Anytime Heuristic for Weighted Matching Through Altruism-Inspired Behavior

Panayiotis Danassis, **Aris Filos-Ratsikas** and **Boi Faltings**
Artificial Intelligence Laboratory, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
{panayiotis.danassis, aris.filosratsikas, boi.faltings}@epfl.ch

Abstract

We present a novel anytime heuristic (ALMA), inspired by the human principle of altruism, for solving the assignment problem. ALMA is decentralized, completely uncoupled, and requires no communication between the participants. We prove an upper bound on the convergence speed that is polynomial in the desired number of resources and competing agents per resource; crucially, in the realistic case where the aforementioned quantities are bounded independently of the total number of agents/resources, the convergence time remains *constant* as the total problem size increases.

We have evaluated ALMA under three test cases: (i) an anti-coordination scenario where agents with similar preferences compete over the same set of actions, (ii) a resource allocation scenario in an urban environment, under a constant-time constraint, and finally, (iii) an on-line matching scenario using *real* passenger-taxi data. In all of the cases, ALMA was able to reach high social welfare, while being orders of magnitude faster than the centralized, optimal algorithm. The latter allows our algorithm to scale to realistic scenarios with hundreds of thousands of agents, e.g., vehicle coordination in urban environments.

1 Introduction

One of the most relevant problems in multi-agent systems (MAS) is finding an optimal allocation between agents. This pertains to role allocation (e.g., team formation for autonomous robots [Gunn and Anderson, 2013]), task assignment (e.g., employees of a factory, taxi-passenger matching [Varakantham *et al.*, 2012]), resource allocation (e.g., parking spaces / charging stations for autonomous vehicles [Geng and Cassandras, 2013]), etc. What follows is *applicable to any such scenario*, but for concreteness we will refer to the allocation of a set of resources to a set of agents, a setting known as the *assignment problem*, one of the most fundamental combinatorial optimization problems [Munkres, 1957].

When designing algorithms for assignment problems, a significant challenge emerges from the nature of real-world applications, which is often distributed and information-restrictive. Even in decentralized algorithms, the number of communication rounds required grows with the size of the problem. However, in practice the real-time constraints impose a limit on the number of rounds, and thus on the size of the problem that can be solved within them. Moreover, sharing plans and preferences creates high overhead, and there is often a lack of responsiveness and/or communication between the participants [Stone *et al.*, 2010]. Achieving fast convergence and high efficiency in such information-restrictive settings is extremely challenging. Yet, humans are able to routinely and robustly coordinate in similar everyday scenarios, often with no explicit communication. One driving factor that facilitates human cooperation is behavioral conventions [Lewis, 2008]. Inspired by human behavior, the proposed heuristic is modeled on an *altruistic* convention. This results to fast convergence to highly efficient allocations, without any communication between the agents.

A distinctive characteristic of ALMA is that agents make decisions locally, based on (i) the contest for resources that *they* are interested in, (ii) the agents that are interested in the *same* resources. If each agent is interested in only a *subset* of the total resources, ALMA converges in time polynomial in the maximum size of the subsets; not the total number of resources. In particular, if the size of each subset is a constant function of the total number of resources, then the convergence time is *constant*, in the sense that it does not grow with the problem size. The same is not true for other algorithms (e.g., the optimal centralized solution) which require time polynomial in the *total* number of agents/resources, even if the aforementioned condition holds. The condition holds by default in many real-world applications; agents have only local knowledge of the world, there is typically a cost associated with acquiring a resource, or agents are simply only interested in resources in their vicinity (e.g., urban environments). This is important, as the proposed approach avoids having to artificially split the problem in subproblems (e.g, by placing bounds or spatial constraints) and solve those separately, in order to make it tractable. Instead, ALMA utilizes a natural domain characteristic, not an artificial optimization technique (i.e., artificial bounds). Coupled to the convergence time, the decentralized nature of ALMA makes it applicable to large-scale, real-world applications (e.g., IoT devices, intelligent infrastructure, autonomous vehicles, etc.).

1.1 Our Results

Our main contributions in this paper are:

(1) We introduce a novel, anytime **AL**truistic **MA**tching heuristic (**ALMA**) for solving the assignment problem. ALMA is decentralized, completely uncoupled (agents are only aware of own history of action/reward pairs [Talebi, 2013]), and requires no communication between the agents.

(2) We prove that if we bound the maximum number of resources an agent is interested in, and the maximum number of agents competing for a resource, the expected number of steps for any agent to converge is independent of the total problem size. Thus, we do not require to artificially split the problem, or similar techniques, to render it manageable.

(3) We provide a thorough empirical evaluation of ALMA on both synthetic and real data. In particular, we have evaluated ALMA under three test cases: (i) an anti-coordination scenario where agents with similar preferences compete over the same set of actions, (ii) a resource allocation scenario in an urban environment, under a constant-time constraint, and finally, (iii) an on-line matching scenario using *real* passenger-taxi data. In all of the cases, ALMA achieves high social welfare (total satisfaction of the agents) as compared to the optimal solution, as well as various other algorithms.

Full Version

Please see [Danassis *et al.*, 2019] for the full version of the paper, which includes the converge proof of Theorem 2.1, and the omitted details from Section 3, including, but not limited to, the omitted graphs and tables.

1.2 Related Work

The assignment problem consists of finding a maximum weight matching in a weighted bipartite graph and it is one of the best-studied combinatorial optimization problems in the literature. The first polynomial time algorithm (with respect to the total number of nodes, edges) was introduced by Jacobi in the 19th century [Borchardt and Jacobi, 1865] [Ollivier, 2009], and was succeeded by many classical algorithms [Munkres, 1957] [Edmonds and Karp, 1972] [Bertsekas, 1979] with the *Hungarian* algorithm of Kuhn 1955 being the most prominent one (see [Su, 2015] for an overview). The problem can also be solved via linear programming [Dantzig, 1990], as its LP formulation relaxation admits integral optimal solutions [Papadimitriou and Steiglitz, 1982]. In Section 3.3, we will apply ALMA on a non-bipartite setting, which corresponds to the more general *maximum weight matching* problem on general graphs. To compute the optimal in this case, we will use the *blossom algorithm* of [Edmonds, 1965] (see [Lovász and Plummer, 2009]).

