2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Task Allocation under Communication Constraints using Motivated Particle Swarm Optimization

Medria K. D. Hardhienata, V. Ugrinovskii, and Kathryn E. Merrick

Abstract— **This paper considers task allocation problems where a group of agents must discover and allocate themselves to tasks. Task allocation is particularly difficult when agents can only exchange information over a limited communication range and when the agents are initialized from a single departure point. To address these constraints, we present a novel approach that incorporates computational models of motivation into a guaranteed convergence particle swarm optimization algorithm. We introduce an incentive function and three motive profiles to guaranteed convergence particle swarm optimization. Our new algorithm is compared to existing approaches with and without motivation under conditions of limited communication. It is tested in the case where the agents are initialized from a single point and random points. Results show that our approach increases the number of tasks discovered by a group of agents under these conditions. Furthermore, it significantly outperforms benchmark PSO algorithms in the number of tasks discovered and allocated when the agents are initialized from a single point.**

I. Introduction

In this paper, we consider a task allocation problem where a group of agents is used to discover and allocate themselves to tasks in a confined area [1]–[4]. Problems of task allocation occur in applications such as detecting and clearing mines [2] and finding and rescuing trapped victims inside a destroyed building [3]. A number of variations of the task allocation problem exist [5], [6]. This paper considers task allocation problems under specific conditions, i.e. where agents have a limited communication range and where agents are initialized from a single point.

A critical aspect of physical multi-agent systems is that agents will not be able to exchange information with other agents that are outside their communication range. Another aspect to consider is that agents are often required to start searching from a single departure point [7]. An example of such problems is the case where the agents enter a building to search for targets from a single entrance [7]. When the geometry inside the building is unknown, this complicates the process of distributing the agents around the search space, hence the ability to initialize the search from a single point is essential. These aspects must be taken into consideration when designing a multi-agent system for task allocation.

Taking the above conditions into account, the focus of this paper is on problems where the tasks are able to broadcast signals and the agents are able to sense the strength of those signals. It is assumed that the strength of the signal sensed by the agents reduces with distance from the tasks and reaches a maximum value when the agent is at the task location. This allows the agents to behave as if they take measurement of an unknown fitness function. The task allocation problem is hence cast in this paper as an optimization problem [1] with the goal to maximize an unknown multimodal fitness function. To solve this problem, a population-based stochastic optimization technique called Particle Swarm Optimization (PSO) is used in this paper as a basic foundation.

The PSO algorithm was first introduced in [8] where a swarm of particles was used to iteratively find the best solution to an optimization problem. The algorithm has been applied to a wide range of problems including the domains of multi-agent search [4], [9] and task allocation [1]. Within the context of task allocation, agents are associated with particles in the optimization problem, and the tasks are associated with the maxima of a fitness function. In this particular problem, we are concerned with tasks of equal level of priority which corresponds to a fitness function which has equal level of maxima. It is assumed that the agents are able to quantify the fitness function at their location, but they do not know its shape or the locations and the number of its maxima/minima, etc.

The original PSO algorithm assumes that the agents' communication range is regarded to be unlimited [10]. To mimic agents in the multi-agent system with limited communication capability, the PSO agents have been further designed to only communicate with other agents within a predefined communication range [4], [9]. The limitation of the work in [4], [9] is that the approaches proposed in those papers only solve task allocation problems with a single task in the search space.

Other studies have been partially explored in [11], [12] regarding the application of the PSO algorithm for solving task allocation problems where there are multiple tasks. It is, however, assumed in [11], [12], that the number of tasks available in the search space was initially known by the agents. Moreover, most PSO-based algorithms assume that the positions of the agents are randomly initialized [4], [9], [11], [12]. Using such an assumption, the agents are required to move towards their initial starting points [13]. However, in the situation where the agents are initialized from a single point and the geometry of the search space is unknown, the algorithms in [4], [9], [11], [12] can not be used to solve such problems.

To address the problems mentioned above, we present a novel approach to the task allocation problem for agents un-

Medria K. D. Hardhienata, V. Ugrinovskii, and Kathryn E. Merrick are with the School of Engineering and Information Technology, UNSW Canberra, Northcott Drive, Canberra, ACT-2600, Australia. e-mail: medria.hardhienata@student.adfa.edu.au, (v.ougrinovski, k.merrick)@adfa.edu.au.

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

der conditions of limited communication. In order to endow the basic PSO algorithm with the ability to discover and allocate agents to multiple tasks, we embed the agents with models of motivation. Three models of motivation, namely, affiliation, achievement, and power models, were introduced in [14] which permit artificial agents to exhibit different behaviors in goal-selection with respect to incentives. Agents with affiliation, achievement, and power motive profiles are characterized with a preference for low, intermediate, and high incentive goals respectively [14]. Based on this idea, we incorporate models of motivation into PSO to allow the agents to exhibit different characteristics in task selection. In contrast to [11], [12], the use of motivation together with the PSO algorithm allows our algorithm to deal with the condition where the number of tasks is not known a priori.

To further prevent the agents from stagnation and prematurely converge on suboptimal solutions, we use a specific variant of the PSO algorithm, namely the guaranteed particle swarm optimization (GCPSO) algorithm [15]. In this paper, we combine the GCPSO algorithm with models of motivation. The new algorithm is further referred to as Motivated-Guaranteed Convergence Particle Swarm Optimization (M-GCPSO).

