Fair subdivision of Multi-Robot Tasks

Juan Camilo Gamboa Higuera[1], and Gregory Dudek[1].

Abstract— **We study the problem of distributing a single global task between a group of heterogeneous robots. We view this problem as a fair division game. In this setting, every robot defines a *preference* function over parts of the task according to its sensing and motion capabilities. These preferences are described by density functions over the task. With such interpretation, we want to find an allocation of the global task that maximizes the probability of task completion. We first formulate the task distribution problem as a fair subdivision problem and provide a centralized algorithm to compute the allocations for each robot. We provide a complexity analysis and computational results of the algorithm.**

I. INTRODUCTION

We consider the problem of coordinating multiple heterogeneous robots to perform a single global task, where a task is an activity to be carried out over an operating region. Examples of such tasks include cleaning duties inside a building (a coverage task), exploration and mapping of an unknown environment and searching for a target inside a bounded region and aerial surveillance. We consider the case where progress on the subtasks can be carried out in parallel and independently between the robots.

Coordination of robots through task allocation is motivated by parallelization of workload and robustness to partial failures [1]. Task subdivision is also motivated by the heterogeneity in robot teams [2], where the assignment of subtasks is dependent on robot abilities. Some of the existing strategies for task allocation consider the tasks as single atomic units [3], [4], or as a hierarchy of such units [5]. These single atomic units are usually single points in space or are predefined by a designer or central planner. In this work we treat tasks as subsets of the global task. In this sense, if the global task is a coverage task, subtasks are subregions instead of single points.

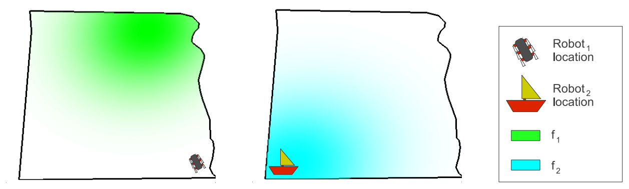
Our work is related to environment partitioning and load balancing through locational optimization [6], [7], [8], [9], in particular to the equitable partitioning of environments [10], [11]. A difference with those approaches is that, in our case, the partition of an environment is based on multiple arbitrary measures, different for each robot. This can arise naturally when different types of robots are jointly performing a task (e.g. planes and boats), or when vehicles have different performance characteristics (e.g. worn vs fresh batteries or tires). We are interested in introducing the heterogeneity of the robot team to drive the task subdivision process. Heterogeneity of the robot team arises from the dependence of the performance of sensing and locomotion on the characteristics of the environment. We encode this dependence with a scoring function for each robot. The goal is to find subdivisions of the task that maximize the accumulated score of all robots, while balancing the workload among them. The accumulated scores represent the utility of assigning work to each robot.

With this goal, the task subdivision problem can be interpreted as one of fair division: we want to maximize the utility of the task division, while allowing each robot to contribute to the task as much as possible. Fair division theory treats problems of dividing an object between n interested players satisfying some optimality criterion [12], [13]. The contribution of this work is the formulation of a task distribution strategy for heterogeneous robots as a problem of fair division. In the context of a coverage task, we present a centralized algorithm that finds globally optimal solutions. We show computational results of the impact on the performance of our proposed algorithm while varying parameters such as the number of robots, resolution (cell-size) of the preference functions and similarity between preferences.

II. BACKGROUND

Fair division is the problem of dividing an object among n interested players, so that each believes it has received a *fair* share. The players might have different opinions on the value of different parts of the object; it is this disagreement what makes the problem interesting and applicable to our task subdivision problem. We define an allocation as optimal and fair if every robot receives the same utility and this utility is maximized. This agrees with the properties of solutions to a max-min fair division game [12], [13]. Figure 1 depicts an example fair division scenario of a coverage task with heterogeneous robots. An underwater robot and a surface vehicle are deployed in an unstructured environment. In a search task in such a context, performance improvements are obtained by assigning the best suited robot for each region of the environment; e.g. the boat to regions where the sea bottom is visible and the underwater robot to regions that are deep enough to move freely. Each vehicle needs to be allocated to a (possibly connected) region where it is best suited to the task. An example optimal allocation is shown in Figure 2(b) . We can state this problem formally as follows. Let the set T represent the task to be divided. Let $u = \{u_1, u_2, \dots, u_n\}$ be a set of utility functions $u_i : \mathcal{T} \rightarrow \mathbb{R}$, where $\mathcal{T}$ is a σ-algebra of T. Define a k-partition of T as a set $\boldsymbol{A} = \{A_1, A_2, \dots, A_k\}$ of elements from $\mathcal{T}$. Each u_i is *countably additive*, i.e. $u_i(A_a \cup A_b) = u_i(A_a) + u_i(A_b)$

[1]School of Computer Science, McGill University {gamboa, dudek} @cim.mcgill.ca



(a) (b)