In reality, a centralized coordinator is not always available, and if so, it has to know the utilities of all the participants, which is often not feasible. In the literature of the assignment problem, there also exist several decentralized algorithms (e.g., [Giordani *et al.*, 2010] [Ismail and Sun, 2017] [Zavlanos *et al.*, 2008] [Bürger *et al.*, 2012] which are the decentralized versions of the aforementioned well-known centralized algorithms – see also [Kuhn *et al.*, 2016] [Elkin, 2004] for general results in distributed approximability under only local information/computation). However, these algorithms require polynomial computational time and polynomial number of messages (such as cost matrices [Ismail and Sun, 2017], pricing information [Zavlanos *et al.*, 2008], or a basis of the LP [Bürger *et al.*, 2012], etc.). Yet, agent interactions often repeat no more than a few hundreds of times. To the best of our knowledge, a decentralized algorithm that requires no message exchange (i.e., no communication network) between the participants, and achieves high efficiency, like ALMA does, has not appeared in the literature before. Let us stress the importance of such a heuristic: as autonomous agents proliferate, and their number and diversity continue to rise, differences between the agents in terms of origin, communication protocols, or the existence of sub-optimal, legacy agents will bring forth the need to collaborate without any form of explicit communication [Stone *et al.*, 2010]. Finally, communication between participants creates high overhead as well.

ALMA is inspired by the allocation algorithm of [Cigler and Faltings, 2013] (adapted in [Danassis and Faltings, 2019] to solve resource allocation problems under rationality constraints). Using such a simple learning rule which only requires environmental feedback, allows our approach to scale to hundreds of thousands of agents. Moreover, it does not require global knowledge of utilities; only local knowledge of personal utilities (in fact, we require knowledge of pairwise differences which are far easier to estimate).

2 Altruistic Matching Heuristic

In this section, we define ALMA and prove its convergence properties. We start by defining the assignment problem.

The assignment problem consists of finding a maximum weight matching in a weighted bipartite graph, $\mathcal{G} = \{\mathcal{N} \cup \mathcal{R}, \mathcal{E}\}$. In the studied scenario, $\mathcal{N} = \{1, \dots, N\}$ agents compete to acquire $\mathcal{R} = \{1, \dots, R\}$ resources. We assume that each agent n is interested in a subset of the total resources, i.e., $\mathcal{R}^n \subset \mathcal{R}$. The weight of an edge $(n, r) \in \mathcal{E}$ represents the utility ($u_n(r) \in [0, 1]$) agent n receives by acquiring resource r. Each agent can acquire at most one resource, and each resource can be assigned to at most one agent. The goal is to maximize the social welfare (sum of utilities), i.e., $\max_{\mathbf{x} \geq 0} \sum_{(n,r) \in \mathcal{E}} u_n(r) x_{n,r}$, subject to $\sum_{r|(n,r) \in \mathcal{E}} x_{n,r} = 1, \forall n \in \mathcal{N}$, and $\sum_{n|(n,r) \in \mathcal{E}} x_{n,r} = 1, \forall r \in \mathcal{R}$.

2.1 Learning Rule

This section describes the proposed heuristic (**ALMA**: **AL**truistic **MA**tching heuristic) for weighted matching. We make the following two assumptions: First, we assume (possibly noisy) knowledge of personal preferences by each agent. Second, we assume that agents can observe feedback from their environment. This is used to inform collisions and detect free resources. It could be achieved by the use of visual, auditory, olfactory sensors etc., or by any other means of feedback from the resource (e.g., by sending an occupancy message). Note here that these messages would be between the requesting agent and the resource, not between the participating agents themselves, and that it suffices to send only 1 bit of information (e.g., 0, 1 for occupied / free respectively).

ALMA learns the right action through repeated trials as follows. Each agent sorts his available resources (possibly $\mathcal{R}^n \subseteq \mathcal{R}$) in decreasing order of utility ($r_1, r_2, \ldots, r_i, r_{i+1}, \ldots, r_{R^n}$). The set of available actions is denoted as $\mathcal{A} = \{Y, A_{r_1}, \ldots, A_{r_{R^n}}\}$, where Y refers to yielding, and A_r refers to accessing resource r. Each agent has a strategy (g_n) that points to a resource and it is initialized to the most preferred one. As long as an agent has not acquired a resource yet, at every time-step, there are two possible scenarios. If $g_n = A_r$ (strategy points to resource r), then agent n attempts to acquire that resource. If there is a collision, the colliding parties back-off with some probability. Otherwise, if $g_n = Y$, the agent choses a resource r for monitoring. If the resource is free, he sets $g_n \leftarrow A_r$. Alg. 1 presents the pseudo-code of ALMA, which is followed by every agent individually. The back-off probability ($P_n(\cdot)$) and the next resource to monitor ($S_n(\cdot)$) are computed individually and locally based on the current resource and each agent's utilities, as will be explained in the following section. Finally, note that if the available resources change over time, the agents simply need to sort again the currently available ones.

