Hybrid SUSD-Based Task Allocation for Heterogeneous Multi-Robot Teams

Shengkang Chen[1], Tony X. Lin[1], Said Al-Abri[3], Ronald C. Arkin[2] and Fumin Zhang[1]

***Abstract*— Effective task allocation is an essential component to the coordination of heterogeneous robots. This paper proposes a hybrid task allocation algorithm that improves upon given initial solutions, for example from the popular decentralized market-based allocation algorithm, via a derivative-free optimization strategy called Speeding-Up and Slowing-Down (SUSD). Based on the initial solutions, SUSD performs a search to find an improved task assignment. Unique to our strategy is the ability to apply a gradient-like search to solve a classical integer-programming problem. The proposed strategy outperforms other state-of-the-art algorithms in terms of total task utility and can achieve near optimal solutions in simulation. Experimental results using the Robotarium are also provided.**

I. Introduction

Effective coordination of a heterogeneous multi-robot team (which consists of robots with diverse capabilities suitable for different tasks) is critical for successful mission completion. Central to ensuring effective coordination lies in solving a multi-robot task allocation (MRTA) problem. The goal of this problem is to determine an assignment between robots and tasks that optimizes some metric, such as maximizing total task utility. Effective solutions to the MRTA problem for heterogeneous teams potentially enable various applications ranging from exploration to search and rescue.

Because the multi-robot task allocation (MRTA) problem is a combinatorial optimization problem and the solution space is often non-continuous, solving the MRTA problem is generally very difficult. To rectify both difficulties, we propose a transformation of the problem into a parameter estimation problem for a probability distribution from which valid MRTA assignments are possible random draws. We then estimate the unknown parameters by leveraging a derivative-free optimization strategy known as Speeding-Up and Slowing-Down (SUSD) [1] that allows for following the gradient by using only function evaluations. Despite the lack of a well-defined gradient from random samples, SUSD is still able to improve by treating utility evaluations of random draws as noisy utility evaluations of the unknown parameters. However, since SUSD is a gradient-tracking strategy, it is susceptible to local maxima.

We therefore propose a hybrid task allocation approach to the MRTA problem. Our approach consists of two parts: a simple decentralized market-based task allocation algorithm using sequential auctioning from [2] which provides an initial guess near a good (sub)optimal solution, and an optimization of the allocation by using the Speeding-Up and Slowing-Down (SUSD) strategy. Figure 1 illustrates the overall framework of our hybrid task allocation approach.

First, the market-based algorithm calculates an acceptable initial solution. Then, the centralized SUSD task allocation algorithm improves upon the initial solution and sends the updated allocation back to the multi-robot team. We demonstrate the effectiveness of our proposed approach by comparing it with other state-of-the-art algorithms in simulations. By combining a simple market-based algorithm and SUSD, our proposed hybrid task allocation algorithm can provide a near optimal solution while maintaining a high degree of robustness.

Our contributions are as follows: **i)** the development of a hybrid task allocation framework that leverages a decentralized market-based approach and a centralized optimization-based approach, **ii)** the application of the SUSD strategy to the task allocation problem, which is a form of integer-programming which has not previously been explored for SUSD [3], [4], and **iii)** extensive simulation studies and experiments.

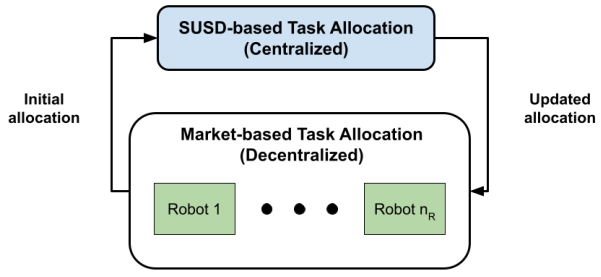


Fig. 1: Architecture diagram for the hybrid SUSD-based task allocation framework. When the multi-robot team discovers new tasks, the team uses a decentralized market-based task allocation algorithm to estimate an initial allocation. Then, the centralized SUSD-based task allocation module tries to improve the initial allocation and sends the updated allocation back to the multi-robot team.

The research work is supported by ONR grants N00014-19-1-2556 and N00014-19-1-2266; AFOSR grant FA9550-19-1-0283; NSF grants CNS-1828678, S&AS-1849228 CNS-2016582 and GCR-1934836; and NOAA grant NA16NOS0120028.

[1]Shengkang Chen, Tony X. Lin, and Fumin Zhang are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA schen754, tlin339, fumin@gatech.edu. [2] Ronald C. Arkin is with the School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332, USA arkin@gatech.edu [3] Said Al-Abri is with the Department of Electrical and Computer Engineering, Sultan Qaboos University, Alkhod, Oman ssabry@squ.edu.om

II. Background

Researchers have studied task allocation problems extensively and proposed various approaches in the past decade ([5], [6], [7] are some of the survey papers in this area). We can categorize task allocation approaches into four main categories: optimization-based, market-based, behavior-based and trait-based.