Fig. 1. Example fair division scenario between two robots: an underwater robot (a) and a boat (b). The object to be distributed is a territory depicted by the thick black boundary. For each robot, there is a function f_i that defines how well suited are the robots for performing a given task at any point

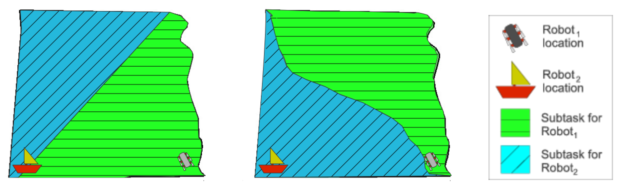
for any pair of disjoint sets $A_a, A_b \subseteq T$. In our setting we consider utilities that are non atomic; i.e. for every set $A \subset T$ there exists another set $B \subset A$ for which $u_i(A) > u_i(B) > 0$. Define the measure v with respect to which each u_i is absolutely continuous. From the Radon-Nikodym theorem [14], there must exist a function f_i for each u_i such that $u_i(A) = \int_A f_i \mathrm{d}v$, where v is a measure with respect to which all u_i are absolutely continuous. In a coverage task, the area of the region is one such measure. The functions f_i represent a density distribution of the value for robot i over the subsets of T. Thus, we can interpret each f_i as the *preference* of robot i. The simplest optimization criterion is to maximize the sum of utilities; an optimal partition is an n-partition of T such that $\sum u_i(A_i)$ is maximized. For instance, let $f^* = \max f_i$, an upper envelope of all the measures. Then every n-partition of T satisfies

$$\sum_{i=1}^{n} u_i(A_i) = \sum_{i=0}^{n} \int_{A_i} f_i \mathrm{d}v \leq \sum_{i=0}^{n} \int_{A_i} f^* \mathrm{d}v = \int_T f^* \mathrm{d}v \tag{1}$$

Thus the partition that maximizes the sum of utilities is obtained by assigning every subset of T to the player that values it the most. This solution is optimal in that it maximizes the global value, but it might assign all of T to a single robot if $u_i(A) > u_j(A) \quad \forall i, j$. To avoid this, we can use an optimization criterion which finds partitions by maximizing the minimum utility among players. This can be stated as the following optimization problem.

$$\begin{aligned} \max_{A} \quad & u_{\min} \\ \text{subject to} \quad & u_{\min} \leq \int_{A_i} f_i \mathrm{d}v \qquad \forall i \\ & A_i \cap A_j = \emptyset \qquad \forall i, j \\ & \bigcup_{i=1}^{n} A_i = T \end{aligned} \tag{2}$$

In the case every u_i is non atomic, Dubins and Spanier [12] proved that the space of allocations is compact and convex, therefore optimal max-min partitions must exist. They also proved that optimal partitions are equitable when all the utilities are absolutely continuous with respect to each other; i.e., if all the players assign a non zero value to every possible subset of T, then the optimal partition will allocate every player the same share (see Corollaries 1.1 and 1.2 in [12]). Dall'Aglio [13] also showed that the space of allocations is convex, observed the optimal allocations are Pareto optimal



(a) (b)

Fig. 2. Example of fair and optimal allocations for the scenario in Figure 1. A fair allocation (a) guarantees at least half of the maximum utility, while an optimal partition (b) might result in every player obtaining their first choice

and showed that each element A_i of an optimal partition of T corresponds to a dual optimal variable. Dall'Aglio and Di Luca [15] provided an algorithm that solves the dual problem by using the subgradient method [16] and exploiting the fact that optimal allocations are equitable when the utilities are mutually absolutely continuous. In the next section we will provide a formulation of the task subdivision problem that fits nicely with the fair subdivision background. The benefit of using this approach is that we do not need to make any assumptions about the shape of the object we are trying to divide; i.e. T does not need to be simply connected or polygonal.

III. PARTITIONING AN ENVIRONMENT AMONG HETEROGENEOUS ROBOTS

A. Problem statement

A task $T \in \mathbb{T}$ is assigned to a group of n robots R. The space of tasks considered here is $\mathbb{T} = \mathbb{R}^d$, where d is the number of dimensions of the configuration space of the robots. Each robot defines a performance score for each subset of T as the function $f_i : \mathcal{T} \rightarrow \mathbb{R}$. We consider the task of collecting sensor information over a bounded region. Every robot has both a coverage speed profile $s_i(x)$, which represents its dependence of speed on terrain, and a sensor quality profile $q_i(s_i(x), x)$, which represents the dependence of its sensor performance on location and speed. The speed and quality scores for each location are dependent on each other: if the speed increases then the sensor quality will decrease, and to improve the quality of the collected data, the robot must reduce its speed.The objective is to maximize the speed of coverage by the group of robots and the quality of collected data, while guaranteeing a maximum amount of work done. Let $s_{i,max}$ be the maximum speed of coverage that guarantees a minimum quality $q_{i,min} = q_i(s_{i,max}(x), x)$. We assume that the time it will take a robot to cover any point $x \in T$ is,