2.2 Back-off Probability & Resource Selection

Let $\mathcal{R}$ be totally ordered in decreasing utility under $\prec_n, \forall n \in \mathcal{N}$. If more than one agent compete for resource r_i (step 4 of Alg. 1), each of them will back-off with probability that depends on their utility loss of switching to their respective remaining resources. The loss is given by Eq. 1.

$$loss_n^i = \frac{\sum_{j=i+1}^{k} (u_n(r_i) - u_n(r_j))}{k - i} \tag{1}$$

where $k \in \{i+1, \ldots, R^n\}$ denotes the number of remaining resources to be considered. For $k = i + 1$, the formula only takes into account the utility loss of switching to the immediate next best resource, while for $k = R^n$ it takes into account the average utility loss of switching to all of the remaining resources. In the remainder of the paper we assume $k = i + 1$, i.e., $loss_n^i = u_n(r_i) - u_n(r_{i+1})$. The actual back-off probability can be computed with any monotonically decreasing function f on $loss_n$, i.e., $P_n(r_i, \prec_n) = f_n(loss_n^i)$. In the evaluation section, we have used two such functions, a linear (Eq. 2), and the logistic function (Eq. 3). The parameter ϵ places a threshold on the minimum / maximum back-off probability for the linear curve, while γ determines the steepness of the logistic curve.

$$f(loss) = \begin{cases} 1 - \epsilon, & \text{if } loss \leq \epsilon \\ \epsilon, & \text{if } 1 - loss \leq \epsilon \\ 1 - loss, & \text{otherwise} \end{cases} \tag{2}$$

$$f(loss) = \frac{1}{1 + e^{-\gamma(0.5 - loss)}} \tag{3}$$

Using the aforedescribed rule, agents that do not have good alternatives will be less likely to back-off and vice versa. The ones that do back-off select an alternative resource and examine its availability. The resource selection is performed in sequential order, i.e., $S_n(r_{\text{prev}}, \prec_n) = r_{\text{prev}+1}$, where r_{prev} denotes the resource selected by that agent in the previous round. We also examined using a weighted or uniformly at random selection, but achieved inferior results.

Algorithm 1 ALMA: Altruistic Matching Heuristic.

```
Require: Sort resources ($\mathcal{R}^n \subseteq \mathcal{R}$) in decreasing order of
    utility $r_1, r_2, \ldots, r_i, r_{i+1}, \ldots, r_{R^n}$.
Require: Initialize $g_n \leftarrow A_{r_1}$, and $r_{\text{prev}} \leftarrow r_1$.
1: procedure ALMA
2:     if $g_n = A_r$ then
3:         Agent $n$ attempts to acquire $r$. Set $r_{\text{prev}} \leftarrow r$.
4:         if Collision($r$) then
5:             back-off (set $g_n \leftarrow Y$) with prob. $P_n(r, \prec_n)$.
6:     else ($g_n = Y$)
7:         $n$ monitors $r \leftarrow S_n(r_{\text{prev}}, \prec_n)$. Set $r_{\text{prev}} \leftarrow r$.
8:         if Free($r$) then set $g_n \leftarrow A_r$.
```

2.3 Altruism-Inspired Behavior

Human cooperation is unique in the sense that humans cooperate with strangers, even if there are no prospects of future interactions or reputation gains [Fehr and Rockenbach, 2004]. ALMA is inspired by the human principle of altruism [Nowak and Sigmund, 2005] [Gintis, 2000]. We would expect an altruistic person to give up a resource either to someone who values it more, if that resulted in an improvement of the well-being of society – which does not imply knowledge of the preferences of others, rather than a general expectation – [Charness and Rabin, 2002], or simply to be nice to others [Simon, 2016]. Such behavior is especially common in situations where the backing-off subject has equally good alternatives. E.g., in human pick-up teams, each player typically attempts to fill his most preferred position. If there is a collision, a colliding player might back-off because his teammate is more competent in that role, or because he has an equally good alternative, or simply to be polite; the player backs-off now and assumes that role at some future game. From an alternative viewpoint, following such an altruistic convention leads to a faster convergence which outweighs the loss in utility. Such conventions allow humans to routinely and robustly coordinate in large scale and under dynamic and unpredictable demand. Behavioral conventions are a fundamental part of human societies [Lewis, 2008], yet they have not appeared meaningfully in empirical modeling of MAS. Inspired by human behavior, ALMA attempts to reproduce these simple rules in an artificial setting.

2.4 Convergence

Theorem 2.1. For N agents and R resources, the expected number of steps until the system of agents following Alg. 1 converges to a complete matching is bounded by (4), where $p^* = f(loss^*)$, and $loss^*$ is given by the equation below.

$$\mathcal{O}\left(R \frac{2 - p^*}{2(1 - p^*)} \left(\frac{1}{p^*} \log N + R\right)\right) \tag{4}$$

$$loss^* = \arg\min_{loss_n^r} \left(\min_{r \in \mathcal{R}, n \in \mathcal{N}} (loss_n^r), 1 - \max_{r \in \mathcal{R}, n \in \mathcal{N}} (loss_n^r) \right)$$

Proof. To improve readability, we will only provide a sketch of the proof. Please see [Danassis *et al.*, 2019] for the complete proof. The proof is based on [Danassis and Faltings, 2019] [Cigler and Faltings, 2013].

We first assume that every agent, on every collision, backs-off with the same constant probability p. We start with the case of having N agents competing for 1 resource and model our system as a discrete time Markov chain. Intuitively, this Markov chain describes the number of individuals in a decreasing population, but with two caveats: the goal (absorbing state) is to reach a point where only one individual remains, and if we reach zero, we restart. We prove that the expected number of steps until we reach a state where either 1 or 0 agents compete for that resource is $\mathcal{O}\left(\frac{1}{p}\log N\right)$. Moreover, we prove that with high probability, $\Omega\left(\frac{2(1-p)}{2-p}\right)$, only 1 agent will remain (contrary to reaching 0 and restarting the process of claiming the resource), no matter the initial number of agents. Having proven that, we move to the general case of N agents competing for R resources.