The idea to combine PSO and models of motivation was first proposed in our previous work on Motivated Particle Swarm Optimization (MPSO) without communication constraints [13]. One of the contributions of this paper is to extend our study in [13] to specifically address situations where the agents only communicate with neighbors that are within a predefined communication range. Compared to its previous version in [13], the algorithm in this paper employs a new incentive function that is more sensitive to changes in the number of agents around a potential best-found position of a task. The construction of this function takes roots in the domain of risk-sensitive decision making [16]. The modification of the incentive function using the principles of risk-sensitivity is the second contribution of this paper. In addition, we consider a wider range of motive profiles to create agents with different preferences for certain kinds of incentives.

In comparison with the standard GCPSO algorithm that does not employ motivated agents, simulation results indicate that the M-GCPSO algorithm increases task discovery. Moreover, a significant increase in the number of tasks discovered and allocated agents is observed when the agents are initialized from a single point. The results also show that the proposed algorithm significantly improves the performance of the original MPSO algorithm under limited communication constraints.

This paper is divided into four sections. We explain our approach for solving task allocation problems and the procedure of the M-GCPSO algorithm in Section II. The simulation setup and the results of the simulations are presented in Section III. The paper concludes with a summary of the proposed approach and a discussion of future work in Section IV.

II. Motivated Guaranteed Convergence Particle Swarm Optimization under Conditions of Limited Communication

This section presents our approach for solving task allocation problems and describes the proposed Motivated Guaranteed Convergence Particle Swarm Optimization (M-GCPSO) algorithm.

A. The M-GCPSO algorithm

Let S denotes an ordered set of M agents $S = \left\{s^1, s^2, ..., s^m, ..., s^M\right\}$ and x_t^m denotes the position of agent s^m at iteration t respectively. At each iteration, the set of neighbors of agent s^m in the present M-GCPSO algorithm is defined as

$$\mathcal{N}_t^m = \left\{ s^b \colon \|x_t^m - x_t^b\| < \delta \right\}, \tag{1}$$

where δ is the maximum communication range of the agents, $\delta > 0$ [17].

Each agent s^m is assumed to be able to remember the location of the highest signal strength it has sensed so far (personal best position), y_t^m. The value of y_t^m in the M-GCPSO algorithm is updated based on the standard PSO procedure at each iteration as follows [8]:

$$y_{t+1}^m = \begin{cases} y_t^m & \text{if } f(x_{t+1}^m) \leq f(y_t^m), \\ x_{t+1}^m & \text{if } f(x_{t+1}^m) > f(y_t^m), \end{cases} \tag{2}$$

where f is a fitness function that is defined by the strength of the signal field created by all tasks. Note that to perform this update, only values of f are used and precise knowledge of f is not required. Now we introduce the set $\mathcal{S}^m$ consisting of personal best positions of the agents in $\mathcal{N}^m$, $\mathcal{S}_t^m = \left\{ y_t^b \colon s^b \in \mathcal{N}_t^m \right\}$.

To calculate the position of the highest signal strength, g_t^m, found by the neighborhood, each agent s^m in the M-GCPSO algorithm will first consider all potential neighborhood best positions. Let the highest sensed value of a signal in the neighborhood of agent s^m be attained at $\hat{y}_t^m = \arg\max_{y_t^z \in \mathcal{S}^m}(f(y_t^z))$, the set of potential neighborhood best positions, $\mathbf{G}_t^m$, is calculated as

$$\mathbf{G}_t^m = \{ y_t^z \mid y_t^z \in \mathcal{S}_t^m \wedge \ |f(\hat{y}_t^m) - f(y_t^z)| \leq \mu \}. \tag{3}$$

In the above equation, the value of μ is set to 0.1 as it has been observed in [13] that this value leads to good performance. Next, the set $\mathbf{G}_t^m$ is augmented with an artificial randomly generated position in the search space, y^*, which results in a new set $\mathbf{G}_t^{m*}$. That is,

$$\mathbf{G}_t^{m*} = \mathbf{G}_t^m \cup \{ y_t^* \}. \tag{4}$$

The above equation is used to prevent the agents from concentrating within a small region in the search space [13].

Using Equation (4), the neighborhood best position g_t^m in the M-GCPSO algorithm is calculated as follows:

$$g_t^m \in \arg\max_{y_t^l \in \mathbf{G}_t^{m*}} T_{res}(I_t^m(y_t^l)), \tag{5}$$

where $I_t^m(\cdot)$ is the incentive function and $T_{res}(\cdot)$ is the motivation function of agent s^m. The construction of $I_t^m(\cdot)$ and $T_{res}(\cdot)$ will be explained in the following two subsections. In the case where Equation (5) results in more than one point, the point nearest to the agent is selected to be g_t^m.

Once each agent s^m has computed its neighborhood best position g_t^m, it updates its velocity and position as follows:

$$\hat{v}_{t+1}^m = \begin{cases} \chi(v_t^m + \varphi_1 r_1(y_t^m - x_t^m) & \\ \quad +\varphi_2 r_2(g_t^m - x_t^m)), \text{if } g_t^m \neq y_t^m \wedge g_t^m \neq y_t^* & \text{(6a)} \\ \chi v_t^m - x_t^m + g_t^m & \\ \quad +\rho_t(1-2r_3), \quad \text{if } g_t^m = y_t^m & \text{(6b)} \\ \chi v_t^m + \lambda(1-2r_4), \quad \text{if } g_t^m = y_t^* & \text{(6c)} \end{cases}$$

$$v_{t+1}^m = \begin{cases} \hat{v}_{t+1}^m & \text{if } -v_{max} \leq \hat{v}_{t+1}^m \leq v_{max} \\ v_{max} & \text{if } \hat{v}_{t+1}^m > v_{max} \\ -v_{max} & \text{if } \hat{v}_{t+1}^m < -v_{max}\ , \end{cases} \quad (7)$$