$$\mathrm{d}t(x) = \frac{\mathrm{d}T}{s_{i,max}(x) + \epsilon_s}$$

where $\mathrm{d}T \in T$ is an infinitesimal task element and $\epsilon_s << 1$ is a constant added to avoid division by zero. Given the task of collecting sensor data over a region $T \in \mathbb{R}^d$ and a group of n robots, each with its maximum speed profile $s_{i,max}(x)$, we want every part of T to be assigned to the robots in order to minimize the combined time. Formally, let $\boldsymbol{A} = \{A_1, \ldots, A_n\}$ be the allocations of T to each robot $1 \leq i \leq n$. Thus we want to solve the following optimization

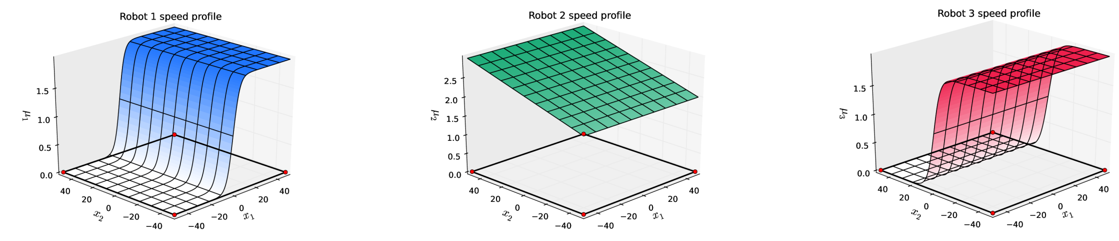


Fig. 3. Robot speed profiles for a coverage task in a square region: (a) $s_1(x) = \frac{2}{1+e^{-40x_1+20}} \quad m^2/s$, (b) $s_2(x) = 2 + \frac{1}{100}x_2 \quad m^2/s$ and (c) $s_3(x) = \frac{2}{1+e^{30x_2-15}} \quad m^2/s$

problem,

$$\begin{aligned} \min_{A} \quad & \sum_{i=1}^{n} \int_{A_i} \frac{1}{s_{i,max}(x) + \epsilon_s} \mathrm{d}T \\ \text{subject to} \quad & A_i \cap A_j = \emptyset \qquad \forall i, j \in R \\ & \bigcup_{i=1}^{n} A_i = T \end{aligned} \tag{3}$$

We assume that the robots are holonomic and are able to compute plans that avoid visiting locations more than once. Therefore the time spent in covering one region is the sum of the times spent at every point.

B. *Proposed methodology*

To find solutions to this problem, we require that for every $x \in T$, at least one i exists such that $s_{i,max}(x) > 0$, otherwise x should be removed from T. We define the indicator function $\mathbb{1}_i$

$$\mathbb{1}_i(x) = \begin{cases} 1, & \text{if } x \in A_i \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

Letting $f_i(x) = -\frac{1}{s_{i,max}(x)}$, the problem (Eq. 3) is equivalent to

$$\begin{aligned} \max_{\mathbb{1}_1,..,\mathbb{1}_n} \quad & \sum_{i=1}^{n} \int_{T} \mathbb{1}_i(x) f_i(x) \mathrm{d}T \\ \text{subject to} \quad & \sum_{i=1}^{n} \mathbb{1}_i(x) = 1 \qquad \forall x \in T \\ & \mathbb{1}_i(x) \in \{0, 1\} \qquad \forall i \in R, x \in T \end{aligned} \tag{5}$$

which is in the form of a max-sum fair division problem [12], implying that the optimal solution is to assign every point $x \in T$ to the fastest robot.

In this case, the robot utilities correspond to the negative of an estimate of the time to cover A_i. Since this allocation may not be balanced, we also define the max-min problem below (Eq. 6),

$$\begin{aligned} \max_{\mathbb{1}_1,..,\mathbb{1}_n} \quad & u_{\min} \\ \text{subject to} \quad & u_{\min} \leq \int_{T} \mathbb{1}_i(x) f_i(x) \mathrm{d}T \qquad \forall i \in R \\ & \sum_{i=1}^{n} \mathbb{1}_i(x) = 1 \qquad \forall x \in T \\ & \mathbb{1}_i(x) \in \{0, 1\} \qquad \forall i \in R, x \in T \end{aligned} \tag{6}$$

where $u_{\min}$ is the least accumulated score between the robots. To find optimal solutions of the problems (Eq. 5) and (Eq. 6), we solve the dual optimization problem which is convex, has a finite number of optimization variables and its solutions can be transformed easily into primal solutions [13], [17]. We can show, by using Lagrangian duality, that the dual of this problem is

$$\begin{aligned} \min_{\boldsymbol{\lambda}} \quad & \int_{T} \max_{i} \{\lambda_i f_i(x)\} \mathrm{d}T \\ \text{subject to} \quad & \sum_{i=1}^{n} \lambda_i = 1 \qquad \forall i \in R, x \in T \\ & \lambda_i \geq 0 \qquad \forall i \in R \end{aligned} \tag{7}$$

