1

1

Self-organized Multi-robot Task Allocation using Attractive Field Model: Comparison of Centralized and Local Communication

Md Omar Faruque Sarker and Torbjørn S. Dahl

***Abstract*—This paper proposes to solve multi-robot task-allocation (MRTA) problem using a set of previously published generic rules for self-organised division of labour derived from the observation of ant, human and robotic social systems. The concrete form of these rules, the *attractive filed model* (AFM), provide sufficient abstraction to accommodate different sensing and communication models. We have validated the effectiveness of AFM for MRTA using two bio-inspired communication and sensing strategies: *global sensing - no communication* and *local sensing - local communication*. The former is realized using a centralized communication system and the latter is emulated as a peer-to-peer local communication scheme. Both strategies have been evaluated using 16 e-puck robots.**

I. Introduction

MULTI-ROBOT systems can provide high levels of performance, fault-tolerance and robustness in complex and distributed tasks through parallelism and redundancy [1]. *But how those tasks can be allocated dynamically among multiple robots?* Traditionally, this issue has been identified as *multi-robot task allocation* (MRTA) [2], analogous to *division of labour* (DOL) in biological and human social systems [3]. MRTA is generally defined as the problem of assigning tasks in an appropriate time to the appropriate robots considering the changes in task-requirements, team-performance and the environment [4]. The complexities of the *distributed* MRTA problem arise from the fact that there is no central planner or coordinator for task assignments, and in a large multi-robot system, robots generally have limited abilities to sense, to communicate and to interact locally.

M. O. F. Sarker and T. S. Dahl is with Cognitive Robotics Research Centre, University of Wales, Newport, Newport Business School, Allt-yr-yn Campus Newport, NP20 5DA, United Kingdom. e-mail: Mdomarfaruque.Sarker@newport.ac.uk

Within the context of the Engineering and Physical Sciences Research Council (EPSRC) project, "Defying the Rules: How Self-regulatory Systems Work", we have developed the *attractive field model* (AFM) [5], a model of self-regulated DOL formed by the studies of self-regulated task-allocation in ant and human social systems. AFM suggests four generic rules to explain self-regulation in those social systems. These four rules are: *continuous flow of information*, *concurrency*, *learning* and *forgetting*. Primarily, these rules help to derive local control laws for regulating an individual's task-allocation behaviour in such a way that facilitates DOL in the entire group.

In biological social systems, communications among the group members, as well as the perception of tasks, are two key components of self-organized DOL. In robotics, existing self-organized task-allocation methods rely heavily upon local sensing and local communication between individuals. AFM however, differs significantly in this point. Being a highly abstract model, it avoids a strong dependence on any specific communication strategy such as the local communication and interaction found in many existing approaches to MRTA. AFM uses the abstraction of attractive fields or continuous flows of information from tasks to agents. In order to enable continuous flow of information in our multi-robot system, we have implemented two types of sensing and communication strategies inspired by the self-regulated DOL found in two types of social wasps: *polistes* and *polybia* [6]. Depending on the

group size, these species follow different strategies for communication and sensing of tasks. Polistes wasps are called *independent founders* in which reproductive females establish colonies alone or in small groups (in the order of 10^2), but independent of any sterile workers. Polybia wasps, on the other hand, are called *swarm founders* in which a swarm of workers and queens initiate colonies consisting of several hundreds to millions of individuals. Independent founders do not rely on any cooperative task performance while swarm founders interact with each-other locally to accomplish their tasks. The work mode of independent founders can be considered as *global sensing - no communication (GSNC)* where the individuals sense the task requirements throughout a small colony and do these tasks without communicating with each other. The work mode of the swarm founders, on the other hand, can be treated as *local sensing - local communication (LSLC)* where the individuals can only sense tasks locally due to the large colony size and can communicate locally to exchange information, e.g. task-requirements. This study emulates these two types of wasps behaviours in a multi-robot manufacturing shop-floor scenario using AFM.

The main contributions of this paper are as follows. This study compares the performance of two bio-inspired sensing and communication strategies in producing self-regulated MRTA. Besides, this study presents a novel taxonomy of MRTA solutions derived based on the organization of task-allocation, interaction and communication among robots.

