i. For $(\vec{v}, \vec{v_1})$, let a be the (s_1, i)-mobile agent and let b be the (i, t_2)-mobile agent, as shown in Table 8.

Let $\vec{v_2}$ be such that $(\vec{v}, \vec{v_2})$ are (t, s')-adjacent. Since s' is of type 1 with respect to $\vec{v}$, $(\vec{v}, \vec{v_2})$ has switching cost 1. Let c be the mobile agent for $(\vec{v}, \vec{v_2})$ as shown in Table 8.

	s_1	i	t_1	t_2	s'
$\vec{v}$	a	b	c		
$\vec{v_1}$		a	c	b	
$\vec{v_2}$	a	b			c
$\vec{v_3}$		b	c		a
$\vec{v_4}$				b	a, not c

Table 8: Demand vectors and the corresponding assignment of agents.

Let $\vec{v_3}$ be such that $(\vec{v}, \vec{v_3})$ are (s_1, s')-adjacent. Since s' is of type 1 with respect to $\vec{v}$, $(\vec{v}, \vec{v_3})$ has switching cost 1. Thus, agents b and c are not mobile for $(\vec{v}, \vec{v_3})$ as shown in Table 8. If a is the mobile agent for $(\vec{v}, \vec{v_3})$ then a is assigned to task s' for $\vec{v_3}$ and otherwise a is assigned to task s. In either case, the assignment of both a and b differs between $\vec{v_3}$ and $\vec{v_1}$. Since $(\vec{v_1}, \vec{v_3})$ are adjacent, they cannot differ on the assignment of any other agents besides a and b. Thus, $(\vec{v_1}, \vec{v_3})$ has b as its (t_2, i)-mobile agent and a as its (i, s') mobile agent, as shown in Table 8. Thus, with respect to $\vec{v_1}$, task s' is of type 2 with intermediate task i and intermediate agent a. We have also shown that the mobile agent for $(\vec{v}, \vec{v_3})$ is a.

Let $\vec{v_4}$ be such that $(\vec{v_1}, \vec{v_4})$ are (t_1, s')-adjacent. Since task s' is of type 2 with intermediate task i and intermediate agent a with respect to $\vec{v_1}$, $(\vec{v_1}, \vec{v_4})$ have switching cost 2 and intermediate task i and (i, s')-mobile agent a. Regardless of the (t_1, i)-mobile agent for $(\vec{v_1}, \vec{v_4})$, agent c is not assigned to task s' for $\vec{v_4}$, as shown in Table 8.

We note that $(\vec{v_2}, \vec{v_4})$ are (s_1, t_2)-adjacent, however according to Table 8 they disagree on the assignment of a, b, and c, a contradiction. □

We are now ready to prove Lemma 4.3.

Lemma 4.10 (restatement of Lemma 4.3). *Let $\vec{v}$ be a demand vector with at least four non-zero entries. Then there exists a task t such that t is of type 2 with respect to $\vec{v}$ and $\vec{v}$ has at least four non-empty tasks distinct from t. Let i be the intermediate task of t with respect to $\vec{v}$. Let $\vec{v'}$ be such that $(\vec{v}, \vec{v'})$ are (i, t)-adjacent. Then, t is a type 2 task with intermediate task i with respect to $\vec{v'}$.*

Proof. First we will show that there exists a task t such that t is of type 2 with respect to $\vec{v}$ and $\vec{v}$ has at least four non-empty tasks distinct from t. By Lemma 4.8, there exists a task of type 2 with respect to $\vec{v}$. Suppose for contradiction that every such task t' is such that $\vec{v}$ has less than four non-empty tasks distinct from t'. Then, every empty task is of type 1, because if there were an empty task t'' of type 2 then $\vec{v}$ would have at least four non-empty tasks distinct from t'' since $\vec{v}$ has at least four non-zero entries. By Lemma 4.5, task i is of type 1 with respect to $\vec{v}$. Since i is an intermediate task with respect to $\vec{v}$, i is non-empty. Then, by Lemma 4.6 i is the only type 1 task with respect to $\vec{v}$. Thus, we have shown that every empty task is of type 1 and there are no empty type 1 tasks, so *every* task is non-empty with respect to $\vec{v}$. Then since there are at least 5 tasks total, there are at least four non-empty tasks distinct from t', a contradiction.