Optimization-based task allocation methods aim to find optimal solutions given a set of constraints. For example, the Hungarian methods (centralized [8] and distributed [9]) can find optimal solutions to task allocation problems in which each robot is assigned one task and there is an equal number of robots and tasks. Other strategies based on a mixed-integer linear program formulation solve the allocation problem [10], [11] by relaxing the typically integer constraints of the problem. However, these relaxations still yield problems that are strongly NP-hard [5], making tractable solutions to these problems impossible. As such, other results proposed in the literature leverage heuristic strategies (such as simulated annealing [12] and genetic algorithms [13]) to find sub-optimal solutions in reasonable time. One such family of heuristic strategies that are popularly used in the literature are market-based methods.

Market-based allocation methods are decentralized strategies inspired by capitalist economies. In these algorithms, robots bid on tasks based on task utilities and auction criteria. Due to their robustness against communication loss and their distributed nature [7], various market-based algorithms have been proposed, including MURDOCH [14], Prim Allocation [15], and the Consensus-Based Bundle Algorithm (CBBA) [16]. However, these strategies may require excessive inter-robot communication and can provide worse solutions than optimization-based approaches [17].

Instead of adapting task allocations, behavior-based methods leverage behavior changes in robots to handle tasks. In these methods [18], [19], [20], robots adapt their behaviors based on sensor inputs to achieve effective coordination. These methods are robust and responsive, but exhibit only locally optimal solutions and are difficult to design for more complex problems.

Trait-based task allocation methods [21], [22] aim to find the transition matrix that modifies the traits (capabilities) of heterogeneous robots to satisfy the requirements of tasks. While trait-based methods focus on the coalition formulation problem where each task requires multiple robots to complete, our proposed SUSD-based task allocation algorithm focuses on the task allocation problem where each robot is assigned with multiple tasks.

In addition to the four main categories of task allocation approaches, researchers have proposed hybrid approaches by combining different types of approaches [23]. In [24], Zitouni et al. proposed a hybrid algorithm that combines the ant colony algorithm to build task bundles and CBBA to resolve assignment conflicts. Another hybrid task allocation algorithm [25] combines pheromone maps and a market-based algorithm to provide UAVs with communication network support. Different from these algorithms, our proposed hybrid algorithm uses the SUSD strategy, a derivative-free optimization approach, to improve the solutions of a market-based task allocation algorithm.

The Speeding-Up and Slowing-Down strategy (SUSD) is an optimization method inspired by the schooling behaviors of fish [3], [4]. Agents that adopt the SUSD strategy will move towards the local minimum based on local function evaluations without explicit gradient calculations. The SUSD strategy has been used mainly in distributed source seeking [3], [4], [26], but has also been used for optimization applications (especially in cases where the gradient is ill-defined [1]). In this paper, we apply the SUSD strategy to task allocation to improve the solutions of market-based approaches for which no well-defined gradient exists.

III. Problem Formulation

Consider a team of robots $\mathscr{I} = \{1, 2, \ldots, n_R\}$ and a set of tasks $\mathscr{J} = \{1, 2, \ldots, n_T\}$ to complete in which each robot i is associated with a position $\mathbf{r}_i$ and each task j is associated with a position $\mathbf{v}_j$. We assume each robot may complete multiple tasks but tasks can only be assigned to single robots. We describe this assignment as a binary decision matrix

$$\mathbf{X} = \begin{bmatrix} —\mathbf{x}_1^\intercal— \\ —\mathbf{x}_2^\intercal— \\ \vdots \\ —\mathbf{x}_{n_R}^\intercal— \end{bmatrix} \tag{1}$$

where $\mathbf{x}_i \in \{0,1\}^{n_T}$ describes which tasks robot i is assigned and valid decisions obey the constraint $\sum_{j \in \mathscr{J}} \mathbf{x}_{ij} = 1$, i.e, only one element of each column in $\mathbf{X}$ is nonzero.

While our formulation allows for any robot to complete any task, we assume **a)** tasks have unique specifications that grant certain robots higher utility, and **b)** that farther travel distances degrade the reward associated with completing the task. Assume the n_T unassigned tasks have n_S different task types where each type $s \in \{1, 2, \ldots n_S\}$. Let $\mathbf{Y} \in \{0,1\}^{n_S \times n_T}$ be the matrix that represents the task types of the n_T unassigned tasks.

$$\mathbf{Y} = \begin{bmatrix} | & | & & | \\ \mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_{n_T} \\ | & | & & | \end{bmatrix}$$

where $\mathbf{y_j}$ is a one-hot vector representing the task type of task j. And we use matrix $\mathbf{Q} \in \mathbb{R}^{n_S \times n_R}$ where $[\mathbf{Q}]_{si} \in \mathbb{R}_{\geq 0}$ represents the quality of task execution of robot i on type s task. We encode these task specifications as a matrix $\mathbf{S} = \mathbf{Q}^\intercal \mathbf{Y} \in \mathbb{R}^{n_R \times n_T} \geq 0$ and the distance-degradation of the reward as a discount factor $\lambda < 1$. Leveraging $\mathbf{S}$ and λ, we introduce a utility function u that encodes the reward associated with a particular robot assignment $\mathbf{x}_i$. The individual utility is given by