This paper has been organized as follows. Section 2 reviews the related literature on MRTA. Section 3 presents the new taxonomy of MRTA solutions. Section 4 presents AFM based task-allocation solution for multi-robot systems. Section 5 describes our centralized and local communication schemes used in this study. Section 6 and Section 7 present the design and implementation of our experiments respectively. Section 8 describes our results. Section 9 discusses the performance of self-regulated MRTA under both centralized and local communication strategies. Section 10 presents conclusions and directions for future works.

II. Related work

Since the 90s, many robot control architectures have been designed and evolved to address various MRTA issues from different perspectives. Based on the high-level design of those solutions, we have classified them into two broad categories: i) *explicit* task-allocation and ii) *bio-inspired self-organized* task-allocation. Early research on explicit task-allocation was dominated by explicit cooperation [7], use of dynamic role assignment [8] and market-based bidding approach [9]. Under these approaches, robots perform explicit task-allocation activities, commonly including communication with other group members for negotiating tasks. These approaches are intuitive and comparatively straight forward to design and implement and can be analysed formally. However, these approaches typically work well only when the number of robots are small (≤ 10) [10].

Task performance in self-organized approaches, on the other hand, relies on the collective behaviours resulting from the local interactions of many simple and mostly homogeneous or interchangeable agents. Robots choose their tasks independently using general principles of self-organization such as: positive and negative feedback, multiple interactions among entities and their environment, and randomness or amplification of fluctuations [11]. Moreover interaction among individuals and their environment can be modulated by stigmergic, local and global communications. Among many variants of self-organized task-allocation mechanisms, the most common type is response threshold-based task-allocation [12]. In this approach, a robot's decision to select a particular task depends largely on its perception of a stimulus, e.g., the demand for a task to be performed, and its corresponding response threshold for that task.

Under the deterministic response-threshold approach, each robot has an activation threshold for each task that needs to be performed. When a particular task-stimulus exceeds a predefined threshold the robot starts working on that task, typically reducing their related stimuli. When the task-stimuli falls below the fixed threshold the robot abandons

that task. This type of approach has been effectively applied in foraging [13], [14] and aggregation [15]. The fixed response threshold can initially be same for all robots [16] or they can be different according robot capabilities or the configuration of the system [14]. Adaptive response-threshold models change the thresholds over time. A robot's response threshold for a given task is often decreased as a result of the robot performing that task. This enables a robot to select that particular task more frequently or in other words specialise on that task [12], [15]. Unlike the deterministic approach, probabilistic approaches offer a selection process based-on a probability distribution over the tasks. In this case, all tasks commonly have at least a small, non-zero probability of being chosen. This random element typically prevents starvation of low-stimulus tasks. Bio-inspired, self-organized MRTA mechanisms rely to a much smaller degree on modelling the environment, tasks or robot capabilities. Most existing research in this area considers a single global task e.g. foraging, area cleaning and box-pushing with the focus of the work being the design of individual robot controllers that can accomplish the task [4].

Both of the above task-allocation approaches have their relative strengths and weaknesses. In an arbitrary event handling domain, a comparison between self-organized and predefined market-based task-allocation was reported in [17]. Here it is found that explicit task-allocation was more efficient when the required information could be captured accurately. The threshold-based approach offered similar quality of allocation at a fraction of cost in noisy environment. Gerkey and Mataric [4] compared the complexity and performance of key architectures, including ALLIANCE [18], the Broadcast of local eligibillity (BLE) architecture [19], M+ [20], MURDOCH [21], First piece auctions [22] and Dynamic role assignment [8]. All of these rely on explicit task-allocation mechanisms. The computational and communication requirements of these MRTA systems were expressed as a function of the number of robots and tasks. Although the study did not explicitly measure the scalability of the architectures, it clearly shown that many explicit task-allocation mechanisms fail to scale well in environments where the number of robots and tasks are large, given limited overall communication bandwidth and limited processing power for individual robots.