Now, we will show that t is a type 2 task with intermediate task i, with respect to $\vec{v'}$. Since $\vec{v}$ has at least four non-empty tasks excluding task t, $\vec{v'}$ has at least four non-zero entries. Thus, we can apply Lemma 4.9 to $\vec{v'}$.

We claim that task i is of type 1 with respect to $\vec{v'}$. The proof of this claim is simple and is depicted in Table 9.

By Lemma 4.4, $(\vec{v}, \vec{v'})$ have switching cost 1. Let b be the mobile agent for $(\vec{v}, \vec{v'})$, as shown in Table 9. Let s be a task with $s \neq i, t$. Let $\vec{v_1}$ be such that $(\vec{v}, \vec{v_1})$ are (s, t)-adjacent. Since task t is of type 2 with respect to $\vec{v}$, $(\vec{v}, \vec{v_1})$ has switching cost 2 and intermediate task i. Let a be the (s, i)-mobile agent, as shown in Table 9.

	s	i	t
$\vec{v}$	a	b	
$\vec{v'}$	a		b
$\vec{v_1}$		a	b

Table 9: Demand vectors and the corresponding assignment of agents.

If the (i,t)-mobile agent for $(\vec{v}, \vec{v_1})$ is some agent $c \neq b$, then $\vec{v'}$ and $\vec{v_1}$ disagree on the position of a, b, and c, which is impossible since $(\vec{v'}, \vec{v_1})$ are adjacent. Thus, b is the (i,t)-mobile agent for $(\vec{v}, \vec{v_1})$ as shown in Table 9. From Table 9, it is clear that $(\vec{v'}, \vec{v_1})$ are (s,i)-adjacent and have switching cost 1, and $(\vec{v'}, \vec{v})$ are (t,i)-adjacent and have switching cost 1. Thus, i is of type 1 with respect to $\vec{v'}$.

By Lemma 4.9, task i is the only task of type 1 with respect to $\vec{v'}$, so task t is of type 2. It remains to show that task t has intermediate task i with respect to $\vec{v'}$. If task t has a different intermediate task i' with respect to $\vec{v'}$, then by Lemma 4.5, task i' is of type 1 with respect to $\vec{v'}$, but we already know that task i is the only task of type 1 with respect to $\vec{v'}$. Thus, task t has intermediate task i with respect to $\vec{v'}$. □

5 Acknowledgments

We would like to thank Yufei Zhao for a discussion.

References

[1] Samuel N Beshers and Jennifer H Fewell. Models of division of labor in social insects. *Annual review of entomology*, 46(1):413–440, 2001.

[2] Eduardo Castello, Tomoyuki Yamamoto, Yutaka Nakamura, and Hiroshi Ishiguro. Task allocation for a robotic swarm based on an adaptive response threshold model. In *2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)*, pages 259–266. IEEE, 2013.

[3] Jianing Chen. *Cooperation in Swarms of Robots without Communication*. PhD thesis, University of Sheffield, 2015.

[4] David Conlon, Jacob Fox, and Benny Sudakov. Hypergraph ramsey numbers. *Journal of the American Mathematical Society*, 23(1):247–266, 2010.

[5] Alejandro Cornejo, Anna Dornhaus, Nancy Lynch, and Radhika Nagpal. Task allocation in ant colonies. In *International Symposium on Distributed Computing*, pages 46–60. Springer, 2014.

[6] Anna Dornhaus, Nancy Lynch, Frederik Mallmann-Trenn, Dominik Pajak, and Tsvetomira Radeva. Self-stabilizing task allocation in spite of noise. *arXiv preprint arXiv:1805.03691*, 2018.