$$u(\mathbf{x}_i) = \lambda^{\|\mathbf{r}_i - \mathbf{v}_{\gamma_1}\|_2} [\mathbf{S}]_{i,\gamma_1} + \sum_{j=2}^{M_i} \lambda^{\|\mathbf{v}_{\gamma_{j-1}} - \mathbf{v}_{\gamma_j}\|_2} [\mathbf{S}]_{i,\gamma_j} \tag{2}$$

in which $M_i = \sum_{j \in \mathscr{J}} \mathbf{x}_j$ is the number of tasks allocated to robot i and $\boldsymbol{\gamma} = \begin{bmatrix} \gamma_1, & \gamma_2, & \ldots, & \gamma_{M_i} \end{bmatrix}$ are the indices of all nonzero assignments in $\mathbf{x}_i$ arranged in order so as to minimize the total travel time. To find each $\boldsymbol{\gamma}_i$, each robot uses a greedy algorithm to sort these tasks.

The total assignment utility of the robot team is therefore given by

$$U(\mathbf{X}) = \sum_{i \in \mathscr{I}} u(\mathbf{x}_i) \tag{3}$$

for all valid assignments of $\mathbf{X}$. We describe now the problem of interest.

Problem 1 (Task Allocation): Assume robot positions $\mathbf{r}_i$ and task positions $\mathbf{v}_j$ are known and the task specification matrix $\mathbf{S}$ is available. The problem of interest is to solve the integer programming problem

$$\begin{aligned} \underset{\mathbf{X}}{\arg\max} \quad & U(\mathbf{X}) \\ \text{s.t.} \quad & [\mathbf{X}]_{ij} \in \{0,1\} \\ & \sum_{j \in \mathscr{J}} \mathbf{x}_{ij} = 1, \quad \forall i \in \{1,2,\ldots,n_R\} \end{aligned} \tag{4}$$

Note that Problem 1 is an integer-programming problem by virtue of the binary decision matrix $\mathbf{X}$ which makes this problem generally intractable to solve for large values of n_R or n_T. We instead propose a strategy to improve upon existing task allocation methods by applying a derivative-free optimization step after an initial guess is provided.

IV. Proposed Approach

We describe now a method to improve initial solutions to Problem 1, which we denote $\mathbf{X}^M$. For this work, we assume these solutions are generated using a decentralized market-based approach (please refer to the Appendix for more details) but any method to solving Problem 1 is applicable. Our approach to improving initial solutions is focused on applying a gradient descent search using $\mathbf{X}^M$ as an initial condition. The principal difficulties with applying a gradient search to integer programming problems are associated with the lack of both a continuously defined solution space and a well-defined gradient $\nabla_{\mathbf{X}} U$.

We first propose a relaxation of the integer constraints by treating candidate solutions $\mathbf{X}$ as samples from a probability distribution $p(\mathbf{X} \mid \Theta)$ parameterized by a continuous valued matrix Θ. As such, Problem 1 can be treated as a continuous search for the Θ^* such that samples $\mathbf{X}^{SUSD} \sim p(\mathbf{X} \mid \Theta^*)$ always yield $U(\mathbf{X}^{SUSD}) \geq U(\mathbf{X}^M)$. However, the continuous search is non-trivial due to the lack of a well-defined gradient $\nabla_\Theta U$ since the gradient of a sample with respect to the distribution parameters is ill-defined.

We therefore combine the aforementioned relaxation with a derivative-free search algorithm called Speeding-Up and Slowing-Down (SUSD) which only requires function evaluations to follow the positive gradient. Our approach has been shown to follow the positive gradient even in conditions when the gradient is not available or explodes [1]. In this work, we treat the sampled solutions as noisy evaluations of the utility function (3) and apply the SUSD search strategy to refine Θ. A high-level description of the approach is shown in Algorithm 1.

Algorithm 1 Hybrid SUSD-based Task Allocation

procedure SUSD-TA($\mathscr{I}$, $\mathscr{J}$)
  $\mathbf{X}^M$ = MARKET-BASED($\mathscr{I}$, $\mathscr{J}$, $\mathbf{S}$)
  $\{\Theta_i^{(0)}\}_{i=1}^{n_V}$ = INIT(α, β, $\mathbf{X}^M$) ▷ As in (9)
  $\mathbf{X}^{SUSD}, \Theta^{SUSD}$ = SUSD($\{\Theta_i^{(0)}\}_{i=1}^{n_V}$) ▷ As in (7)
  return $\mathbf{X}^{SUSD}, \Theta^{SUSD}$
end procedure

A. Continuous Relaxation of Integer Programming

To relax the aforementioned integer program, we use a probability distribution $p(\mathbf{X} \mid \Theta)$ to transform the binary decision problem into a continuous parameter estimation problem. The distribution is chosen as the normalized exponential function (also known as the softmax). The probability of robot i being assigned task j given parameter matrix Θ is

$$p_{ij} = \mathbb{P}([\mathbf{X}]_{ij} = 1 \mid \Theta) = \frac{e^{\Theta_{ij}}}{\sum_{m=1}^{n_R} e^{\Theta_{mj}}}. \tag{5}$$

To sample valid assignments $\mathbf{X}$ from a particular Θ, we create a weight vector $\mathbf{w}_j = [p_{1j}, p_{2j}, \ldots, p_{n_R j}]$ for each task and use roulette-wheel selection [27] to select the assigned robot. To help promote randomness in the search, we also occasionally set each column of Θ to a uniform probability distribution with probability ε,

$$p_{ij} = \mathbb{P}([\mathbf{X}]_{ij} = 1 \mid \Theta) = \frac{1}{n_R}. \tag{6}$$

Incorporating the uniform sampling (6) has helped us to find better solutions when optimizing with SUSD. A detailed description of the random sampling for $p(\mathbf{X} \mid \Theta)$ is shown in Algorithm 2.