III. A TAXONOMY OF MRTA

In order to understand the interplay among task-allocation, communication and interaction of a particular MRTA approach we have presented these aspects in two 2D characteristic plots. From this classification we can get an insight about the complexities involved in various kinds of MRTA problems and the design choices involved in them.

Fig. 1 presents the classification of MRTA solutions based on task-allocation and communication strategies used to disseminate information. Here horizontal axis shows the number of active nodes that provides the task-allocation to the group . The origin of this axis represents the explicit task-allocation by a single centralized entity. For example, in any explicit task-allocation approach, we can use one external centralized entity or one of the robots (aka *leader*) to manage the task-allocation process. In many predefined methods, e.g. in market-based systems, multiple nodes can act as mediators or task-allocators. Under predefined task-allocation approach, a small number of robots can have fully distributed task-allocation where each robot acts as an independent task-allocator (e.g. as in ALLIANCE architecture).

The right extreme of horizontal axis in Fig. 1 represents the swarm task-allocation method which is fully distributed, i.e. each robot selects its tasks independently and without the help of a leader. However, robots might be dependent on other entities for getting the latest *information* about tasks. In a recent study on swarm-robotic flocking, it was shown that a swarm can be guided to a target by a few informed individuals or leaders while the remaining robots remain self-organized [23]. Task-allocation in a swarm of robots through one central entity may be rare since one of the major motivations behind swarm robotic systems is to make the swarm task-allocation more robust by avoiding single point of failure.

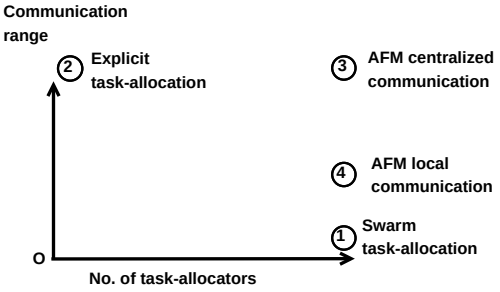


Fig. 1: Classification of MRTA solutions based on task-allocation and communication strategies

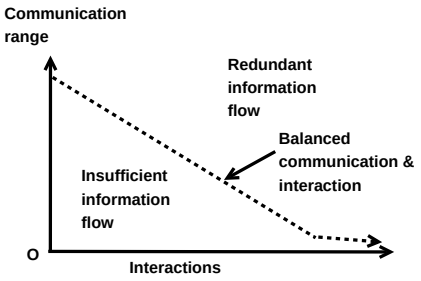


Fig. 2: Information flow caused by different levels of communication and interaction

The vertical axis in Fig. 1 shows the communication range of individual entities. In case of implicit communication via stigmergy in swarm task-allocation this range has been considered as zero. On the other hand, in case of explicit task-allocation a global range can cover all entities. In this research we have placed our two kinds of communication strategies vertically aligned with swarm task-allocation. *AFM centralized communication* represents a GSNC based system where communication range is global and task-allocation is fully distributed. Similarly, AFM local communication also represents the the fully distributed task-allocation but with limited communication range under LSLC strategy.

The information flow caused by different levels of communication and interaction has been depicted in Fig. 2. From this plot we can see the two extremes of information flow. *Insufficient information flow* occurs when individual's communication range is low and their interaction is limited. In such cases, system designer can find suitable trade-off to get a balanced information flow by increasing communication range or interaction frequency or both. On another extreme of this situation, we can find lots of *redundant information flow* where all individuals are both communicating and interacting. Thus the dotted line in the plot refers to a balanced communication and interaction. This dotted line does not completely intersect the horizontal line since when the communication range is limited to the next neighbour, individuals can still continue to interact with others as much as possible.