$$x_{t+1}^m \quad = \quad x_t^m + v_{t+1}^m\ . \quad (8)$$

where v_t^m denotes the velocity of agent s^m at time step t; v_{max} is the maximum allowed velocity; λ is a constant and ρ_t is a scaling factor that are both used to scale the contribution of the random search. Here, we set $\lambda = 2$ as it has been observed in this case to provide good performance; χ is a constriction coefficient which is calculated based on $\chi = \frac{2}{|2-\varphi-\sqrt{\varphi(\varphi-4)}|}$ with $\varphi = \varphi_1+\varphi_2 = 4.1$ [18]. In this paper, we set $\varphi_1 = \varphi_2 = 2.05$ and $\chi = 0.729844$ as suggested in [18]; r_1, r_2, r_3 and r_4 are random values in the range $[0, 1]$, sampled from a uniform distribution.

Note that three cases are considered in Equation (6) to update the velocity of the agents in the M-GCPSO algorithm. The first two cases, Equations (6a) and (6b), are based on the GCPSO algorithm [15]. At each iteration, the scaling factor used in Equation (6b), ρ_t, is updated based on an adaptive search procedure as described in [15]. Note that the general form of Equations (6a) and (6b) is presented in [15] using the inertia model which is mathematically equivalent to the constriction model [10].

To further prevent the agents from concentrating within a small region in the search space [13], Equation (6c) is used in the M-GCPSO algorithm. As y_t^m and g_t^m are not involved in Equation (6c), the agents will not be forced to move towards their personal best and neighborhood best positions. This allows the agents to perform a broader search in the search space.

Note also that we implement the constriction model (6) while enforcing the limit on the agent's velocity (7) as this combination has been observed to increase the convergence rate [19]. In the case where x_{t+1}^m is outside the search space, the agent will be pulled back to the search area by projecting its position to the boundary line. The velocity of the agent in the direction orthogonal to the boundary line will be changed by taking a random value from the interval $(-1, 0)$ [20].

In Sections II-B and II-C, we will further describe how the incentive function, I, and motivation function, T_{res}, are used in the M-GCPSO algorithm.

B. The Incentive Function

In the motivational psychology theory, incentive is defined as a situational characteristic that relates to a motive's satisfaction [21]. It has been observed in [21] that individuals with different motive profiles have different preferences for certain kinds of incentives. As a result, they select goals differently. Within the context of task allocation, this incentive function is presented in this paper based on the situational characteristics 'distance to the task' and 'number of agents around the task'. Three motive profiles that have different preferences for this incentive are then presented.

Specifically in the M-GCPSO algorithm, the incentive function serves as a means of evaluating each potential neighborhood best position in the set $\mathbf{G}_t^{m*}$ [13]. Let $d_t^m(y)$ be the distance between the current position x_t^m and the point y, and $a(y)$ be the number of agents within a circle of a given radius h centered at y (in this paper, $h = 2$ is chosen as it has shown to provide good performance). Let $d_{t,max}^m$ be the maximum distance between agent s^m and any element of the set $\mathbf{G}_t^m$, $d_{t,max}^m = \max_{y_t^z \in \mathbf{G}_t^m} d_t^m(y_t^z)$. It is important to choose a scaling parameter, $d_{t,max}^m$, so that the set $\mathbf{G}_t^m$ can be normalized before it is augmented with the additional position y_t^*. Note that the value of $d_{t,max}^m$ yields zero when $x_t^m \in \mathbf{G}_t^m$.

To describe how the value of the incentive relates to the components of the distance and the number of agents around the point y, we introduce a function $I(a, D)$ in the M-GCPSO algorithm as follows:

$$I(a, D) = c_1(a) + c_2(a)e^{-(1-D)}e^{\alpha\left(\frac{M-a}{M}\right)}, \quad (9)$$

with $D \in [0, 1], a \in [0, M], \alpha > 0$, and functions c_1, c_2, are defined in Equation (11) below. The constant α defines how sensitive the incentive function is to changes in a. Here, we use $\alpha = 2.5$ as it creates the desired shape of the incentive curve.

Using Equation (9), the incentive function $I_t^m(y)$ is defined as:

$$I_t^m(y) = \begin{cases} I\left(a(y), \frac{d_t^m(y)}{d_{t,max}^m}\right), & \text{if } y \neq y^*, \\ I\left(0, 1\right), & \text{if } y = y^*. \end{cases} \quad (10)$$

There are two cases considered in Equation (10):

- Case $y \neq y^*$: This case determines the incentive value of position y in the set $\mathbf{G}_t^{m*}$ where y is not the additional point, y^*. In the case where $d_{t,max}^m = 0$, the value of $1 - \frac{d_t^m(y)}{d_{t,max}^m}$ is set to zero. To make the incentive function more sensitive to large distance changes, the distance component in the incentive function is expressed in an exponential form [13]. Furthermore, the component of the number of agents around the position y is also expressed in an exponential form, and thus makes the incentive more sensitive to changes in a.
- Case $y = y^*$: This case determines the incentive value assigned to the additional position y^*. In contrast to the first case where the incentive value depends on $d_t^m(y)$ and $a(y)$, the additional position y^* is assigned an

incentive $I_t^m(y^*) = c_1(0) + c_2(0)e^{\alpha} = I_t^m(0,1)$, which is the highest incentive value. This is to encourage some agents to pursue this position, and hence, encourage them to perform global exploration of the search space.