At any time, at most N agents can compete for each resource. We call this period a round. During a round, the number of agents competing for a specific resource monotonically decreases, since that resource is perceived as occupied by non-competing agents. Let the round end when either 1 or 0 agents compete for the resource. This will require $\mathcal{O}\left(\frac{1}{p}\log N\right)$ steps. If all agents backed-off, it will take on average R steps until at least one of them finds a free resource. We call this period a break. In the worst case, the system will oscillate between a round and a break. According to the above, one oscillation requires in expectation $\mathcal{O}\left(\frac{1}{p}\log N + R\right)$ steps. If $R = 1$, as mentioned in the previous paragraph, in expectation there will be $\frac{2-p}{2(1-p)}$ oscillations. For $R > 1$ the expected number of oscillations is bounded by $\mathcal{O}\left(R\frac{2-p}{2(1-p)}\right)$. Thus, we conclude that if all the agents back-off with the same constant probability p, the expected number of steps until the system converges to a complete matching is $\mathcal{O}\left(R\frac{2-p}{2(1-p)}\left(\frac{1}{p}\log N + R\right)\right)$.

Next, we drop the constant probability assumption. Intuitively, the worst case scenario corresponds to either all agents having a small back-off probability, thus they keep on competing for the same resource, or all of them having a high back-off probability, thus the process will keep on restarting. These two scenarios correspond to the inner ($\frac{1}{p}$) and outer ($\frac{2-p}{2(1-p)}$) probability terms of bound (4) respectively. Let $p^* = f(loss^*)$ be the worst between the smallest or highest back-off probability any agent $n \in \mathcal{N}$ can exhibit, i.e., having $loss^*$ given by Eq. 2.1. Using p^* instead of the constant p, we bound the expected convergence time according to bound (4). □

It is worth noting that the back-off probability p^* in bound (4) does not significantly affect the convergence time. For example, using Eq. 2 with a quite small $\epsilon = 0.01$, the resulting quantities would be at most $100R\log N$, and $50R^2$. Most importantly, though, this is a rather loose bound (e.g., agents would rarely back-off with probabilities as extreme as p^*).

Apart from the convergence of the whole system, we are interested in the expected number of steps any individual agent would require in order to acquire a resource. In real-world scenarios, there is typically a cost associated with acquiring a resource. E.g., a taxi driver would not be willing to drive to the other end of the city to pick up a low fare passenger. As a result, each agent is typically interested in a subset of the total resources, i.e., $\mathcal{R}^n \subset \mathcal{R}$, thus at each resource there is a bounded number of competing agents. Let R^n denote the maximum number of resources agent n is interested in, and N^r denote the maximum number of agents competing for resource r. By bounding these two quantities (i.e., we consider R^n and N^r to be constant functions of N, R), Corollary 2.1.1 proves that the expected number of steps any individual agent requires to converge [1] is independent of the total problem size (i.e., N, and R), or, in other words, that the convergence time is *constant* in these quantities.

Corollary 2.1.1. Let $R^n = |\mathcal{R}^n|$, such that $\forall r \in \mathcal{R}^n : u_n(r) > 0$, and $N^r = |\mathcal{N}^r|$, such that $\forall n \in \mathcal{N}^r : u_n(r) > 0$. The expected number of steps until an agent $n \in \mathcal{N}$ following Alg. 1 converges is bounded by (5), where $p_n^* = f(loss^\star)$ and $loss^\star$ is given by Eq. 6, independent of the total problem size N, R.

$$\mathcal{O}\left(\max_{n' \in \cup_{r\in\mathcal{R}^n}\mathcal{N}^r} R^{n'} \frac{2-p_n^*}{2(1-p_n^*)}\left(\frac{1}{p_n^*}\log(\max_{r\in\mathcal{R}^n} N^r) + \max_{n' \in \cup_{r\in\mathcal{R}^n}\mathcal{N}^r} R^{n'}\right)\right) \quad (5)$$

$$loss^\star = \arg\min_{loss_n^r}\left(\min_{r\in\mathcal{R}^n, n\in\mathcal{N}^r}(loss_n^r), 1 - \max_{r\in\mathcal{R}^n, n\in\mathcal{N}^r}(loss_n^r)\right) \quad (6)$$

Proof. The expected number of steps until an agent $n \in \mathcal{N}$ successfully acquires a resource is upper bounded by the total convergence time of the sub-system he belongs to, i.e., the sub-system consisting of the sets of $\mathcal{R}^n$ resources and $\cup_{r\in\mathcal{R}^n}\mathcal{N}^r$ agents. In such scenario, at most $\max_{r\in\mathcal{R}^n} N^r$ agents can compete for any resource. Using Theorem 2.1 for $\max_{r\in\mathcal{R}^n} N^r$ agents, $\max_{n'\in\cup_{r\in\mathcal{R}^n}\mathcal{N}^r} R^{n'}$ resources, and worst-case $loss^\star$ given by any agent in $\cup_{r\in\mathcal{R}^n}\mathcal{N}^r$ (i.e., Eq 6) results in the desired bound. Note that agents do not compete for already claimed resources (step 8 of Alg. 1), thus the convergence of an agent does not require the convergence of agents of overlapping sub-systems. □

3 Evaluation

In this section we evaluate ALMA under various test cases. For the first two, we focus on convergence time, and relative difference in social welfare (SW), i.e., $(achieved - optimal)/optimal$. In every metric, we report the average value out of 128 runs of the same problem instance. Error bars represent one standard deviation (SD) of uncertainty. The third test case is an on-line setting, thus we report the achieved SW (not the relative difference), and the empirical competitive ratio (average out of 128 runs, as before). In Section 3.1 we use the logistic function, while in 3.2 & 3.3 we use the linear function (Eq. 2) with $\epsilon = 0.1$.

It is important to stress that our goal is not to improve the convergence speed of a centralized, or decentralized algorithm. Rather, the computation time comparisons of Sections

[1]In scenarios where $R < N$, or in case of deadlocks, convergence implies that the state of the agent does not change.