The presence of interaction can be due to the nature of the problem, e.g. cooperation is necessary in co-operative transport tasks. Alternately, this interaction can be a design choice where interaction can improves the performance of the team, e.g. cooperation in cleaning a work-site is not necessary but it can help to improve the efficiency of this task. The communication overhead can be resulted from the interactions of the robots. Various task-allocation methods rely upon variable degrees of robot-robot communications. The communication capabilities of individual robots can limit (or expand) the level of interaction that can be made in a given group. Thus in one way, considering the interaction requirements of a MRTA problem, the system designer can select suitable communication strategies that both minimizes the communication overhead and maximizes the performance of the group. Alternately, the communication capabilities of the robots can guide a system designer to design interaction rules of a robot team, e.g. the specification of a robot's on-board camera can dictate the degree of possible visual interactions among robots. The suitable trade-off between these two axes can give a balanced design of MRTA solution.

IV. Task-allocation using AFM

AFM provides an abstract framework for solving task-allocation problem among multiple agents. The mathematical model has been discussed in [5] and an experimental validation of AFM in multi-robot system has been reported in [24]. Here we have briefly summarized the core concepts of AFM within the context of a manufacturing shop-floor scenario.

AFM formalises the four requirements of self-organised social systems (Section 1) in terms of the relationships between properties of individual agents and of the task-allocation system as a whole. Here, tasks can be described as the sources of the attractive fields with varying levels of urgencies. They encode the strength of the attractive fields as perceived by the agents. All communication is considered part of the attractive fields. There is also a permanent field representing the *no-task* option of not working in any of the available tasks. This option is modelled as a random walk. The edges of the AFM network between tasks and agents are weighted and the value of this weight describes the strength of the stimulus as perceived by the agents. In a spatial representation of the model, the strength of the field depends on the physical distance of the agent to the source. The strength of a field is increased through the sensitisation of the agent through experience with performing the task.

In our robotic interpretation of AFM, each robot is modelled as an agent and each task is modelled as a spatial location. The robots repeatedly select tasks and if the robot is outside a fixed task boundary, it navigates towards the task. If the robot is within the task boundary it remains there until the end of the time step when a new (or the same) task is selected. Under our manufacturing shop-floor scenario, each task represents a manufacturing machine that is capable of producing goods from raw materials, but they also require constant maintenance works for stable operations.

Let W_j be a finite number of material parts that is loaded into a machine j in the beginning of its production process and in each time-step, ω_j units of material parts can be processed ($\omega_j \ll W_j$). So let Ω_j^p be the initial production workload of j which is simply: W_j/ω_j unit. We assume that all machines are identical. In each time step, each machine always requires a minimum threshold number of robots, *(μ)*, to meet its constant maintenance work-load, Ω_j^m unit. However, if μ or more robots are present in a machine for production purpose, we assume that, no extra robot is required to do its maintenance work separately.

Now let us fit the above production and maintenance work-loads and task performance of robots into a unit task-urgency scale. Let us divide our manufacturing operation into two subsequent stages: i) *production cycle* (production and maintenance tasks), and ii) *maintenance cycle* (maintenance tasks only). Initially a machine starts working in production cycle. When there is no production work left, then it enters into maintenance cycle. Under both modes, let α_j be the amount of workload occurs in a unit time-step if no robot serves a task and it corresponds to a fixed task-urgency $\Delta\phi_{INC}$. On the other hand, let us assume that in each time-step, a robot, i, can decrease a constant workload β_i by doing some maintenance work along with doing any available production work. This corresponds to a negative task urgency: $-\Delta\phi_{DEC}$. So, at the beginning of production process, task-urgency, occurred in a machine due to its production work-loads, can be encoded by Eq. 1.

$$\Phi_{j,INIT}^P = \Omega_j^p \times \Delta\phi_{INC} + \phi_j^{m0} \tag{1}$$

where ϕ_j^{m0} represents the task-urgency due to any initial maintenance work-load of j. Now if no robot attends to serve a machine, each time-step a constant maintenance workload of α_j^m will be added to j and that will increase its task-urgency by $\Delta\phi_{INC}$. So, if k time steps passes without any production work being done, task urgency at k^{th} time-step will follow Eq. 2.