The incentive control parameters, $c_1(a)$, $c_2(a)$, control the shape of the incentive function and are set according to [22]:

$$c_1(a) = \begin{cases} c_1^{(1)} & \text{if } a \leq 2, \\ c_1^{(2)} & \text{if } a > 2, \end{cases} \tag{11a}$$

$$c_2(a) = \begin{cases} c_2^{(1)} & \text{if } a \leq 2, \\ c_2^{(2)} & \text{if } a > 2. \end{cases} \tag{11b}$$

In this paper, we use $c_1^{(1)} = 0.6, c_2^{(1)} = 0.025, c_1^{(2)} = 0, c_2^{(2)} = 0.05$ as these values have been empirically found to provide good performance. A discussion on the effect of different incentive values on the performance of the algorithm is provided in [22].

In Equation (11), two different profiles were chosen for $a \leq 2$ and $a > 2$. This is to give higher incentives to those positions where there are only two or fewer agents in the position y, and to provide lower incentives where the number of agents around y is greater than two. Fig. 1 shows the graph of the $I(a, D)$ function for all possible values of a in the case $M = 30$.

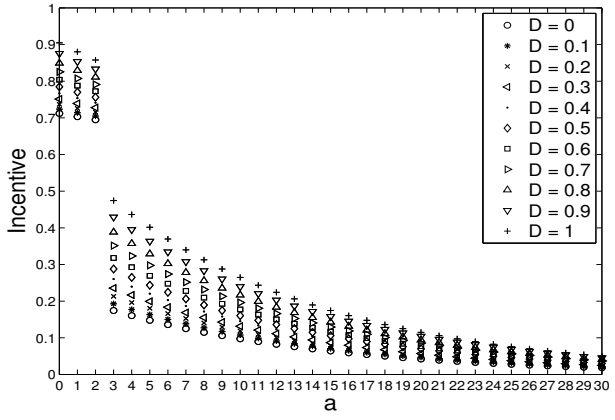


Fig. 1. The graph of $I(a, D)$ function for all possible values of a, in the case M=30 and $D \in [0, 1]$.

C. The Motivation Function

To mimic certain aspects of human behavior in goal selection, the shape of the motivation curve for the agents is constructed by replicating the shape suggested by psychological studies of human behavior [14]. While in the MPSO algorithm [13] we have constructed affiliation-motivated agents which have a high preference for goals with lower incentives (Profile 1), this paper considers a wider range of agents by introducing two new motive profiles, namely Profiles 2 and 3. Compared to the motive profiles proposed in [13], the new profiles are significantly different in that they exhibit different goal-selection characteristics for certain kinds of incentives. In this paper, Profile 2 is characterized by a high preference for goals with intermediate incentives and a moderate preference for goals with high incentives, while Profile 3 is characterized by a high preference for goals with high incentives and a moderate preference for goals with intermediate incentives. By endowing agents with one of the three motive profiles, we expect the agents to exhibit different characteristics such as those who tend to: prioritize allocation to tasks (Profile 1), prioritize allocation but also have an intermediate tendency to explore (Profile 2), and prioritize exploration but also have an intermediate tendency to allocate (Profile 3). The motivation curves of the corresponding motive profiles 1-3 are presented in Figs. 2(a)-2(c).

In this paper, the three motive profiles were defined and implemented using a motivation function, $T_{res}(I)$, of the form:

$$T_{res}(I) = T_1(I) + T_2(I) + T_3(I)$$

$$\begin{aligned} T_{res}(I) = & \frac{S_1}{1+e^{\rho_1^+(I-M_1^+)}} - \frac{S_1}{1+e^{\rho_1^-(I-M_1^-)}} \\ & + \frac{S_2}{1+e^{\rho_2^+(M_2^+-(1-I))}} - \frac{S_2}{1+e^{\rho_2^-(M_2^--(1-I))}} \\ & + \frac{S_3}{1+e^{\rho_3^+(M_3^+-I)}} - \frac{S_3}{1+e^{\rho_3^-(M_3^--I)}} \end{aligned}$$

where I is the incentive variable; $T_\theta(I)$ is the motivational tendency; M_θ^+, M_θ^- are each the turning points of approach and avoidance of a goal; ρ_θ^+, ρ_θ^- are gradients of approach and avoidance of a goal respectively; S_θ denotes the relative motivation strength that controls the strength of a particular motive compared to other motives; and $\theta = 1, 2, 3$ represent the constant indexes corresponding to the motivational tendencies for choosing goals with low, intermediate, and high incentives respectively.

Using Equation (12) and the constants presented in Table I, we created the three motive profiles mentioned above as shown in Figs. 2(a)-2(c). In this paper, the shape of the motivation profiles were selected as those shapes have been empirically found to create agents with desirable behaviors in task allocation.

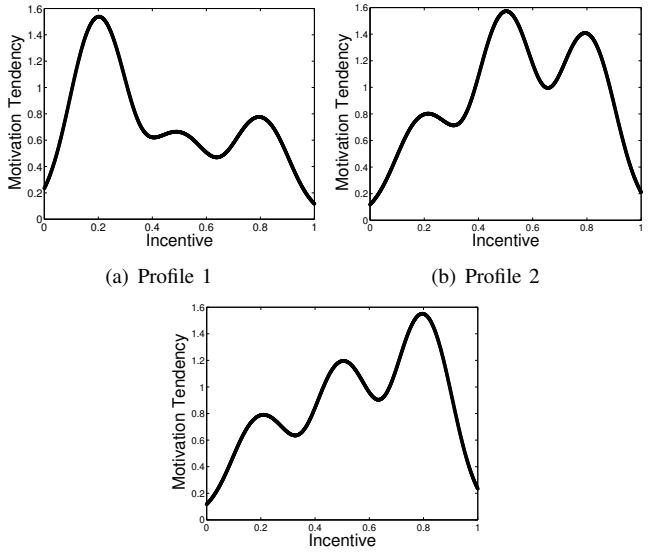
III. Validation and Performance of the M-GCPSO Algorithm