Algorithm 2 Random Sampling for Task Allocation

procedure RANDOM_SAMPLING(Θ, ε)
  $n \sim U[0,1]$ ▷ Uniformly sample between 0 and 1
  if $n < \varepsilon$ **then**
    $\Theta \leftarrow$Uniform(n_R) ▷ As in (6)
  end if
  $\mathbf{X} = \mathbf{0}$ ▷ Initialize empty assignment
  for $j \in \mathscr{J}$ **do**
    $\mathbf{w}_j \leftarrow [p_{1j}, p_{2j}, \ldots, p_{n_R j}]$
    $i^* \leftarrow$ Roulette($\mathbf{w}_j$)
    $[\mathbf{X}]_{i^* j} = 1$ ▷ Assign robot i^* to task j
  end for
  return X
end procedure

B. Application of SUSD

SUSD is a derivative-free optimization strategy that was originally used for distributed source-seeking in robot teams [3], [4]. The strategy relies on maintaining a set of candidate

solutions and applying repeated function evaluations to estimate and follow the negative gradient. To ensure improvement of the candidate solutions, we invert (3) by assuming a max utility U_{max} is known and apply the difference between the max and the evaluation as the function evaluation. In this work, we maintain $n_V \geq n_R \times n_T$ candidate solutions and apply the SUSD search strategy to improve candidates. Letting $\mathbf{X}^k(t)$ be the k^{th} sampled assignment at iteration t as described previously, the SUSD strategy is given by

$$\text{vec}(\Theta^k(t+1)) = \text{vec}(\Theta^k(t)) + \eta z^k(t)\mathbf{n}(t) + \mathbf{u}_f^k(t) \quad (7)$$

in which $\mathbf{u}_f^{(k)}(t)$ is a formation control input as in [1] to help keep candidate solutions close together, $z^k(t) = 1 - \exp(U(\mathbf{X}^k(t)) - U_{max}(t))$ is the exponential mapping of the sampled assignment in which $U_{max}(t)$ is the maximum evaluated $U(\mathbf{X}^k(t))$ of all candidate solutions also as in [1]. $\text{vec}(\cdot)$ converts a matrix to a vector by vertically stacking each column, $\eta > 0$ is a tuning gain to control the search speed and stability, and $\mathbf{n}(t)$ is the eigenvector associated with the smallest eigenvalue of the covariance matrix

$$\mathbf{C}(t) = \sum_{k=1}^{n_V} (\text{vec}(\Theta^k(t) - \bar{\Theta}(t)))(\text{vec}(\Theta^k(t) - \bar{\Theta}(t)))^\intercal \quad (8)$$

in which $\bar{\Theta}(t) = \frac{1}{n_V}\sum_{k=1}^{n_V}\Theta^k(t)$ is the average parameter. Since strategy (7) depends on finding the eigenvector of the covariance across all candidate solutions, the condition $n_V \geq n_R \times n_T$ ensures a unique eigenvector associated with the minimum eigenvalue exists so long as the candidate solutions are linearly independent which can be achieved by incorporating small random noise.

In our previous work, we have shown that applying (7) allows the shared search direction $\mathbf{n}(t)$ to converge within a small neighborhood of $\nabla_\Theta U$ when direct evaluations $U(\Theta^k(t))$ are available [1]. While we are unable to find direct evaluations in this work, we treat evaluations $U(\mathbf{X}^k(t))$ as noisy evaluations instead and show, in simulation, that applying the proposed strategy yields good results.

To initialize the candidate solutions, we use the initial assignments $\mathbf{X}^M$ and transform these assignments to initial parameter values using

$$\Theta^k(0) = \alpha\mathbf{X}^M + \beta\mathbf{1} + \varepsilon^k \quad (9)$$

in which $\alpha \gg \beta > 0$ are user-defined hyperparameters, $\varepsilon^k \sim U(0,1)^{n_R \times n_T}$, and $\mathbf{1}$ is a column vector of ones of dimension $n_R \times n_T$. Applying (9) allows us to generate linearly independent candidates and bias these candidates towards the provided initial solution. The best solution is then chosen out of all sampled solutions over each iteration of the SUSD search. The search strategy is also detailed in Algorithm 3.