$$\Phi_{j,k}^P = \Phi_{j,INIT}^P + k \times \Delta\phi_{INC} \tag{2}$$

However, if a robot attends to a machine and does some production works from it, there would be no extra maintenance work as we have assumed that μ = 1. Rather, the task-urgency on this machine will decrease by $\Delta\phi_{DEC}$ amount. If ν_k robots work on a machine simultaneously at time-step k,

this decrease will be: $\nu_k \times \Delta\phi_{DEC}$. So in such cases, task-urgency in $(k+1)^{th}$ time-step can be represented by:

$$\Phi^P_{j,k+1} = \Phi^P_{j,k} - \nu_k \times \Delta\phi_{DEC} \quad (3)$$

At a particular machine j, once $\Phi^P_{j,k}$ reaches to zero, we can say that there is no more production work left and this time-step k can give us the *production completion time* of j, T^P_j. Average production time-steps of a shop-floor with M machines can be calculated by the following simple equation.

$$T^{PMM}_{avg} = \frac{1}{M}\sum_{j=1}^{M} T^{PMM}_j \quad (4)$$

T^P_{avg} can be compared with the minimum number of time-steps necessary to finish production works, T^P_{min}. This can only happen in an ideal case where all robots work for production without any random walking or failure. We can get T^P_{min} from the total amount of work load and maximum possible inputs from all robots. If there are J machines and N robots, each machine has Φ^P_{INIT} task-urgency, and each time-step robots can decrease N $\times$ $\Delta\phi_{DEC}$ task-urgencies, then the theoretical T^P_{min} can be found from the following Eq. 5.

$$T^P_{min} = \frac{J \times \Phi^P_{INIT}}{N \times \Delta\phi_{DEC}} \quad (5)$$

$$\zeta^P_{avg} = \frac{T^P_{avg} - T^P_{min}}{T^P_{min}} \quad (6)$$

Thus we can define ζ^P_{avg}, average production completion delay (APCD) by following Eq. 6: After a machine enters into maintenance cycle, if no robot serves a machine, the growth of task-urgency will follow Eq. 2. However, if ν_k robots are serving this machine at a particular time-step k^{th} , task-urgency at $(k+1)^{th}$ time-step can be represented by:

$$\Phi^M_{j,k+1} = \Phi^M_{j,k} - (\nu_k - \mu) \times \Delta\phi_{DEC} \quad (7)$$

By considering $\mu = 1$, Eq. 7 will reduces to Eq. 3. Here, $\Phi^M_{j,k+1}$ will correspond to the *pending maintenance work-load* of a particular machine at a given time. This happens due to the random task switching of robots with random-walking options. We can find the *average pending maintenance work-load* (APMW) per time-step per machine, χ^M_j (Eq. 8) and average PMW per machine per time-step, χ^M_{avg} (Eq. 9).

$$\chi^M_j = \frac{1}{K}\sum_{k=1}^{K} \Phi^M_{j,k} \quad (8)$$

$$\chi^M_{avg} = \frac{1}{J}\sum_{j=1}^{M} \chi^M_j \quad (9)$$

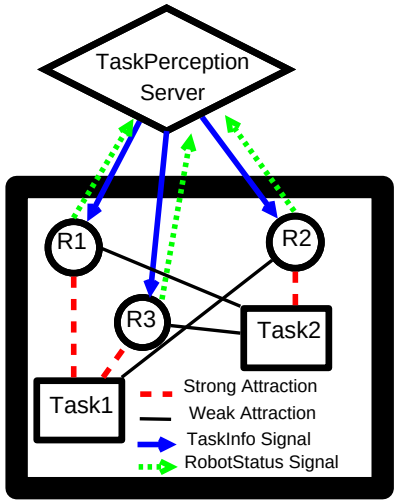


Fig. 3: A centralized communication scheme

V. Communication Models

A. Centralized communication model

AFM relies upon a system-wide continuous flow of information which can be realized using any suitable communication model. A simple centralized communication scheme is outlined in Fig. 3. In this model we have used bi-directional signal-based exchange of communication messages between a centralized *task perception server* (TPS) and a *robot-controller client* (RCC). The main role of TPS is to send up-to-date task-information to RCCs. This task-information mainly contains the location and urgency of all tasks which is used by the RCCs for running their task-allocation algorithm. The urgency value of each task is dynamically updated by TPS after receiving the status signals from the working robots of that particular task. Fig. 3 shows how three robots are attracted to two different tasks and their communications with TPS. Here although the

robots are selecting task independently based-on the strength of their attractive fields to different tasks, they are depended on the TPS for task-information. AFM suggests the communication of task-urgencies among robots. This communication helps the robots to gain information that can be treated as "global sensing". However in this model robots do not communicate among themselves. Hence this model can be approximated as the GSNC strategy.