To evaluate the efficacy of the new algorithm where motivation is used, we first compare the proposed algorithm with the corresponding GCPSO variant [15] that does not employ motivated agents. In this paper, the GCPSO algorithm is configured with the same communication constraints as defined in Equation (1). Note that the standard GCPSO algorithm differs from the M-GCPSO algorithm in that the neighborhood best position g_t^m is calculated using the sensed value $f(y_t^z)$ directly, i.e., $g_t^m = \arg\max_{y_t^z \in \mathcal{S}^m}(f(y_t^z))$. Furthermore, at each iteration, the agents in the GCPSO algorithm only update their velocity based on Equations (6a) and (6b).

TABLE I

CONSTANTS OF THE MOTIVATION FUNCTION

Constants	M-GCPSO		
	Profile 1	Profile 2	Profile 3
S_1	2	1	1
M_1^+	0.3	0.3	0.3
M_1^-	0.1	0.1	0.1
ρ_1^+	20	20	20
ρ_1^-	20	20	20
S_2	0.8	2	1.5
M_2^+	0.4	0.4	0.4
M_2^-	0.6	0.6	0.6
ρ_2^+	20	20	20
ρ_2^-	20	20	20
S_3	1	1.8	2
M_3^+	0.7	0.7	0.7
M_3^-	0.9	0.9	0.9
ρ_3^+	20	20	20
ρ_3^-	20	20	20



(a) Profile 1 (b) Profile 2

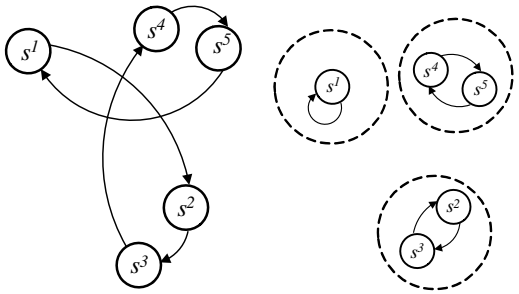
(c) Profile 3

Fig. 2. Three different motive profiles that are used in the Motivated-Guaranteed Convergence Particle Swarm Optimization algorithm.

The second algorithm to compare with the proposed algorithm is the MPSO algorithm in [22] that is configured in this paper under the same communication constraints as defined in Equation (1). This algorithm is chosen to evaluate how the proposed modification of the MPSO algorithm that is introduced in this paper and described in Section II improves the performance of the MPSO algorithm under limited communication constraint. A comparison between the topology that is used in the original MPSO and the one used for the MPSO algorithm in this paper is shown in Fig.3.

A. Metrics

In this section, we summarize the definitions of performance metrics introduced in [22] and used in this paper.



(a) (b)

Fig. 3. An example of different communication topologies of the MPSO algorithm when the agents are positioned as shown. (a) The communication topology of the original MPSO with unlimited communication constraints [13], (b) The modified communication topology of the MPSO algorithm with limited communication constraints.

1) Average number of tasks to which the agents are allocated [22]: Task n is said to be allocated if there is at least one agent that stays at the task for at least τ consecutive time steps. For the simulations, we use $\tau = 20$. Notice that the value of τ can be set depending on how long one wants an agent to stay at the task to consider the agent as being allocated to the task. The average number of tasks to which the agents are allocated over J simulations is computed as $\bar{Q} = \frac{1}{J}\sum_{j=1}^{J}\hat{Q}_j$, where $\hat{Q}_j$ refers to the number of tasks to which the agents are allocated in simulation j.

2) Average number of discovered tasks [22]: Task n at $\acute{x}_n$ is said to be discovered if there is at least one agent s^m such that it satisfies $\|x_t^m - \acute{x}_n\| \leq \varepsilon$. In this case, we specify $\varepsilon = 0.4$ which indicates that an agent has discovered a task if its distance to the task position is ≤ 0.4. Note that this value can be set according to the desired level of accuracy. The average number of discovered tasks over J simulations is calculated as $\bar{W} = \frac{1}{J}\sum_{j=1}^{J}\hat{W}_j$ where $\hat{W}_j$ is the number of discovered tasks in simulation j.

3) Distribution of agents among tasks [22]: As mentioned earlier in this paper, the desired distribution from the perspective of this paper is to have a uniform distribution of agents among the tasks. To quantify how well the agents distribution among the tasks follows this desired distribution, we present a relative entropy type metric to measure the difference between the observed population density of allocated agents at each task, p_O, and the uniform distribution, p_U. In this case, a uniform distribution of agents refers to an ideal situation where each task $n \in \{1, \ldots, N\}$ is allocated the same number of agents.

Let $\hat{M}^n$ denotes the number of agents allocated to task n and $\hat{M}$ is the total number of allocated agents. The observed probability (relative frequency) distribution of agents among N tasks is computed as $p_O(n) = \frac{\hat{M}^n}{\hat{M}}, \quad n = 1, \ldots, N$. Note that $p_O \in \mathbb{R}^N$, $p_O \geq 0$ and $\sum_{n=1}^{N} p_O(n) = 1$.