Algorithm 3 The SUSD-based Task Allocation

procedure SUSD($\mathscr{I}, \mathscr{J}, \{\Theta^k(0)\}_{k=1}^{n_V}$)
  $\Theta^*, \mathbf{X}^*, u^* \leftarrow \emptyset, \emptyset, 0$ ▷ Initialize no solution
  for t = 1 to T **do**
    $\mathbf{C}(t) \leftarrow \text{Cov}(\{\Theta^k(t)\})$ ▷ As in (8)
    $\mathbf{n}(t) \leftarrow$PCA($\mathbf{C}(t)$) ▷ Get search direction
    for each $k \in \mathscr{I}$ **do**
      $\mathbf{X}^k(t) \leftarrow \text{Sample}(\Theta^k(t))$ ▷ Draw one solution
      $u^k \leftarrow U(\mathbf{X}^k(t))$ ▷ Get utility of sample
      $\mathbf{u}_f^k(t) \leftarrow$form$(k, \{\Theta^k(t)\})$ ▷ Formation Input
      $\Theta^k(t+1) \leftarrow$SUSD$(u^k, \mathbf{n}(t), \mathbf{u}_f^k(t))$ ▷ As in (7)
      if $u^k > u^*$ **then** ▷ Update best solution
        $\Theta^*, \mathbf{X}^*, u^* \leftarrow \Theta^k(t), \mathbf{X}^k(t), u^k$
      end if
    end for
  end for
  return $\Theta^*, \mathbf{X}^*$
end procedure

V. Evaluation

A. Simulations

In this section, we evaluate our proposed hybrid SUSD-based task allocation (SUSD-TA) algorithm in a 2D simulation environment with two different types of sensing tasks. The heterogeneous multi-robot team consists of robot equipped with different sensors that can perform some types of tasks better than other robots. The multi-robot team needs to find an assignment between tasks and robots to maximize the total task utility. Given the assignment, each robot needs to navigate to its assigned tasks' location in order to execute the tasks. To calculate the task utilities, we set the distance discount factor $\lambda = 0.6$.

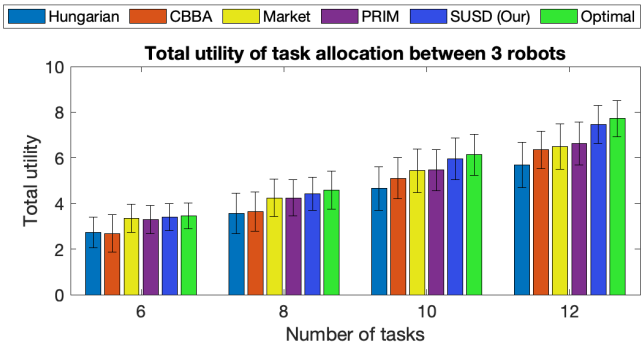
We ran the simulation 30 times for each scenario. In each simulation run, robot locations are randomly selected, so are the task types and task locations. All algorithms are tested in the same setups (same random seeds) for a fair comparison. We have implemented several popular task allocation approaches for comparative studies:

1) Hungarian: the Hungarian method [8] with dummy robots and tasks when the number of tasks is different from the number of robots.
2) CBBA: the Consensus-Based Bundle Algorithm with a modified utility function [16] to avoid circular bidding.
3) PRIM: Prim Allocation, a market-based algorithm based on Prim's algorithm [15].
4) Market: a simple sequential auction algorithm where robots bid on one task at a time [2].
5) Optimal: a brute-force search over all possible cases to find the best solution.

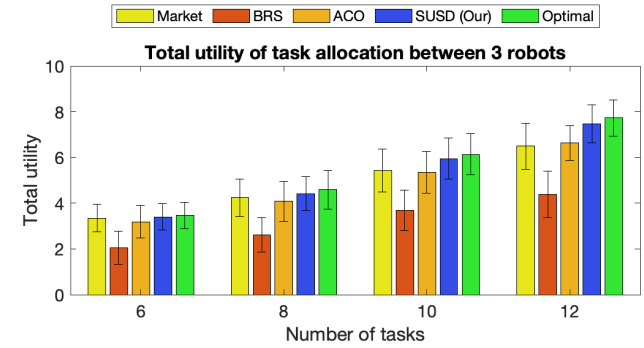
Moreover, to further test our hybrid algorithm, we have implemented two alternative algorithms by replacing SUSD in our algorithm with the Basic Random Search (BRS) algorithm from [28] or the ant colony optimization (ACO) algorithm from [29]:

1) BRS: an alternative hybrid algorithm applying the Basic Random Search (BRS) algorithm to improve the solutions of the market-based algorithm.
2) ACO: an alternative hybrid algorithm applying the ant colony optimization (ACO) algorithm to improve the solutions of the market-based algorithm.

Scenario 1-three heterogeneous robots: In this scenario,

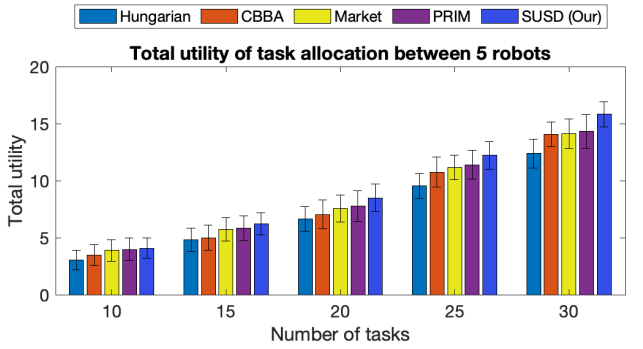


(a) Comparisons with popular task allocation algorithms.