3.1 & 3.2 are meant to ground the actual speed of ALMA, and argue in favor of its applicability on large-scale, real-world scenarios. Given the nature of the problem, we elected to use a specialized algorithm to compute the optimal solution, rather than a general LP-based technique (e.g., the Simplex method). Specifically, we opted to use the Hungarian algorithm [Kuhn, 1955] which, first, has proven polynomial worse case bound, and second, as our simulations will demonstrate, can handle sufficiently large problems.

3.1 Test Case #1: Noisy Common Preferences

Setting As a first test case, we cover the extreme scenarios. The first pertains to an anti-coordination scenario where agents with similar preferences compete over the same set of actions. We call this 'noisy common preferences' and model the utilities as: $\forall n, n' \in \mathcal{N}, |u_n(r) - u_{n'}(r)| \leq$ noise, where the noise is sampled from a zero-mean Gaussian distribution, i.e., noise $\sim \mathcal{N}(0, \sigma^2)$. In the second scenario the utilities are initialized uniformly at random for each agent and resource.

Results The convergence time for the system of agents is linear to the number of resources R. ALMA requires approximately 4 to 6 orders of magnitude less computation time, and scales more gracefully, than the centralized Hungarian algorithm. Note also that in real-world applications we would have to take into account communication time, communication reliability protocols, etc., which create additional overhead for the Hungarian or any other algorithm for the assignment problem. The relative difference in social welfare reaches asymptotically zero as R increases. For a small number of resources, ALMA achieves the worst social welfare, approximately 11% worse than the optimal. Intuitively this is because when we have a small number of choices, a single wrong matching can have a significant impact to the final social welfare, while as the number of resources grow, the impact of an erroneous matching is mitigated. For 16384 resources we lose less than 2.5% of the optimal.

3.2 Test Case #2: Allocation in a Cartesian Map

In this section we examine the applicability of ALMA in large-scale MAS. Specifically we are interested in resource allocation in urban environments (e.g., parking spots / charging stations for autonomous vehicles, taxi - passenger matchings, etc.). The aforementioned problems become ever more relevant due to rapid urbanization, and the natural lack of coordination in the usage of resources. The latter result in the degradation of response (e.g., waiting time) and quality metrics in large cities [Varakantham, 2016].

Setting Let us consider a Cartesian map representing a city on which are randomly distributed vehicles and charging stations. The utility received by a vehicle n for using a charging station r is proportional to the inverse of their distance, i.e., $u_n(r) = 1/d_{n,r}$. Since we are in an urban environment, let $d_{n,r}$ denote the Manhattan distance. Typically, there is a cost each agent is willing to pay to drive to a resource, thus there is a cut-off distance, upon which the utility of acquiring the resource is zero (or possibly negative). This is a typical scenario encountered in resource allocation in urban environments, where there are spatial constraints and local interactions. The way such problems are typically tackled, is

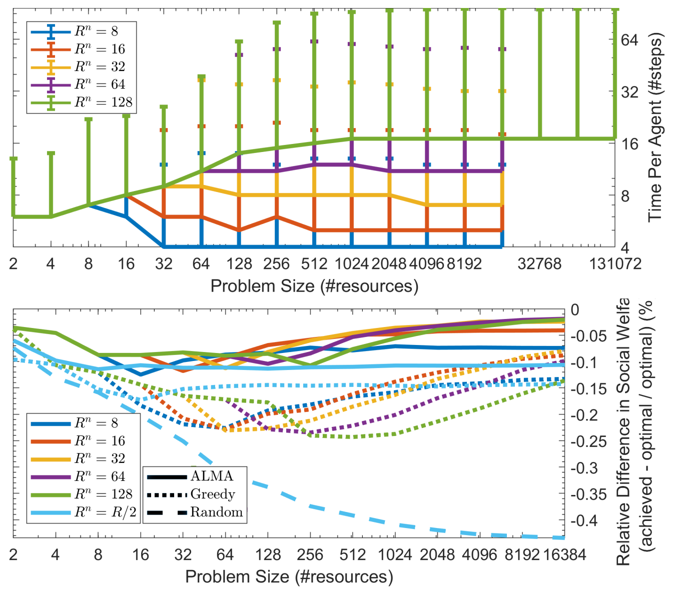

Figure 1: (top) Average time (#steps) for an agent to acquire a resource (double log), (bottom) Relative difference in SW (%) (single log), for increasing R, and $N = R$.

by dividing the map to sub-regions, and solving each individual sub-problem. For example, Singapore is divided into 83 zones based on postal codes [Cheng and Nguyen, 2011], and taxi drivers' policies are optimized according to those [Nguyen *et al.*, 2017] [Varakantham *et al.*, 2012]. On the other hand, not placing bounds means that the current solutions will not scale. To the best of our knowledge, we are the first to propose an anytime heuristic for resource allocation in urban environments that can scale in constant time without the need to artificially split the problem. Instead, ALMA exploits the two typical characteristics of an urban environment: the anonymity in interactions and homogeneity in supply and demand [Varakantham, 2016] (e.g., assigning any of two equidistant charging stations to a vehicle, would typically result to the same utility). This results in a simple learning rule which, as we will demonstrate in this section, can scale to hundreds of thousands of agents (we provide simulation results with up to 131072 agents and resources).

Convergence Time We begin by placing a bound on the maximum number of resources each agent is interested in, and on the maximum number of agents competing for a resource, specifically $R^n = N^r \in \{8, 16, 32, 64, 128\}$. According to Corollary 2.1.1, bounding these two quantities should result in convergence in constant time. The latter is corroborated by Fig. 1 (top), which shows that the average time-steps until an agent successfully claims a resource remains constant as we increase the total problem size (R, N). Compared to the Hungarian algorithm, ALMA requires approximately 7 orders of magnitude less computation time, and this number would grow boundlessly as we increase N, R.