This dual problem is convex, so we can use an iterative search algorithm to find the optimal values for λ_i. Let $g(\boldsymbol{\lambda}) = \int_T \max_i \{\lambda_i f_i(x)\} \mathrm{d}T$, i.e. the objective function we are trying to minimize. Let $D_{\boldsymbol{\lambda}}$ be its domain. The function g is equivalent to the one in the minimization step of the algorithm in Section 6 of [13]. Since the objective function in (Eq. 7) is not differentiable we may use a subgradient search method [16]. Such method consists of the following update rule,

$$\boldsymbol{\lambda}^{(t+1)} = \boldsymbol{\lambda}^{(t)} - \alpha_t \boldsymbol{\gamma}(\boldsymbol{\lambda}^{(t)}) \tag{8}$$

where t is the iteration index and α_t is the step size. Let $\boldsymbol{\lambda}^{(t)}$ be an interior point of $D_{\boldsymbol{\lambda}}$. The vector $\boldsymbol{\gamma}(\boldsymbol{\lambda}^{(t)})$ is called a subgradient of g around $\boldsymbol{\lambda}^{(t)}$ if it satisfies

$$g(\boldsymbol{\lambda}^{(t+1)}) - g(\boldsymbol{\lambda}^{(t)}) \geq (\boldsymbol{\lambda}^{(t+1)} - \boldsymbol{\lambda}^{(t)})^T \boldsymbol{\gamma}(\boldsymbol{\lambda}^{(t)}) \tag{9}$$

for all $\boldsymbol{\lambda} \in D_{\boldsymbol{\lambda}}$; i.e., the subgradient defines a hyperplane that supports the set of feasible solutions around the point $\boldsymbol{\lambda}$. By defining the max-sum indicator function $\mathbb{1}_{i,max}^{(t)}$ as

$$\mathbb{1}_{i,max}^{(t)}(x) = \begin{cases} 1, & \text{if } \lambda_i^{(t)} f_i(x) \geq \lambda_j^{(t)} s_j(x) \qquad \forall j \neq i \\ 0, & \text{otherwise} \end{cases}$$

it can be easily shown that the vector

$$\gamma_i^{(t)} = \int_T \mathbb{1}_{i,max}^{(t)}(x) f_i(x) \tag{10}$$

is a subgradient that satisfies (Eq. 9). To deal with the constraint $\sum_{i=1}^{n} \lambda_i = 1$, the subgradient method must be modified by including a normalizing constant ν,

$$\sum_{i=1}^{n} \lambda_i^{(t+1)} = \sum_{i=1}^{n} (\lambda_i^{(t)} - \alpha_t \gamma_i^{(t)} + \nu) = 1$$

from which it is easy to see that $\nu = -\alpha_t \frac{\sum_{i=1}^{n} \gamma_i^{(t)}}{n}$. To keep every $\lambda_i^{(t+1)}$strictly positive then the step length α_t should satisfy

$$\alpha_t < \lambda_i^{(t)} \bigg/ \left(\gamma_i^{(t)} - \frac{\sum_{i=1}^{n} \gamma_i^{(t)}}{n} \right)$$

for the cases when $\gamma_i^{(t)} > \left(\sum_{i=1}^{n} \gamma_i^{(t)} \right) / n$. This constraint provides an upper limit for the step length at any time step. This constraint is enforced in step 21 of the Algorithm 1 by multiplying the upper limit by a constant $\eta < 1$. The choice of the step length α_t for (Eq. 8) determines the rate of convergence and the accuracy of the solution. In particular, as shown in [16], if we want the method to converge to the optimal solution then α_t should follow a *diminishing step length rule*. Such rule requires the step size to satisfy

$$\alpha_t = \frac{\alpha'_t}{\|\gamma_i^{(t)}\|}$$
$$\alpha'_t \geq 0, \quad \lim_{t \to \infty} \alpha'_t = 0, \quad \sum_{i=1}^{\infty} \alpha'_t = \infty \tag{11}$$

Sequences of the form $\alpha'_t = \alpha'_0 / t^p$, with $0 < p \leq 1$, satisfy (Eq. 11). For our experiments we set $p = 1$. Subgradient methods are usually used without any formal stopping criteria since these are problem specific. Dall'Aglio and Di Luca [15] provide two stopping criteria for the subgradient method for fair division: until $\max_{j \in R} u_j(A_j) - \min_{j \in R} u_j(A_j) < \epsilon$ or by checking the distance between geometric upper and lower bounds on the optimal set. Both are based on the observation that the optimal solutions for the fair division problem are equitable when the utilities u_i are normalized, mutually absolutely continuous and linearly independent. The utilities in our case are not normalized, so the stopping criterion is

$$\left(\max_{j \in R} u_j - \min_{j \in R} u_j \right) \bigg/ \max_{j \in R} u_j < \epsilon$$

where $u_j = \int_T \mathbb{1}_i f_i(x) \mathrm{d}T$.