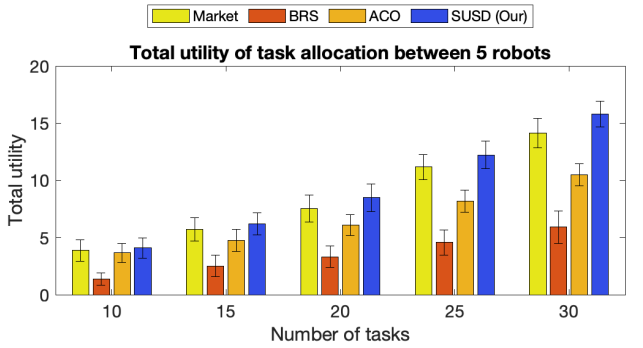


(b) Comparisons with other hybrid algorithms (combing a market-based algorithm with an optimization algorithm).

Fig. 2: Comparisons with different task allocation algorithms and optimal solutions in terms of total task utility (a team of 3 robots).



(a) Comparisons with state-of-the-art task allocation algorithms.



(b) Comparisons with other hybrid algorithms (combing a market-based algorithm with an optimization algorithm).

Fig. 3: Comparisons with different task allocation algorithms in terms of total task utility (a team of 5 robots).

we consider a heterogeneous multi-robot team composed of three robots to operate in a 10-meter by 10-meter environment. There are two different types of robots in the team, and we use a matrix $\mathbf{Q}$ to represent the different sensing capabilities of the robots:

$$\mathbf{Q} = \begin{bmatrix} 2 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

The first column of $\mathbf{Q}$ describes the robot 1's quality of task execution for each type of task, where the quality of task execution on the first type of task is 2 and only 1 for the second type of task. In other words, the robots will receive a higher reward when executing the first type of task than the second type.

In each case, the robot team needs to complete 6 to 12 tasks. Furthermore, we set $\alpha = 5$ and $\beta = 0.1$.

In Fig. 2, we compared our proposed hybrid SUSD-based task allocation algorithm with both existing state-of-the-art algorithms (Fig. 2a) and two alternative hybrid algorithms that combine a market-based algorithm and an optimization-based algorithm in the same way as our hybrid algorithm (Fig. 2b). As shown in Fig. 2a, the market-based algorithm achieves a performance comparable to other state-of-the-art algorithms and provides close to optimal solutions when the task number is small. However, as the number of tasks increases, the performance gap between the market-based and optimal solutions increases. In these cases, our proposed hybrid SUSD-based algorithm can significantly improve the solution of the market-based algorithm (16% improvement in the 12-task case) and provide close to optimal solutions (3.1% gap).

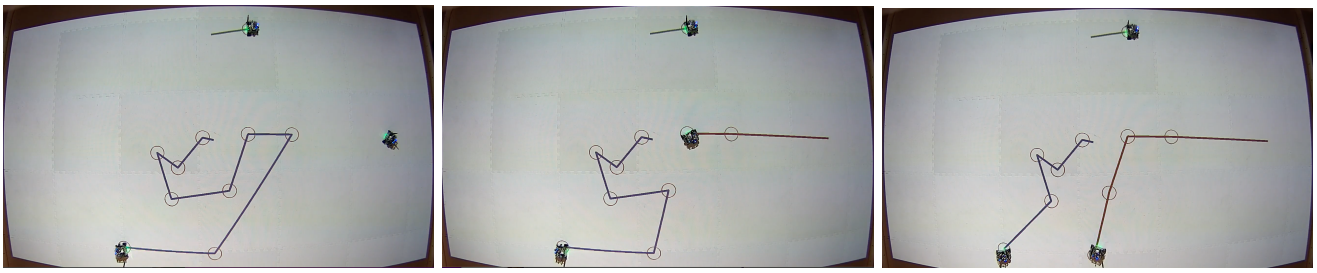
To further validate the efficiency of SUSD improving market-based task allocation algorithm, we compare it with the Basic Random Search (BRS) algorithm and the ant colony optimization (ACO) algorithm to see if these algorithms can also improve the solutions of the market-based algorithm. As shown in Fig. 2b, both BRS and ACO fail to improve the market-based algorithm. Although ACO provides similar performance to the market-based approach, BRS provides significantly worse solutions.

Scenario 2-five heterogeneous robots: In this scenario, we considered a larger multi-robot team composed of five robots with three different types to operate in a 20-meter by 20-meter environment to execute more tasks ranging from 10 to 30. Similarly, we use a matrix $\mathbf{Q}$ to present the different sensing capabilities of robots:

$$\mathbf{Q} = \begin{bmatrix} 2 & 2 & 1 & 1 & 2 \\ 1 & 1 & 2 & 2 & 2 \end{bmatrix}$$

Due to the time complexity of using brute force search to find optimal solutions, we do not compare our algorithm to optimal solutions in this scenario. For the larger robot team, we set $\alpha = 8$ and $\beta = 0.1$.