Efficiency Along to the constant convergence time, Fig. 1 (bottom) demonstrates that ALMA is able to reach high quality matchings. As a reference, we have also included the centralized greedy and the random solutions. The former goes through the participating agents randomly, and assigns them their most preferred, unassigned resource. For $R > 2$ resources and different values of R^n, ALMA achieves between $1.9 - 12\%$ loss in SW, while the greedy algorithm achieves $8.0 - 24\%$ and the random algorithm $7.3 - 43.4\%$. While the greedy algorithm seems to perform relatively well in this test case, it does not take into account the utilities between agents. ALMA is of a greedy nature as well, albeit it utilizes a more intelligent backing-off scheme, thus there are scenarios where ALMA would significantly outperform the greedy (e.g., see Section 3.3). Finally, recall that ALMA operates in a significantly harder domain with no communication, limited feedback, and time constraints. In contrast, the greedy method requires either a central coordinator or message exchange (to communicate preferences and resolve collisions).

Anytime Property In the real world, agents are required to run in real time, which imposes time constraints. Restricting the system to only 32, 256, and 1024 time-steps, results in 1.25%, 0.12%, and 0.03% worse SW than the unrestricted version, respectively (we do not suggest that this is the case in any domain. For example, in the domain of Test Case #1, the quality of the achieved matching decreases boundlessly as we decrease the alloted time. Nevertheless, the aforedescribed domain is a realistic one, with a variety of real-world applications). Finally, the repeated nature of such problems suggests that even in the case of a deadlock, the agent which failed to acquire a resource, will do so in some subsequent round.

3.3 Test Case #3: On-line Taxi Request Match

In this section we present a motivating test case involving ride-sharing, via on-line taxi request matching, using *real* data of taxi rides in New York City. Ride-sharing, offers great potential in congestion relief and environmental sustainability. In the past few years, several commercially successful ride-sharing companies have surfaced (e.g., Uber, Lyft, etc.), giving rise to a new incarnation of ride-sharing: dynamic ride-sharing, where passengers are matched in real-time. Ride-sharing, though, results to some passenger disruption due to loss in flexibility, security concerns, etc. Compensation comes in the form of monetary incentives, as it allows passengers to share the travel expenses, and thus reduce the cost. Ride-sharing companies account for a plethora of factors, like current demand, prime time pricing, the cost of the route without ride-sharing, the likelihood of a match, etc. Yet, a fundamental factor of the cost is the traveled distance.

In this test case, we attempt to maximize the total distance saved, by matching taxi requests of high overlap (Fig. 2). The setting is inherently an *on-line* setting, as a matching algorithm is unaware of the requests that will appear in the future and needs to make decisions before requests 'expire' (similar to [Ashlagi *et al.*, 2018]). ALMA fits such a scenario well, as it involves large-scale matchings under dynamic demand, it is highly decentralized, and partially observable.

Setting We use a dataset[2] of all taxi requests (ρ) in New York City during one week (34077 requests). The data include pickup and drop-off times, and geolocations. Requests

[2]kaggle: /debanjanpaul/new-york-city-taxi-trip-distance-matrix/

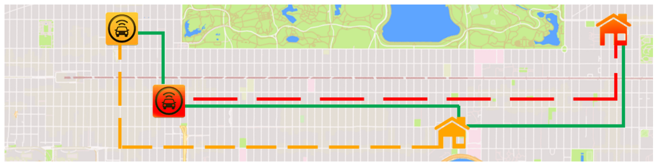

Figure 2: Example of the studied scenario. There are two passengers (yellow and red) with high overlap routes. Each can drive on their own to their respective destinations (dashed yellow and red line respectively), or share a ride (green line) and reduce travel costs.

appear (become open) at their respective pickup time, and wait k_ρ time-steps to find a match. Let a time-step be one minute. After k_ρ time-steps we call request ρ, critical. If a critical request is not matched, we assume they drive off to their destination in a single passenger ride. Let *open*, *critical* denote the sets of open, and critical requests respectively, and let $current = open \cup critical$. To compute k_ρ, we assume the following: There is a minimum $minW$, and a maximum $maxW$ waiting time set by the ride-sharing company, i.e., $minW \leq k_\rho \leq maxW, \forall \rho$. Moreover, since each passenger specifies his destination in advance, we can compute the trip time (l_ρ). Assuming people are willing to wait proportional to their trip time, let $k_\rho = q \times l_\rho$, where $q \in [0, 1]$. The parameters $minW$, $maxW$, and q can be set by the ride-sharing company. We report results on different values for all of the above parameters. For each pair ρ_1, ρ_2 of requests, we compute the driving distance ($d_{\rho_1,\rho_2} = \min$ of all possible combinations of driving between ρ_1, ρ_2's pickup and drop-off) that would be traveled if ρ_1, ρ_2 are matched, i.e., if they share the same taxi. Subsequently, the utility of matching ρ_1 to ρ_2 (distance saved) is $u_{\rho_1}(\rho_2) = d_{\rho_1,\rho_2}$ (km). Given the on-line nature of the setting, it might be beneficial to use the following *non-myopic heuristic*: avoid matching low utility pairs, as long as the requests are not critical, since more valuable pairs might be presented in the future. Thus, if $u_{\rho_1}(\rho_2) < d_{min}$, and $\rho_1, \rho_2 \notin critical$, we do not match ρ_1, ρ_2. In what follows, we select for each algorithm and for each simulation the value $d_{min} \in \{0, 500, 1000, 1500, 2000\}$ that results in the highest score. To compute the actual trip time, and driving distance, we have used the Open Source Routing Machine (project-osrm.org), which computes shortest paths in road networks.

Benchmarks Each request runs ALMA independently. ALMA waits until the request becomes critical, and then matches it by running Alg. 1, where $\mathcal{N} = critical$, and $\mathcal{R} = current$. In this non-bipartite scenario, if an agent is matched under his dual role as a resource, he is immediately removed. As we explained earlier, it is infeasible for an on-line algorithm to compute the optimal matching over the whole period of time. Instead, we consider *just-in-time* and *in batches* optimal solutions. Specifically, we compare to the following [Agatz *et al.*, 2011] [Ashlagi *et al.*, 2018]:

(1) **Just-in-time Max Weight Matching (JiTMWM):** Waits until a request becomes critical and then computes a maximum-weight matching of all the current requests.