Algorithm 1 describes the procedure to compute the optimal time solution for the multi robot problem described in Section III-A. This algorithm assumes that every robot has enough energy to complete its part and ensures that the spatial load is balanced between robots. Since the subgradient is not necessarily a descent direction we must keep track of the best solution found so far,

$$\boldsymbol{\lambda}_{best}^{(t+1)} = \arg\min_{\boldsymbol{l}} g(\boldsymbol{l}), \quad \boldsymbol{l} \in \{\boldsymbol{\lambda}_{best}^{(t)}, \boldsymbol{\lambda}^{(t+1)}\} \tag{12}$$

which is done in steps 18 to 20 of Algorithm 1.

IV. Simulation Results

For the task of covering a square region of $100m \times 100m$ using three robots with speed profiles as depicted in Figure 3, we show an example result of executing the algorithm in Figure 4; with an integration cell size of $0.0625m$, and an error tolerance ϵ of 0.000001. For each robot i, $-u_i$ is the expected time it would take to complete its assigned subtask.

This same algorithm is applicable for dividing non-convex tasks: it will still find time-optimal solutions as long as the densities f_i are absolutely continuous with respect to each other. Nevertheless, additional considerations are necessary for the allocations, such as connectedness and reachability. It must be noted that this algorithm may not converge if $u_i(A) = u_j(A)$ for some $i, j \in R$ and a set A of measure greater than 0. In this case, the algorithm will oscillate between assigning A to i and j. Figure 6 shows how the error decreases non monotonically, with the oscillations at the end demonstrating the aforementioned effect.

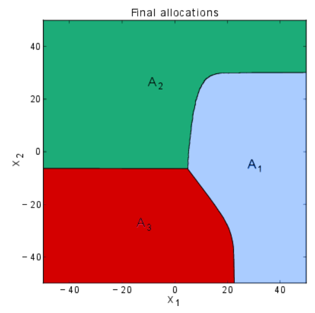


Fig. 4. Final allocation of regions A_i to each robot $i \in R$. The allocated areas are $a_1 = 2966.8m^2$, $a_2 = 4087.2m^2$ and $a_3 = 2946.0m^2$. The utilities are $u_1 = -1487.2s$, $u_2 = -1487.2s$ and $u_3 = -1487.2s$.

Algorithm 1: Min-max time task subdivision

```
Inputs: A set $T \in \mathbb{R}^d$ which represents the task to be completed;
        Speed profiles $s_{i,\max}$ for each robot $i \in R$;
        A sequence of $\alpha_t$ that satisfies (Eq. 11);
        An error tolerance $\epsilon > 0$ and a maximum number of
    iterations $t_{\max}$.
Output: A function $K_{\max} : T \to R$ that encodes the allocation
    $A = \{A_1, \dots, A_n\}$
        which satisfies
          $\bigcup_{i=1}^{n} A_i = T$,
          $A_i \cap A_j = \emptyset \quad \forall i \neq j$,
        and minimizes the maximum time spent by any robot;
 1: $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_n)$, $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_n)$
 2: $\lambda_i \leftarrow 1/n \quad \forall i \in R$
 3: $t \leftarrow 1$, $\epsilon_t \leftarrow \infty$
 4: $f_i(x) = -\frac{1}{s_{i,\max}(x)} \qquad \forall x \in T, i \in R$
 5: while $t \leq t_{max} \wedge \epsilon_t \leq \epsilon$ do
 6:     for each $x \in T$ do
 7:         $K_{\max}(x) \leftarrow \arg\max_i \lambda_i f_i(x)$
 8:         for each $i \in R$ do
 9:             $\mathbb{1}_i(x) \leftarrow \begin{cases} 1, & \text{if } K_{\max}(x) = i \\ 0, & \text{otherwise} \end{cases}$
10:     $g_t \leftarrow \sum_{i=1}^{n} \int_T \mathbb{1}_i(x) \lambda_i f_i \mathrm{d}T$
11:     if $g_t \leq g_{best}$ then
12:         $g_{best} \leftarrow g_t$
13:         $K_{best} \leftarrow K_{\max}$
14:     for each $i \in R$ do
15:         $\gamma_i \leftarrow \int_T \mathbb{1}_i(x) f_i \mathrm{d}T$
16:     $\bar{\gamma} \leftarrow (\sum_{i=1}^{n} \gamma_i)/n$
17:     $\bar{\boldsymbol{\gamma}} = (\bar{\gamma}, \dots, \bar{\gamma})_{1 \times n}$
18:     $\alpha \leftarrow \alpha_t$
19:     for each $i \in \{x \in R : \gamma_i^{(t)} > \bar{\gamma}\}$ do
20:         if $\alpha > \lambda_i^{(t)} / (\gamma_i^{(t)} - \bar{\gamma})$ then
21:             $\alpha \leftarrow \eta \lambda_i^{(t)} / (\gamma_i^{(t)} - \bar{\gamma})$
22:     $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} - \alpha (\boldsymbol{\gamma} - \bar{\boldsymbol{\gamma}})$
23:     $t \leftarrow t + 1$, $\epsilon_t \leftarrow (\max_{j \in R} u_j - \min_{j \in R} u_j) / \max_{j \in R} u_j$
24: return $K_{best}$
```