B. Local communication model

In most swarm robotic research local communication is considered as the one of the most critical components of the swarms where the global behaviours emerges from the local interactions among the individuals and their environment. In this study, we have used the concepts of pheromone active-space of ants to realize our simple LSLC scheme. Ants use various chemical pheromones with different active spaces (or communication ranges) to communicate different messages with their group members [25]. Ants sitting near the source of this pheromone sense and respond quicker than others who wander in far distances. Thus both communication and sensing occurs within a small communication range[1]. We have used this concept of communication range or locality in our local communication model. A suitable range (or radius) of communication and sensing can be set at design time based on the capabilities of robots [15]. Alternately they can also be varied dynamically over time depending on the cost of communication and sensing, e.g. density of peers, ambient noise in the communication channels, or even by aiming for maximizing information spread [26]. In this study, we have followed the former approach as our robots do not have the precise hardware to dynamically vary their communication and sensing ranges.

In our local communication model we have assumed that robots can communicate to its nearby peers within a certain communication radius, r_{comm} and they can sense tasks within another radius r_{task}. A robot R_1 is a *peer* of robot R_2, if spatial distance between R_1 and R_2 is less than this r_{comm}. Similarly, when a robot comes within this r_{task} of a task, it can sense the status of this task. Local communication can also give robots similar task information as found in centralized communication. In this case, it is not necessary for each robot to communicate with every other robot to get information on all tasks. Since robots can random walk and explore the environment we assume that for a reasonably high robot-to-space density, all tasks will be known to all robots after an initial exploration period. In order to update the urgency of a task, robots can estimate the number of robots working on a task in two ways: by either using their sensory perception (e.g. on-board camera) or doing local P2P communication with others.

We can characterize our local communication model in terms of message content, communication frequency and target recipients [27]. Robots can communicate with their peers with any kind of message. Robots communicate only when they meet their peers within a certain communication radius (r_{comm}). In terms of target recipients, our model differs from a traditional publish/subscribe communication model by introducing the concept of dynamic subscription. In a traditional publish/-subscribe communication model, subscription of messages happens prior to the actual message transmission. In that case prior knowledge about the subjects of a system is necessary. But in our model this is not necessary as long as all robots uses a common addressing convention for naming their incoming signal channels. In this way, when a robot meets with another robot it can infer the address of this peer robot's channel name by using a shared rule. A robot is thus always listening to its own channel for receiving messages from its potential peers or message publishers. On the other side, upon recognizing a peer, a robot sends a message to this particular peer. So here neither it is necessary to create any custom subject name-space [27] nor we need to hard-code information in each robot controller about the knowledge of their potential peers *a priori*.

[1]Although, generally communication and sensing are two different issues, however within the context of our self-organized MRTA, we have broadly viewed sensing as the part of communication process, either implicitly via environment, or explicitly via local peers.

VI. Experimental design

The overall aim of our MRTA experiments is to compare the various properties of self-regulated MRTA under both GSNC and LSLC strategies. These experiments are grouped into four series labelled using the following letters: *A, B, C* and *D*. Series A and B experiments are done using a centralized communication system under GSNC strategy and Series C and D experiments are carried out using an emulated local P2P communication under LSLC stategty.

Self-regulated MRTA is often characterised by the plasticity and task-specialization, in both macroscopic and microscopic levels [**?**]. Within our manufacturing shop-floor context, plasticity refers to the collective ability of the robots to switch from random-walking to doing a task (or vice-versa) depending on the work-load present in the system. Here we expect to see that most of the robots would be able to engage in tasks when there would be high workloads (or task-urgencies). The changes of task-urgencies and the ratio of robots engaged in tasks can be good metrics to observe plasticity in MRTA. Self-regulated MRTA is generally accompanied with task-specializations of agents. That means that few robots will be more active than others. From the interpretation of AFM, we can see that after doing a task a few times, a robot will soon be sensitized to it. Therefore, from the raw log of task sensitization of robots, we can be able to find the pattern of task-sensitization of robots per task basis.

