View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Dagstuhl Research Online Publication Server

Lower Bounds for Dynamic Distributed Task Allocation

Hsin-Hao Su
Boston College, MA, USA

Nicole Wein
Massachusetts Institute of Technology, Cambridge, MA, USA

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 [20], 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.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases distributed task allocation, combinatorics, lower bounds, multi-agent systems, low-distortion embedding, dynamic algorithms, biological distributed algorithms

Digital Object Identifier 10.4230/LIPIcs.ICALP.2020.99

Category Track A: Algorithms, Complexity and Games

Funding *Nicole Wein*: supported by an NSF Graduate Fellowship and NSF Grant CCF-1514339.

Acknowledgements We would like to thank Yufei Zhao for a discussion.

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 [7] 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 [4, 20, 5, 17].

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

© Hsin-Hao Su and Nicole Wein;
licensed under Creative Commons License CC-BY

experimental work on heuristics for dynamic task allocation (see e.g. [10, 19, 13, 14, 11, 12]). 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 [15]. 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, 18]).

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 [20] present and analyze gossip-based algorithms for dynamic task allocation in ant colonies. Radeva, Dornhaus, Lynch, Nagpal, and Su [17] 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 [20].

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

1.1.3.1 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 [6]).

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, 9, 22]. Our setting also relates more generally to swarm robotics:

1.1.3.2 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. [21, 3, 8, 16]). 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 [16, 23], agricultural foraging, mining, drone package delivery, and environmental monitoring [19].

1.2 Past Work

Our problem was previously studied only by Su, Su, Dornhaus, and Lynch [20], who presented two upper bounds and a lower bound.

The first upper bound is a very simple set of functions $f_1^{n,k}, \dots, f_n^{n,k}$ 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 $f_1^{n,k}, \dots, f_n^{n,k}$ 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 $\sum_{i=0}^{j-1} d_i < \mathrm{ID}(a)$ and $\sum_{i=0}^{j} d_i \geq \mathrm{ID}(a)$. 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 $f_1^{n,k}, \dots, f_n^{n,k}$ with maximum switching cost 1 for $n \geq 2$ and $k \geq 3$. Suppose for contradiction that there exists a set of functions $f_1^{n,k}, \dots, f_n^{n,k}$ 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 $f_1^{n,k}, \ldots, f_n^{n,k}$ 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 4.** *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.*

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

The value of k for Theorem 5 is an extremely large constant derived from hypergraph Ramsey numbers. Specifically, there exists a constant c so that Theorem 5 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 4 for $n = 3$, $k = 5$, and more importantly, to prove Theorem 5. We begin by supposing for contradiction that there exists a set of functions $f_1^{n,k}, \ldots, f_n^{n,k}$ with switching cost 2 and 3, respectively, and then reason about the structure of these functions. The main challenge in proving Theorem 5 as compared to Theorem 4 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 4 for $n = 3$ and $k = 5$ serves as a simple illustrative example of our proof framework, while the proof of Theorem 5 is more involved. In particular, in step 1 of the proof of Theorem 5, 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 4. 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 6. 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 4 and 5 in the language of the problem restatement.

▶ **Theorem 7** (Restatement of Theorem 4). *Let $n \geq 3$ and $k \geq 5$. Every function $\pi_{n,k}$ has maximum distortion at least 3.*

▶ **Theorem 8** (Restatement of Theorem 5). *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 7 for $n = 3$, $k = 5$, and Theorem 8. 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 7, 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 8, 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 7 for $n = 3$, $k = 5$

In this section, we prove Theorem 7 for $n = 3$, $k = 5$, which serves as a simple illustrative example of our proof framework from Section 3.1.

▶ **Theorem 9** (Special case of Theorem 7). *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 10** (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 11.** *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 11 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 11. 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 9. 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 12.** *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 11 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 11, the pair $\{x, y\}$ freezes either x or y. Without loss of generality, say $\{x, y\}$ freezes x. By Lemma 12, 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. ◀

We defer the proof of Theorem 8, which is the remainder of Section 3, to the full version.

4 The remaining parameter regime

▶ **Theorem 13** (restatement of Theorem 4). *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 13 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 7 but we are considering multisets, in step 1 of the proof framework Lemma 11 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. (However we do use this proof framework to prove Theorem 8.)

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 13, we need to show that Theorem 9 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 14.** *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 $f_1^{n,k}, \ldots, f_n^{n,k}$. ◀

▶ **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 13 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 $f_1^{n,k}, \dots, f_n^{n,k}$ 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.

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.

We defer the proof of Theorem 13, which is the remainder of Section 4 to the full version.

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 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.

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

6 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.

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

8 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.

9 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.

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

11 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.

12 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.

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

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

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

16 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.

17 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.

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

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

20 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.

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

22 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.

23 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.