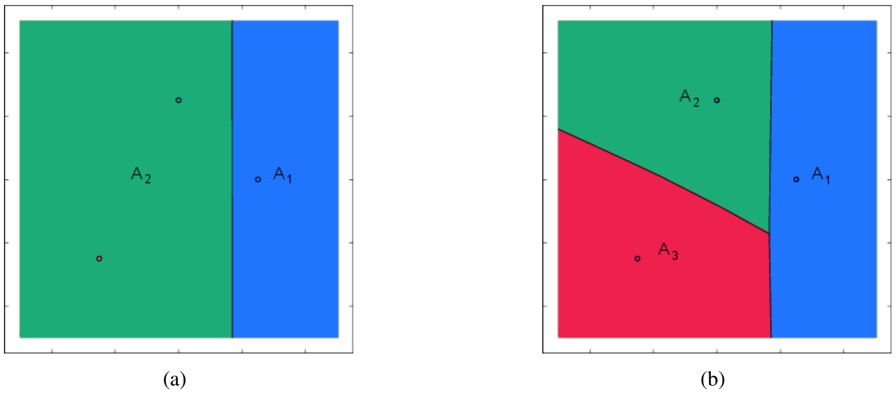


Fig. 5. Example of introducing a distance term on the utility densities f_i (a) Allocation with no distance term (b) Allocation with distance term weighted by $\kappa = 5e-6$. The dots denote the positions of the robots.

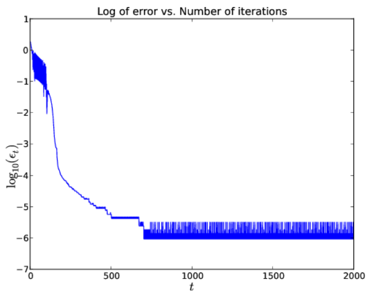


Fig. 6. Logarithmic plot of error vs. number of iterations

One way of dealing with possible conflicts is to assign every $x \in A$ uniformly at random to any robot i with $f_i(x) = \max_j f_i(x)$. If the task is defined in a metric space, then other properties of the allocations A_i can be used to resolve the conflicts such as connectedness and minimizing distances. In our example, the robot position can be represented in the task space. Thus, there exists a distance function between the robot position $p_i \in \mathbb{R}^d$ and every point $x \in T$. So for any conflicting set A we could replace every f_i with a

$$f_i^{'}(x) = f_i(x) - \kappa d_i(p_i, x)$$

where κ determines the influence of the distance factor on the solution. In such cases, the problem is that of finding generalized Voronoi tessellations of A. Figure 5 shows the case for three robots with speed profiles defined as

$$s_1(x) = 2/\left(1 + e^{-40x_1+20}\right) \quad m^2/s$$
$$s_2(x) = 1 \quad m^2/s$$
$$s_3(x) = 1 \quad m^2/s$$

with $\kappa = 0$ and $\kappa = 5e-6$. In the first case the final utilities are $u_1 = -1662.7$, $u_2 = -3337.5$ and $u_3 = 0$; the algorithm fails to split the task between the robots 2 and 3. By introducing the distance term the final utilities are $u_1 = -1667.1$, $u_2 = -1667.2$ and$u_3 = -1667.1$, balancing the load between the three robots.

A. *Complexity analysis of Algorithm 1*

The time complexity and accuracy of Algorithm 1 depend on discretization of T and the desired accuracy of the solution, ϵ. Let $|T|$ be the number of elements into which T is discretized. Each iteration of the subgradient update takes $n(|T|+1) + O_{int}(n+1)$ operations, where O_{int} is the number of operation in the numerical integration method used. For example, if integration is done by using a midpoint or trapezoidal rule over the discretization of T, then $O_{int} = |T|$ and the run time of each iteration is $O(n|T| + n + |T|)$. To obtain an estimate of the number of iterations until convergence we use the same analysis as in [16]. The error of the subgradient search is bounded by [16]