The uniform distribution of agents among N tasks, on the other hand, is defined as $p_U(n) = \frac{1}{N}, \quad n = 1, \ldots, N$. To quantify the discrepancy of p_O from p_U, the relative entropy

of the two distributions is calculated as follows:

$$R(p_O||p_U) = \sum_{n=1}^{N} p_O(n) \log \frac{p_O(n)}{p_U(n)}, \quad (12)$$

where $R(p_O||p_U) \geq 0$, with equality only if $p_O = p_U$. Here, we follow the convention that $p_O(n) \log \frac{p_O(n)}{p_U(n)} = 0$, whenever $p_O(n) = 0$, according to the continuity arguments given in [23]. Note this definition requires that $p_U > 0$ for all n which is the case for the uniform distribution.

According to the above definition, a relative entropy of 0 indicates a situation where the observed density is equal to the reference density. Even though the quantity of relative entropy is not a true distance between probability distributions in the mathematical sense, it is commonly used to assess how close a probability distribution is to a reference. Lower values of relative entropy are interpreted as an indication of a stronger similarity between the probability distributions compared. In our case, lower values of relative entropy point to a more uniform distribution of the agents among the tasks.

To obtain sufficient statistical evidence of how uniformly the agents are distributed among the tasks, multiple simulations are conducted in this study. We hence consider the average relative entropy over $\check{J}$ simulations which is defined by:

$$\hat{R}(p_O||p_U) = \frac{1}{\check{J}} \sum_{j}^{\check{J}} R_j(p_O||p_U), \quad (13)$$

where $R_j(p_O||p_U)$ is the relative entropy between the observed and uniform distributions of agents among tasks in simulation j.

B. Simulation Setup

The simulations were conducted using Matlab version 7.8.0 ($R2009$a). To evaluate the performance of the algorithms in a more realistic situation, the parameters used in our simulations are derived from the multi-agent search scenario in [4]. That is, we use a fixed communication range (δ) of 2.0 m, and set the agent's maximum velocity and time per iteration (T) to 0.1287 m/s and 9.6 s respectively based on [4]. To represent tasks with equal priority, a multimodal test function based on [24] is applied and modified to have equal maxima values, $f(x_1, x_2) = \max_{\forall n} f_n(x_1, x_2)$, with $f_n(x_1, x_2)$ defined as $f_n(x_1, x_2) = \frac{H_n}{1+\omega_n[(x_1-X_n)^2+(x_2-Y_n)^2]}$. Here, (X_n, Y_n) denotes the coordinates of task n, H_n denotes the significance (level of priority) of task n, and ω_n determines how fast the signals from task n reduce with distance. All tasks were set with $\omega = 1$ and $H = 1$. This setting gives all tasks an equal priority, hence we expect that agents will allocate themselves to the tasks evenly.

To maintain the same task density as in [4], the size of the search space used in the simulations is adjusted following the scale used in [4]. The search space is defined in the range of $0 < x_1 < 24$ and $0 < x_2 < 24$ with nine tasks ($N = 9$) distributed in the search region with fixed positions. The value of v_{max} in the algorithm is obtained by multiplying the agent's maximum velocity with the time per iteration.

In all simulations, the algorithms were tested using 30 agents, enough to allocate agents to all the tasks. For the M-GCPSO algorithm, a proportion of 2:2:1 between the number of agents with Profile 1, Profile 2, and Profile 3 was chosen. This choice was made to maintain similar proportion as used in the MPSO algorithm [13] where a higher proportion is given to the types of agents who tend to prioritize allocation and a lower proportion to the agents who tend to prioritize exploration. In the previous MPSO algorithm [13], a ratio of 3:2 between the affiliation- and power-motivated agents was used for a small number of agents and a ratio of 4:1 was set for a large number of agents. In this paper, we assume that 30 agents falls in the category of a large number of agents, and thus a proportion of 4:1 is chosen. However, as there are two types of agents in the M-GCPSO algorithm that tend to prefer allocation (i.e. agents with Profiles 1 and 2), we divide the proportion for those two types of agents into two equal proportions. Thus, a proportion of 4:1 in the previous MPSO algorithm is considered to be equal to a proportion of 2:2:1 in the M-GCPSO algorithm. In these simulations, the GCPSO algorithm is set with 30 homogenous agents. The MPSO algorithm, on the other hand, uses a proportion of 4:1 between the affiliation-motivated and power-motivated agents as considered in the previous paper in [13].

Furthermore, the number of agents used in the simulations is denoted here as numbers within brackets. M-GCPSO (12+12+6), for instance, means that the M-GCPSO algorithm is generated using 12 agents with Profile 1, 12 agents with Profile 2, and 6 agents with Profile 3.

In this paper, we conducted four simulations as follows:

1) Simulation 1: This simulation is designed to investigate the effect of varying communication ranges in the case where the agents are initialized from random points. In this simulation, nine tasks are distributed in the search space and a total of 30 agents are employed. For the purpose of measuring the impact of having different communication ranges, in this simulation we focus on performance in terms of the average number of tasks to which agents are allocated.
2) Simulation 2: This simulation has the same setup as Simulation 1, but the agents are initialized from a single point. In this simulation, we again focus on the number of tasks to which the agents are allocated as the performance metric.
3) Simulation 3: The third simulation is conducted to investigate the performance of the algorithms using broader performance metrics that are presented in Section III-A. In this simulation, the agents are initialized from random positions. This simulation is tested with nine tasks and 30 agents. Different from Simulations 1 and 2, for this simulation the value of the communication range is fixed to 2 as used in the multi-agent search scenario proposed in [4].
4) Simulation 4: This simulation has the same config-

uration as Simulation 3, except that the agents are initialized from a single point. In this simulation, the same set of metrics as used in Simulation 3 is employed.

The maximum number of iterations is set to 300. To obtain a significant statistical results, the simulations were repeated 50 times with different random seeds. All other parameters take values specified in Section II.