Compared with the first scenario, the second scenario (Fig. 3) makes task allocation more challenging by increasing



(a) Market-based Allocation ($U = 7.99$)

(b) Hybrid SUSD-based Allocation ($U = 8.09$)

(c) Optimal Allocation ($U = 8.14$)

Fig. 4: Task allocation with a team of three homogeneous on the Robotarium. In experiments, robots started at fixed locations and the task locations are randomly generated. The red circles represent go-to-goal tasks' locations, and the colored lines represent robots' paths to execute their assigned tasks. While the market-based allocation is acceptable, SUSD strategy in our hybrid algorithm improves the market-based allocation to achieve higher total task utilities U. Furthermore, the total task utility of our hybrid algorithm is close to the utility of the optimal allocation.

the number of robots and tasks. A similar distribution between popular task allocation algorithms and our proposed algorithm compared with the first scenario is shown in Fig. 3a. Moreover, in Fig. 3b, BRS and ACO in hybrid algorithms provide significantly worse solutions than the market-based approach in this more complex scenario, but our proposed approach using SUSD can still improve the solutions (10.4% improvement). These results indicate that the hybrid SUSD-based task allocation algorithm is a valid approach for improving task assignments from the market-based algorithm.

B. Experiments

We have also validated our hybrid SUSD-based task allocation algorithm on the Robotarium [30], a remotely accessible testbed with physical robots. During experiments, a team of three homogeneous robots needs to find the task assignment of 10 go-to-goal tasks using the proposed algorithm. The robots need to navigate to their assigned tasks respectively. Snapshots of the experiments are shown in Fig. 4 where the red circles represent the task locations and the colored lines are the robots' paths to execute their assigned tasks. The market-based algorithm provides an acceptable solution (Fig. 4a), and our hybrid algorithm improved the solution to achieve a higher total task utility U (Fig. 4b). Moreover, the total task utility achieved by our hybrid algorithm is close to the total task utility of the optimal allocation.

VI. Discussion and Conclusion

In this paper, we present a hybrid task allocation algorithm that combines a simple market-based algorithm and the Speeding-Up and Slowing-Down (SUSD) strategy. Using the solution of the market-based algorithm, SUSD strategy can significantly improve it to achieve better task allocation. SUSD strategy can leverage the solutions of the market-based algorithm as starting locations for its virtual agents to perform a directed search along the negative gradient of the valuation function in the task allocation state space to find a better solution without explicit gradient calculations. The advantage of this algorithm includes the robustness of using a decentralized algorithm and the performance of using a centralized optimization algorithm. Moreover, this hybrid algorithm is flexible and can utilize SUSD to improve the solutions of other decentralized algorithms.

We demonstrate the performance of our proposed algorithm through simulations and compare it with state-of-the-art algorithms and alternative hybrid algorithms that use different optimization approaches. Our algorithm can provide better solutions than these algorithms. One of the limitations of this approach is the computational cost of applying Principal Component Analysis (PCA) in SUSD. In the future, we plan to develop a distributed version of the hybrid task allocation algorithm for better computational efficiency and robustness.

Appendix

For the market-based approach, we use a simple sequential auction algorithm from [2] where robots bid on one task at a time. A simplified version is shown in Algorithm 4. For each task j announced by one of the robot, each robot i submits its bid b_{ij} where the bid value b_{ij} is the additional utility gain for robot i to execute task j given its current task assignments $\mathbf{x}_i$, where $\mathbf{e}_j$ is a basis vector with jth entry equals to 1. The announcer will pick the robot with the highest bid as the winner. The winner robot i^* will add task j to its task assignments $\mathbf{x}_{i^*}$.

Algorithm 4 The Market-based Task Allocation (Sequential Auction)

procedure MARKET-BASED_MRTA(n_R, n_T)
  for task $j \in \{1, 2, \ldots, n_T\}$ **do**
    for robot $i \in \{1, 2, \ldots, n_R\}$ **do**
      $b_{ij} = \mathrm{U}(\mathbf{x}_i + \mathbf{e}_j) - \mathrm{U}(\mathbf{x}_i)$
    end for
    $i^* = \arg\max_{i \in \mathscr{I}}(b_i)$
    $\mathbf{x}_{i^*} \leftarrow \mathbf{x}_{i^*} + \mathbf{e}_j$
  end for
  return X
end procedure

REFERENCES

[1] S. Al-Abri, T. X. Lin, M. Tao, and F. Zhang, "A derivative-free optimization method with application to functions with exploding and vanishing gradients," *IEEE Control Systems Letters*, vol. 5, pp. 587–592, 4 2021.

[2] M. Otte, M. Kuhlman, and D. Sofge, "Multi-robot task allocation with auctions in harsh communication environments," vol. 2018-January. Institute of Electrical and Electronics Engineers Inc., 7 2017, pp. 32–39.