$$g(\boldsymbol{\lambda}_{best}) - g(\boldsymbol{\lambda}^*) \leq \left(K_2^2 + K_1^2 \sum_{i=1}^{t} \alpha_i^2\right) \Big/ 2\sum_{i=1}^{t} \alpha_i \quad (13)$$

where $\boldsymbol{\lambda}^*$ is a global optimum. The constants K_1 and K_2 are defined as follows. First, we note that $g(\boldsymbol{\lambda})$ is Lipschitz continuous since we assume every f_i is absolutely continuous. Therefore, the magnitude of the subgradient vector is bounded by some constant $K_1 \geq \|\boldsymbol{\gamma}\|$. This bound is attained when , i.e. when $u_i = u^* \quad \forall i$. Thus, $\|\boldsymbol{\gamma}\| \leq K_1 = \sqrt{n\ u^{*2}}$. The distance between the starting point $\lambda_i = 1/n$, $\forall i \in R$ and the optimal point $\boldsymbol{\lambda}^*$ is also bounded by a constant K_2. Since $D_{\boldsymbol{\lambda}}$ is the n-simplex, K_2 is the distance between the center of the n-simplex and one of its corners. Therefore,

$$\|\boldsymbol{\lambda}^{(1)} - \boldsymbol{\lambda}^*\|^2 \leq K_2^2 = \left(1 - \frac{1}{n}\right)^2 + \sum_{i=1}^{n} \left(\frac{1}{n}\right)^2 = 1 - \frac{1}{n}$$

The right hand side of this inequality is minimized when

$$\alpha_i = \frac{K_2}{K_1\sqrt{t}} \qquad \forall i$$

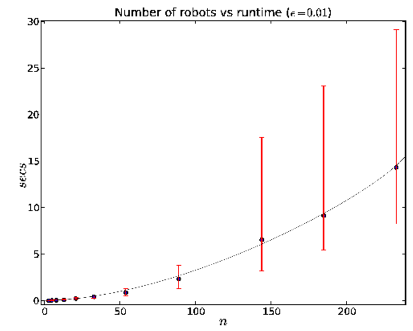
providing a bound on the error. Replacing this value in (Eq. 13) and setting $g(\boldsymbol{\lambda}_{best}) - g(\boldsymbol{\lambda}^*) = \epsilon$ yields

$$\epsilon \leq \frac{K_2^2 + K_1^2 t \left(\frac{K_2}{K_1\sqrt{t}}\right)^2}{2t\frac{K_2}{K_1\sqrt{t}}} = \frac{K_1K_2}{\sqrt{t}}$$

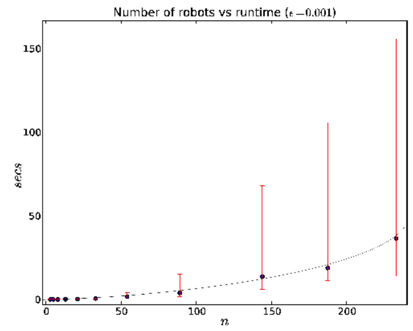
which means that the algorithm must be run for at least

$$\left(\frac{K_1K_2}{\epsilon}\right)^2 \approx n\left(\frac{u^*}{\epsilon}\right)^2$$

steps to guarantee the desired accuracy. Combining this number with the run time of each iteration, we have that the complexity of the algorithm is $O(n\left(\frac{u^*}{\epsilon}\right)^2(n(|T|+1) + O_{int}(n+1))$. For the case when the trapezoidal rule is used, the complexity is $O((u^*n)^2|T|/\epsilon)$. Figure 7 shows the increase in running time with increasing number of robots.



(a)



(b)

Fig. 7. Plot of the increase in runtime (in seconds) against the number of robots. The data points correspond to the average of 20 runs of the algorithm with 2, 3, 5, 8, 13, 21, 33, 54, 89, 144 and 233 robots, with $\epsilon = 0.01$ and $\epsilon = 0.001$. The dots are the mean running time in seconds, the bars show the minimum and maximum runtime for each value of n

V. CONCLUSIONS AND FUTURE WORK

In this work we presented an approach for subdividing and allocating a single global task between multiple heterogeneous robots. We have determined the computational complexity of the approach and illustrated its effectiveness. This approach deals with the heterogeneity of the robot team by letting the robots define a preference function over the task space. In the specific case of terrain coverage, we have illustrated how our approach can be used to find a suitable time-optimal task allocation that corresponds, as well, to a spatial decomposition. This same approach should be easily generalizable to $\mathbb{R}^n$, as long as integrals are computed efficiently; e.g. with a Monte Carlo algorithm. Such subdivision optimizes the time spent by any robot on its own allocation. These time estimates are useful for selecting meeting times and locations in a multi robot rendezvous problem [18]. Although the algorithm proposed in this work is able to find partitions of arbitrary speed profiles efficiently, there is no guarantee that the partition is the best one for real robots with non-holonomic constraints. In such cases, the assumption of additive utility function will not hold; i.e. depending on the motion model of the robots, some shapes of allocations will be preferred. We are currently working on a decentralized version of the algorithm that allows robots to compute their allocation without knowledge of the other robots, with updates to the allocations once new information becomes available.