C. Results

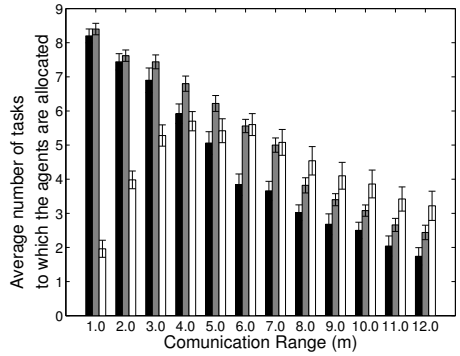
1) Effect of varying communication ranges (Simulations 1 and 2): The results of the first simulation indicate that both the M-GCPSO and GCPSO algorithms have a higher number of tasks allocated when smaller communication ranges are used as shown in Fig. 4(a). As the communication range increases, fewer sub-swarms are created, and hence, the performance of the two algorithms gradually decreases. This suggest that having a very large communication range can potentially degrade the performance of the M-GCPSO and GCPSO algorithms. Conversely, the performance of the MPSO algorithm under limited communication constraints is relative poor as it never manages to allocate agents to more than 70% of the total number of tasks; see Fig. 4(a).

In Simulation 2, there is a decrease in the number of allocated tasks compared to the first simulation as can be seen in Fig.4(b). This is because a smaller part of the search area is initially covered by the agents with a single point initialization. Compared with the M-GCPSO algorithm, the performance of the GCPSO and MPSO algorithms in this simulation drops significantly and remains poor regardless the communication range.

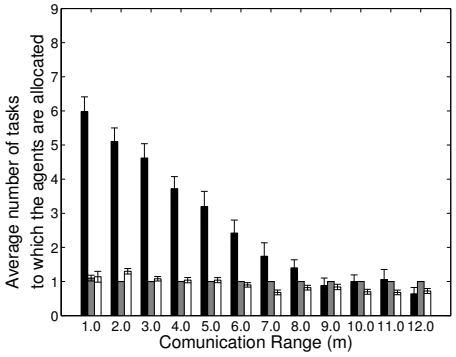
2) Effect of initial positions (Simulations 3 and 4): Table II summarizes the performance of the algorithms under different initial positions after the maximum iterations has been reached. Different from the previous two simulations, the performance of the algorithms in Simulations 3-4 is quantified using broader metrics which also include the average number of discovered tasks as well as the distribution of agents among tasks.

The results of Simulation 3 in Table II (column 3) indicate that the M-GCPSO algorithm substantially discovers more tasks than the GCPSO algorithm at the 95% confidence level. Compared with the GCPSO algorithm, the M-GCPSO algorithm has a mechanism to ensure that agents who have higher values of incentive function are able to carry out a wider exploration. This mechanism significantly contributes to the increase in the number of discovered tasks. In terms of the average number of tasks allocated, it is shown in Table II (column 4) that the performance between the M-GCPSO and GCPSO algorithms is comparable when the agents are initialized randomly. In this case, there is no significant difference at the 95% confidence interval between the two algorithms in how uniformly the agents are distributed among the tasks; see Table II (column 5).

Furthermore, it can be seen in Table II (column 3-4) that the M-GCPSO algorithm in Simulation 3 significantly increases the number of discovered tasks and allocate the agents to more tasks than the MPSO algorithm. The M-GCPSO algorithm also distributes the agents more uniformly among the tasks which is indicated by having a lower relative entropy than the MPSO algorithm; see the last column of Table II.



(a) Simulation 1



(b) Simulation 2

Fig. 4. The effect of different communication ranges on the average number of tasks to which the agents are allocated for M-GCPSO (12+12+6) (black), GCPSO (30) (gray), and MPSO (24+6) (white) in the case where the agents are initialized from: (a) random points, (b) a single point.

In the case where the agents are initialized from a single point (Simulation 4), the M-GCPSO algorithm achieves a significant higher number of discovered tasks and allocates the agents to more tasks than the GCPSO algorithm (Table II (column 3-4)). In this simulation, the M-GCPSO algorithm also distributes the agents more uniformly among the tasks. This is indicated by having a lower relative entropy value than the GCPSO algorithm which is statistically significant at the 95% confidence level.

In comparison to the MPSO algorithm, the M-GCPSO algorithm in Simulation 4 has a higher number of tasks discovered and allocated as can be seen in Table II (column 3-4). In this case, the new algorithm also improves the performance of its previous version by distributing the agents more uniformly among the tasks; see Table II (column 5).

Note that there is a decrease in the overall performance of the M-GCPSO algorithm when the agents are initialized from a single point. It has been observed that this is because the agents with Profile 1 in this case did not critically contribute to the success of the system. When the agents are initialized

TABLE II

PERFORMANCE COMPARISON OF THE MOTIVATED-GUARANTEED CONVERGENCE PSO (M-GCPSO), GUARANTEED CONVERGENCE PSO (GCPSO), AND THE MOTIVATED PSO (MPSO) ALGORITHMS IN THE CASE $N = 9$.

		Average number of discovered tasks	Average number of tasks to which the agents are allocated	Average relative entropy
Random Initialization (Simulation 3)	M-GCPSO (12+12+6)	8.600±0.148	7.440±0.238	0.452±0.052
	GCPSO (30)	8.060±0.205	7.620±0.167	0.431±0.036
	MPSO (24+6)	7.080±0.262	3.980±0.260	1.499±0.107
Single Point Initialization (Simulation 4)	M-GCPSO (12+12+6)	6.980±0.338	5.100±0.401	1.521±0.119
	GCPSO (30)	1.000±0.000	1.000±0.000	3.170±0.000
	MPSO (24+6)	2.580±0.217	1.300±0.232	2.424±0.301

from a single point, these agents tend to prefer a position where there is a large number of agents around that position and converge into a single task. This shows that in this simulation the agents with Profile 1 posses the same behavior as the ordinary GCPSO agents in that they prefer to stay at the task rather than exploring the search space when they are initialized from a single point.