(2) **Batching Max Weight Matching (BMWM):** Waits x time-steps and then computes a maximum-weight matching of all the current requests.

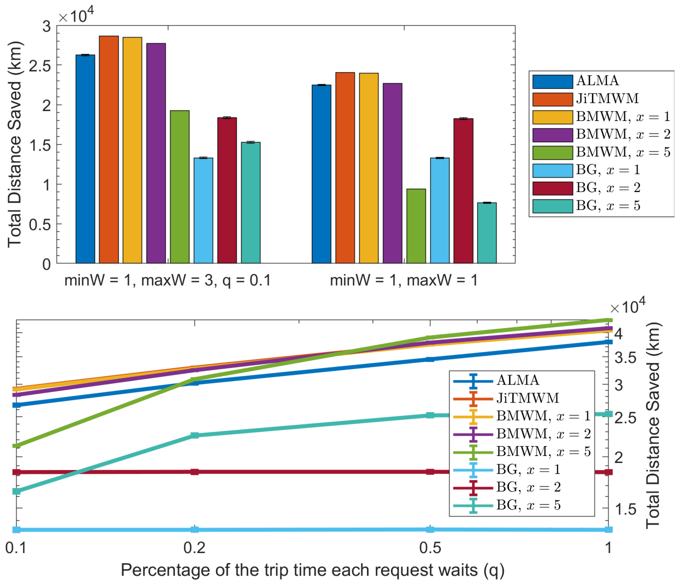

Figure 3: Total distance saved (km) for various values of $minW$, $maxW$, q. (top, left) Pragmatic scenario, (top, right) Requests become critical in one time-step, (bottom) Various levels of waiting time (q) while $minW = 0, maxW = \infty$, (double log. scale).

$(1, 3, 0.1)$	$(1, 1, -)$	$(0, \infty, 0.1)$	$(0, \infty, 0.5)$	$(0, \infty, 1.0)$
0.79	0.85	0.78	0.69	0.67

Table 1: The Empirical Competitive Ratio of ALMA, for different values of $(minW, maxW, q)$.

(3) **Batching Greedy (BG):** Waits x time-steps and then greedily matches current requests (ties are broken randomly). Unmatched open requests are removed.

We have also computed the off-line optimal matching and report the *empirical competitive ratio*, i.e., the ratio of the social welfare of the on-line algorithm over the welfare of the optimal, as is common in the literature of competitive analysis [Borodin and El-Yaniv, 2005].

Efficiency Fig. 3 presents the total distance saved (km) for various values of $minW, maxW$, and q. ALMA loses 8.3% of SW in the pragmatic scenario of Fig. 3 (top, left), and 6.5% when the requests become critical in just one time-step (Fig. 3 (top, right)). If no bounds are placed on the minimum and maximum waiting time (i.e., $minW = 0, maxW = \infty$), ALMA exhibits loss of $8 - 11.5\%$ (Fig. 3, bottom). The above are compared to the best performing benchmark on each scenario (JiTMWM, or BMWM). Moreover, it significantly outperforms every greedy approach. In the first scenario the BGs lose between $35.8 - 53.5\%$, in the second between $24 - 68.2\%$, and in the third between $31.5 - 69\%$.

It is worth noting that ALMA requires just a broadcast of a single bit to indicate the occupancy of a resource, while the compared approaches require either *message exchange* for sharing the utility table, or the use of a *centralized authority*. For example, the greedy solution would require message exchange to communicate users' preferences and resolve collisions in a decentralized setting, and every batching approach would require a common centralized synchronization clock.

Table 1 presents the empirical competitive ratio for one day. As we can see, even in the extreme, unlikely scenarios where we assume that people would be willing to wait for more than 10 or 20 minutes (large q), ALMA achieves high relative efficiency, compared to the off-line (infeasible) benchmark. These scenarios are favorable for the off-line optimal, because requests stay longer in the system and therefore the algorithm takes more advantage of its foreseeing capabilities. In particular, ALMA achieves an empirical competitive ratio of 0.67 for $q = 1$ and even better ratios for more realistic scenarios (as large as 0.85). In spite of the unpredictability of the on-line setting, and the dynamic nature of the demand, ALMA is consistently able to exhibit high performance, in all of the employed scenarios.

4 Conclusion

Algorithms for solving the assignment problem, whether centralized or distributed, have runtime that increases with the total problem size, even if agents are interested in a small number of resources. Thus, they can only handle problems of some bounded size. Moreover, they require a significant amount of inter-agent communication. Humans on the other hand are routinely called upon to coordinate in large scale, and under dynamic and unpredictable demand. Inspired by human behavior, we have introduced a novel anytime heuristic (ALMA) for weighted matching. ALMA is decentralized, requires only partial feedback, and has *constant* in the total problem size running time, under reasonable assumptions on the preference domain of the agents. The presented results provide an empirical proof of the high quality of the achieved solution in a variety of scenarios (synthetic and *real* data, time constraints, on-line settings). As autonomous agents proliferate (IoT devices, intelligent infrastructure, autonomous vehicles, etc.), having robust algorithms that can scale to hundreds of thousands of agents is of utmost importance.

References

[Agatz *et al.*, 2011] Niels A.H. Agatz, Alan L. Erera, Martin W.P. Savelsbergh, and Xing Wang. Dynamic ridesharing: A simulation study in metro atlanta. *Procedia-Social and Behavioral Sciences*, 2011.

[Ashlagi *et al.*, 2018] Itai Ashlagi, Maximilien Burq, Chinmoy Dutta, Patrick Jaillet, Amin Saberi, and Chris Sholley. Maximum weight online matching with deadlines. *arXiv preprint arXiv:1808.03526*, 2018.