[3] W. Wu and F. Zhang, "A speeding-up and slowing-down strategy for distributed source seeking with robustness analysis," *IEEE Transactions on Control of Network Systems*, vol. 3, pp. 231–240, 9 2016.

[4] S. Al-Abri, S. Maxon, and F. Zhang, "Integrating a pca learning algorithm with the susd strategy for a collective source seeking behavior," *Proceedings of the American Control Conference*, vol. 2018-June, pp. 2479–2484, 8 2018.

[5] B. P. Gerkey and M. J. Mataric, "A formal analysis and taxonomy of task allocation in multi-robot systems," *In Intl. J. of Robotics Research*, vol. 23, pp. 939–954, 2004.

[6] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *The International Journal of Robotics Research*, vol. 32, pp. 1495–1512, 10 2013.

[7] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot task allocation: A review of the state-of-the-art," *Studies in Computational Intelligence*, vol. 604, pp. 31–51, 2015.

[8] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, 3 1955.

[9] S. Chopra, G. Notarstefano, M. Rice, and M. Egerstedt, "A distributed version of the hungarian method for multi-robot assignment," *IEEE Transactions on Robotics*, vol. 33, pp. 932–947, 5 2018.

[10] N. Atay and B. Bayazit, "Mixed-integer linear programming solution to multi-robot task allocation problem," pp. 2006–54, 1 2006.

[11] A. Falsone, K. Margellos, and M. Prandini, "A decentralized approach to multi-agent milps: Finite-time feasibility and performance guarantees," *Automatica*, vol. 103, pp. 141–150, 5 2019.

[12] A. R. Mosteo and L. Montano, "Simulated annealing for multi-robot hierarchical task allocation with flexible constraints and objective functions." Institute of Electrical and Electronics Engineers (IEEE), 10 2006.

[13] R. Patel, E. Rudnick-Cohen, S. Azarm, M. Otte, H. Xu, and J. W. Herrmann, "Decentralized task allocation in multi-agent systems using a decentralized genetic algorithm," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3770–3776, 5 2020.

[14] B. P. Gerkey and M. J. Matarić, "Sold!: Auction methods for multi-robot coordination," *IEEE Transactions on Robotics and Automation*, vol. 18, pp. 758–768, 10 2002.

[15] M. G. Lagoudakis, M. Berhault, S. Koenig, P. Keskinocak, and A. J. Kleywegt, "Simple auctions with performance guarantees for multi-robot task allocation," *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 1, pp. 698–705, 2004.

[16] H. L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Transactions on Robotics*, vol. 25, pp. 912–926, 2009.

[17] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz, "Market-based multirobot coordination: A survey and analysis," *Proceedings of the IEEE*, vol. 94, 2006.

[18] L. E. Parker, "Alliance: An architecture for fault tolerant multirobot cooperation," *IEEE Transactions on Robotics and Automation*, vol. 14, pp. 220–240, 1998.

[19] A. Kanakia, B. Touri, and N. Correll, "Modeling multi-robot task allocation with limited information as global game," *Swarm Intelligence*, vol. 10, pp. 147–160, 6 2016.

[20] S. Park, Y. D. Zhong, and N. E. Leonard, "Multi-robot task allocation games in dynamically changing environments." Institute of Electrical and Electronics Engineers (IEEE), 10 2021, pp. 8678–8684.

[21] A. Prorok, M. A. Hsieh, and V. Kumar, "The impact of diversity on optimal control policies for heterogeneous robot swarms," *IEEE Transactions on Robotics*, vol. 33, pp. 346–358, 2017.

[22] J. Track, H. Ravichandar, K. Shaw, and S. Chernova, "Strata: Unified framework for task assignments in large teams of heterogeneous agents: Jaamas track," 2021.

[23] G. M. Skaltsis, H. S. Shin, and A. Tsourdos, "A survey of task allocation techniques in mas," *2021 International Conference on Unmanned Aircraft Systems, ICUAS 2021*, pp. 488–497, 6 2021.

[24] F. Zitouni, S. Harous, and R. Maamri, "A distributed approach to the multi-robot task allocation problem using the consensus-based bundle algorithm and ant colony system," *IEEE Access*, vol. 8, pp. 27 479–27 494, 2020.

[25] R. S. de Moraes and E. P. de Freitas, "Distributed control for groups of unmanned aerial vehicles performing surveillance missions and providing relay communication network services," *Journal of Intelligent and Robotic Systems*, vol. 92, pp. 645–656, 11 2017.

[26] S. Al-Abri and F. Zhang, "A distributed active perception strategy for source seeking and level curve tracking," *IEEE Transactions on Automatic Control*, vol. 67, no. 5, pp. 2459–2465, 2021.

[27] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, 13th ed. Addison-Wesley Professional, 1989.

[28] H. Mania, A. Guy, and B. Recht, "Simple random search provides a competitive approach to reinforcement learning," 3 2018.

[29] J.-P. Wang, Y. Gu, and X.-M. Li, "Multi-robot task allocation based on ant colony algorithm," *Journal of Computers*, vol. 7, 9 2012.

[30] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt, "The robotarium: A remotely accessible swarm robotics research testbed." IEEE, 5 2017, pp. 1699–1706.