IV. CONCLUSION

In this paper, we have presented a novel approach to solve a task allocation problem under the constraint of limited communication. This approach incorporates models of motivation into the guaranteed convergence particle swarm optimization algorithm. In comparison with the basic approach without motivation, the new approach has significantly increased the number of discovered tasks given that the agents are initialized from random points. Moreover, it has significantly outperformed the PSO benchmark algorithm in terms of tasks discovered and allocated when the agents are initialized from a single point. Results of the simulations also indicate that the new approach has significantly improved the ability of the original approach when the agents can only communicate within a predefined communication range.

One potential direction for the future research is to conduct further analytical investigation of the sensitivity of different parameter values to the performance of the algorithm. Another relevant future work is to investigate the effect of changing the proportion of each type of agents since at this moment this is still set a priori. It would also be interesting to explore other situational characteristics for the incentive function. These include the level of difficulty (priority) of a task as well as the distance between agents. Such characteristics can potentially be incorporated in the incentive function to cover broader aspects of task allocation problems.

REFERENCES

[1] Y. Meng and J. Gan, "Self-adaptive distributed multi-task allocation in a multi-robot system," in *IEEE Congress on Evolutionary Computation (CEC-2008)*, Jun. 2008, pp. 398–404.

[2] D. W. Gage, "Many-robot MCM search systems," in *Proceedings of Autonomous Vehicles in Mine Countermeasures Symposium*, 1995, pp. 9–55.

[3] B. Lopez, S. Suarez, and J. L. De La Rosa, "Task allocation in rescue operations using combinatorial auctions," *Artificial Intelligence Research and Development*, vol. 100, pp. 233–243, 2003.

[4] J. Pugh and A. Martinoli, "Inspiring and modeling multi-robot search with particle swarm optimization," in *IEEE Swarm Intelligence Symposium (SIS-2007)*, Apr. 2007, pp. 332–339.

[5] B. Gerkey and M. Mataric, "Multi-robot task allocation: analyzing the complexity and optimality of key architectures," in *IEEE International Conference on Robotics and Automation (ICRA '03)*, vol. 3, 2003, pp. 3862–3868.

[6] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.

[7] S. B. Akat, V. Gazi, and L. Marques, "Asynchronous particle swarm optimization-based search with a multi-robot system: simulation and implementation on a real robotic system," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 18, no. 5, pp. 749–764, 2010.

[8] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *IEEE International Conference on Neural Networks*, vol. 4, Nov. - Dec., 1995, pp. 1942–1948.

[9] J. Hereford, M. Siebold, and S. Nichols, "Using the particle swarm optimization algorithm for robotic search applications," in *IEEE Symposium on Swarm Intelligence (SIS-2007)*, Apr. 2007, pp. 53–59.

[10] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*. John Wiley & Sons, 2006.

[11] K. Derr and M. Manic, "Multi-robot, multi-target particle swarm optimization search in noisy wireless environments," in *2nd Conference on Human System Interactions (HSI '09)*, May 2009, pp. 81–86.

[12] D. Liu, X. Zhou, A. Liang, and H. Guan, "A swarm intelligence based algorithm for distribute search and collective cleanup," in *IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS '10)*, vol. 2, Oct. 2010, pp. 161–165.

[13] M. K. Hardhienata, K. E. Merrick, and V. Ugrinovskii, "Task allocation in multi-agent systems using models of motivation and leadership," in *IEEE Congress on Evolutionary Computation (CEC-2012)*, Jun. 2012, pp. 1–8.

[14] K. E. Merrick and K. Shafi, "Achievement, affiliation, and power: Motive profiles for artificial agents," *Adaptive Behavior*, vol. 19, no. 1, pp. 40–62, 2011.

[15] E. Peer, F. van den Bergh, and A. Engelbrecht, "Using neighbourhoods with the guaranteed convergence pso," in *IEEE Swarm Intelligence Symposium (SIS-2003)*, 2003, pp. 235–242.

[16] P. Whittle, *Risk-sensitive optimal control*. John Wiley & Sons, 1990.

[17] S. Akat and V. Gazi, "Particle swarm optimization with dynamic neighborhood topology: Three neighborhood strategies and preliminary results," in *IEEE Swarm Intelligence Symposium (SIS-2008)*, Sept. 2008, pp. 1–8.

[18] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, Feb. 2002.

[19] R. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *IEEE Congress on Evolutionary Computation (CEC-2000)*, vol. 1, 2000, pp. 84–88.

[20] S. Xu and Y. Rahmat-Samii, "Boundary conditions in particle swarm optimization revisited," *IEEE Transactions on Antennas and Propagation*, vol. 55, no. 3, pp. 760–765, Mar. 2007.

[21] H. Heckhausen and P. K. Leppmann, *Motivation and action.* Springer-Verlag Publishing, 1991.

[22] M. K. Hardhienata, V. Ugrinovskii, and K. E. Merrick, "Models of motivation for particle swarm optimization-based task allocation," submitted for publication.

[23] T. M. Cover and J. A. Thomas, *Elements of information theory*. New York, NY, USA: Wiley-Interscience, 1991.

[24] W. Cedeno and V. Vemuri, "On the use of niching for dynamic landscapes," in *IEEE International Conference on Evolutionary Computation*, Apr. 1997, pp. 361–366.