[Bertsekas, 1979] Dimitri P. Bertsekas. A distributed algorithm for the assignment problem. *Lab. for Information and Decision Systems Working Paper, MIT*, 1979.

[Borchardt and Jacobi, 1865] Carl W. Borchardt and Carl G.J. Jacobi. De investigando ordine systematis aequationum differentialium vulgarium cujuscunque. *Journal für die reine und angewandte Mathematik*, 64:297–320, 1865.

[Borodin and El-Yaniv, 2005] Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. cambridge university press, 2005.

[Bürger *et al.*, 2012] Mathias Bürger, Giuseppe Notarstefano, Francesco Bullo, and Frank Allgöwer. A distributed simplex algorithm for degenerate linear programs and multi-agent assignments. *Automatica*, 2012.

[Charness and Rabin, 2002] Gary Charness and Matthew Rabin. Understanding social preferences with simple tests*. *The Quarterly Journal of Economics*, 2002.

[Cheng and Nguyen, 2011] Shih-Fen Cheng and Thi Duong Nguyen. Taxisim: A multiagent simulation platform for evaluating taxi fleet operations. In *IEEE/WIC/ACM*, 2011.

[Cigler and Faltings, 2013] Ludek Cigler and Boi Faltings. Decentralized anti-coordination through multi-agent learning. *JAIR*, 2013.

[Danassis and Faltings, 2019] Panayiotis Danassis and Boi Faltings. Courtesy as a means to coordinate. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '19, 2019.

[Danassis *et al.*, 2019] Panayiotis Danassis, Aris Filos-Ratsikas, and Boi Faltings. Anytime heuristic for weighted matching through altruism-inspired behavior. *arXiv preprint arXiv:1902.09359*, 2019.

[Dantzig, 1990] George B. Dantzig. *Origins of the simplex method*. ACM, 1990.

[Edmonds and Karp, 1972] Jack Edmonds and Richard M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM*, 1972.

[Edmonds, 1965] Jack Edmonds. Maximum matching and a polyhedron with 0, 1-vertices. *Journal of research of the National Bureau of Standards B*, 1965.

[Elkin, 2004] Michael Elkin. Distributed approximation: A survey. *SIGACT News*, 35(4):40–57, December 2004.

[Fehr and Rockenbach, 2004] Ernst Fehr and Bettina Rockenbach. Human altruism: economic, neural, and evolutionary perspectives. *Current Opinion in Neurobiology*, 14(6):784 – 790, 2004.

[Geng and Cassandras, 2013] Yanfeng Geng and Christos G. Cassandras. New "smart parking" system based on resource allocation and reservations. *IEEE Transactions on Intelligent Transportation Systems*, 2013.

[Gintis, 2000] Herbert Gintis. Strong reciprocity and human sociality. *Journal of theoretical biology*, 2000.

[Giordani *et al.*, 2010] Stefano Giordani, Marin Lujak, and Francesco Martinelli. A distributed algorithm for the multi-robot task allocation problem. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer, 2010.

[Gunn and Anderson, 2013] Tyler Gunn and John Anderson. Dynamic heterogeneous team formation for robotic urban search and rescue. *Procedia Computer Science*, 2013. The 4th Int. Conf. on Ambient Systems, Networks and Technologies (ANT 2013), the 3rd Int. Conf. on Sustainable Energy Information Technology (SEIT-2013).

[Ismail and Sun, 2017] Sarah Ismail and Liang Sun. Decentralized hungarian-based approach for fast and scalable task allocation. In *2017 Int. Conf. on Unmanned Aircraft Systems (ICUAS)*, 2017.

[Kuhn *et al.*, 2016] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. Local computation: Lower and upper bounds. *J. ACM*, 63(2):17:1–17:44, March 2016.

[Kuhn, 1955] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics*, 1955.

[Lewis, 2008] David Lewis. *Convention: A philosophical study*. John Wiley & Sons, 2008.

[Lovász and Plummer, 2009] László Lovász and Michael D. Plummer. *Matching theory*, volume 367. American Mathematical Soc., 2009.

[Munkres, 1957] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 1957.

[Nguyen *et al.*, 2017] Duc Thien Nguyen, Akshat Kumar, and Hoong Chuin Lau. Collective multiagent sequential decision making under uncertainty. In *AAAI*, 2017.

[Nowak and Sigmund, 2005] Martin A. Nowak and Karl Sigmund. Evolution of indirect reciprocity. *Nature*, 2005.

[Ollivier, 2009] François Ollivier. Looking for the order of a system of arbitrary ordinary differential equations. *Applicable Algebra in Engineering, Communication and Computing*, 2009.

[Papadimitriou and Steiglitz, 1982] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

[Simon, 2016] Jay Simon. On the existence of altruistic value and utility functions. *Theory and Decision*, 2016.

[Stone *et al.*, 2010] Peter Stone, Gal A. Kaminka, Sarit Kraus, and Jeffrey S. Rosenschein. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *AAAI*, 2010.

[Su, 2015] Hsin-Hao Su. Algorithms for fundamental problems in computer networks. 2015.

[Talebi, 2013] Mohammad Sadegh Talebi. Uncoupled learning rules for seeking equilibria in repeated plays: An overview. *CoRR*, abs/1310.5660, 2013.

[Varakantham *et al.*, 2012] Pradeep Varakantham, Shih-Fen Cheng, Geoff Gordon, and Asrar Ahmed. Decision support for agent populations in uncertain and congested environments. 2012.

[Varakantham, 2016] Pradeep Varakantham. Sequential decision making for improving efficiency in urban environments. 2016.

[Zavlanos *et al.*, 2008] Michael M. Zavlanos, Leonid Spesivtsev, and George J Pappas. A distributed auction algorithm for the assignment problem. In *Decision and Control, 2008*. IEEE, 2008.