REFERENCES

[1] G. Dudek, M. R. M. Jenkin, E. Milios, and D. Wilkes, "A taxonomy for multi-agent robotics," *AUTONOMOUS ROBOTS*, vol. 3, pp. 375–397, 1996.

[2] M. A. Hsieh, A. Cowley, J. F. Keller, L. Chaimowicz, B. Grocholsky, V. Kumar, C. J. Taylor, Y. Endo, R. C. Arkin, B. Jung, D. F. Wolf, G. S. Sukhatme, and D. C. MacKenzie, "Adaptive teams of autonomous aerial and ground robots for situational awareness: Field reports," *J. Field Robot.*, vol. 24, no. 11-12, pp. 991–1014, Nov. 2007. [Online]. Available: http://dx.doi.org/10.1002/rob.v24:11/12

[3] N. Atay and B. Bayazit, "Emergent task allocation for mobile robots," in *Proceedings of Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.

[4] L. Liu and D. Shell, "Large-scale multi-robot task allocation via dynamic partitioning and distribution," *Autonomous Robots*, pp. 1–17, 2012, 10.1007/s10514-012-9303-2. [Online]. Available: http://dx.doi.org/10.1007/s10514-012-9303-2

[5] R. M. Zlot and A. T. Stentz, "Market-based multirobot coordination for complex tasks," *International Journal of Robotics Research, Special Issue on the 4th International Conference on Field and Service Robotics*, vol. 25, no. 1, pp. 73–101, January 2006.

[6] S. Bhattacharya, N. Michael, and V. Kumar, "Distributed coverage and exploration in unknown non-convex environments," in *Proceedings of 10th International Symposium on Distributed Autonomous Robotics Systems*. Springer, 1-3 Nov 2010.

[7] A. Breitenmoser, M. Schwager, J.-C. Metzger, R. Siegwart, and D. Rus, "Voronoi coverage of non-convex environments with a group of networked robots," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, may 2010, pp. 4982 –4989.

[8] J. Cortes, "Coverage optimization and spatial load balancing by robotic sensor networks," *Automatic Control, IEEE Transactions on*, vol. 55, no. 3, pp. 749 –754, march 2010.

[9] L. Pimenta, V. Kumar, R. Mesquita, and G. Pereira, "Sensing and coverage for a network of heterogeneous robots," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, dec. 2008, pp. 3947 –3952.

[10] M. Pavone, A. Arsie, E. Frazzoli, and F. Bullo, "Distributed algorithms for environment partitioning in mobile robotic networks," *Automatic Control, IEEE Transactions on*, vol. 56, no. 8, pp. 1834 –1848, aug. 2011.

[11] J. G. Carlsson, "Dividing a territory among several vehicles," *INFORMS Journal on Computing*, 2011. [Online]. Available: http://joc.journal.informs.org/content/early/2011/10/17/ijoc.1110.0479.abstract

[12] L. E. Dubins and E. H. Spanier, "How to cut a cake fairly," *The American Mathematical Monthly*, vol. 68, no. 1, pp. pp. 1–17, 1961. [Online]. Available: http://www.jstor.org/stable/2311357

[13] M. Dall' Aglio, "The dubins-spanier optimization problem in fair division theory," *J. Comput. Appl. Math.*, vol. 130, no. 1-2, pp. 17–40, May 2001. [Online]. Available: http://dx.doi.org/10.1016/S0377-0427(99)00393-3

[14] W. Feller, *An Introduction to Probability Theory and Its Applications*, 3rd ed. New York: Wiley, 1968, vol. 1.

[15] M. Dall'Aglio and C. Di Luca, "Finding maxmin allocations in cooperative and competitive fair division," Fondazione Eni Enrico Mattei," Working Paper (available at http://arxiv.org/pdf/1110.4241.pdf), 2011. [Online]. Available: http://arxiv.org/pdf/1110.4241.pdf

[16] S. Boyd, L. Xiao, and A. Mutapcic, "Subgradient methods," Stanford University," Working Paper (available at http://www.stanford.edu/class/ee392o/subgradmethod.pdf), 2003. [Online]. Available: http://www.stanford.edu/class/ee392o/subgrad_method.pdf

[17] J. G. Carlsson, "Dividing a territory among several facilities," University of Minnesota, Twin Cities," Working paper (available at http://menet.umn.edu/~jgc), 2011. [Online]. Available: http://menet.umn.edu/~jgc/facility-partition-rev3.pdf

[18] M. Meghjani and G. Dudek, "Multi-robot exploration and rendezvous on graphs," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, oct. 2012, pp. 5270 –5276.