[7] Ana Duarte, Ido Pen, Laurent Keller, and Franz J Weissing. Evolution of self-organized division of labor in a response threshold model. *Behavioral ecology and sociobiology*, 66(6):947–957, 2012.

[8] Paul Erdős and Richard Rado. Combinatorial theorems on classifications of subsets of a given set. *Proceedings of the London mathematical Society*, 3(1):417–439, 1952.

[9] Chryssis Georgiou and Alexander A Shvartsman. Cooperative task-oriented computing: Algorithms and complexity. *Synthesis Lectures on Distributed Computing Theory*, 2(2):1–167, 2011.

[10] Serge Kernbach, Dagmar Häbe, Olga Kernbach, Ronald Thenius, Gerald Radspieler, Toshifumi Kimura, and Thomas Schmickl. Adaptive collective decision-making in limited robot swarms without communication. *The International Journal of Robotics Research*, 32(1):35–55, 2013.

[11] Min-Hyuk Kim, Hyeoncheol Baik, and Seokcheon Lee. Response threshold model based uav search planning and task allocation. *Journal of Intelligent & Robotic Systems*, 75(3-4):625–640, 2014.

[12] Michael JB Krieger, Jean-Bernard Billeter, and Laurent Keller. Ant-like task allocation and recruitment in cooperative robots. *Nature*, 406(6799):992, 2000.

[13] Kristina Lerman, Chris Jones, Aram Galstyan, and Maja J Matarić. Analysis of dynamic task allocation in multi-robot systems. *The International Journal of Robotics Research*, 25(3):225–241, 2006.

[14] Kathryn Sarah Macarthur, Ruben Stranders, Sarvapali D Ramchurn, and Nicholas R Jennings. A distributed anytime algorithm for dynamic task allocation in multi-agent systems. In *AAAI*, pages 701–706, 2011.

[15] James McLurkin and Daniel Yamins. Dynamic task assignment in robot swarms. In *Robotics: Science and Systems*, volume 8. Citeseer, 2005.

[16] James Dwight McLurkin. *Stupid robot tricks: A behavior-based distributed algorithm library for programming swarms of robots*. PhD thesis, Massachusetts Institute of Technology, 2004.

[17] George F Oster and Edward O Wilson. *Caste and ecology in the social insects*. Princeton University Press, 1979.

[18] Jacques Penders, Lyuba Alboul, Ulf Witkowski, Amir Naghsh, Joan Saez-Pons, Stefan Herbrechtsmeier, and Mohamed El-Habbal. A robot swarm assisting a human fire-fighter. *Advanced Robotics*, 25(1-2):93–117, 2011.

[19] Tsvetomira Radeva, Anna Dornhaus, Nancy Lynch, Radhika Nagpal, and Hsin-Hao Su. Costs of task allocation with local feedback: Effects of colony size and extra workers in social insects and other multi-agent systems. *PLoS computational biology*, 13(12):e1005904, 2017.

[20] Gene E Robinson. Regulation of division of labor in insect societies. *Annual review of entomology*, 37(1):637–665, 1992.

[21] Erol Şahin. Swarm robotics: From sources of inspiration to domains of application. In *International workshop on swarm robotics*, pages 10–20. Springer, 2004.

[22] Hsin-Hao Su, Lili Su, Anna Dornhaus, and Nancy Lynch. Ant-inspired dynamic task allocation via gossiping. In *International Symposium on Stabilization, Safety, and Security of Distributed Systems*, pages 157–171. Springer, 2017.

[23] Zijian Wang and Mac Schwager. Multi-robot manipulation with no communication using only local measurements. In *CDC*, pages 380–385, 2015.

[24] Yongming Yang, Changjiu Zhou, and Yantao Tian. Swarm robots task allocation based on response threshold model. In *2009 4th International Conference on Autonomous Robots and Agents*, pages 171–176. IEEE, 2009.

[25] Emaad Mohamed H Zahugi, Mohamed M Shanta, and TV Prasad. Oil spill cleaning up using swarm of robots. In *Advances in Computing and Information Technology*, pages 215–224. Springer, 2013.