We can measure the quality of MRTA from the shop-floor production completion time i.e. APCD that indicates how much more time is spent in the production process due to the self-organization of robots. In order to calculate APCD, we can find the production completion time for each task from the raw log of task-urgency. In order to see if our system can respond to the gradually increasing workloads, we can measure shop-floor maintenance delay i.e. APMW that can show the robustness of our system. When a task is not being served by any robot for some time we can see that its urgency will rise and robots will respond to this dynamic demand. For measuring APMW we need only the task-urgency data. From the design of AFM, we know that robots that are not doing a task will be de-sensitized to it or forget that task. So at an overall low work-load (or task urgency), less robots will do the tasks and hence less robots will have the opportunity to learn tasks. From the shop-floor work-load data, we can confirm the presence of flexibility in MRTA. In order to characterize the energy-efficiency in MRTA we can log the pose data of each robot that can give us the total translations occurred by all robots in our experiments. This can give us a rough indication of energy-usage by our robots. Since AFM requires a system-wide continuous flow of information, we can measure the communication load to bench-mark our implementation of communication system. Here we can measure how much task-related information, i.e. task-urgency, location etc. are sent to the robots at each time step.

TABLE I: Experimental parameters of Series A & B

Parameter	A | B
Total number of robots (N)	8 | 16
Total number of tasks (M)	2 | 4
Experiment area (A) m^2	2 | 4
Initial production work (Ω_j^p) unit	100
Task urgency increase rate ($\Delta\phi_{INC}$)	0.005
Task urgency decrease rate ($\Delta\phi_{DEC}$)	0.0025
Initial sensitization (K_{INIT})	0.1
Sensitization increase rate (Δk_{INC})	0.03
Sensitization decrease rate (Δk_{DEC})	0.01

In order to see the effects of scaling on MRTA, we have designed two group of experiments. Series A corresponds to a small group where we have used 8 robots, 2 tasks under an arena of 2 m^2. We have doubled these numbers in Series B, C and D, i.e. 16 robots, 4 tasks under an arena of 4 m^2. This proportional design can give us a valuable insight about the effects of scaling on self-regulated MRTA. Table I lists a set of essential parameters of our GSNC strategy based experiments (Series A and B). The initial values of task urgencies correspond to 100 units of production work-load without any maintenance work-load as outlined in Eq. 1. We choose a limit of 0 and 1, where 0 means no urgency and 1 means maximum urgency. Same rule applies to sensitisation, where 0 means no sensitisation and

1 means maximum sensitisation. This also implies that if sensitization is 0, task has been forgotten completely. On the other hand, if sensitization is 1, the task has been learnt completely.

Our LSLC startegy based experiments (Series C and D) follow the similar design of experimental parameters and observables of Series B experiments as outlined above. For the sake of simplicity, the communication and sensing range values are kept same in both series of experiments. Series D uses 1m ranges that enables our robot to capture more task information at a time. That is similar to "global sensing and global communication" of information by the robots, since in a large group of robots, it is not often practical that a robot communicates with all other robots. On the other hand, Series C uses the half of the range values of Series D (0.5m). This enables our robots to capture less information, that mimics a close LSLC strategy.

VII. Implementation

Ideally, AFM can be implemented as a complete distributed task-allocation system where each agent selects its own task based on its own external perception about task-urgencies (i.e. attractive fields), distances from tasks and internal task-sensitisation records. Such an implementation requires powerful robots with sophisticated sensors (camera, laser etc.) and sufficient computation and communication capabilities. In that case, robots can keep task-urgency information up-to-date through suitable local communication schemes with their peers who can monitor the tasks. By using suitable navigation and mapping modules, they can also accurately calculate the distances from tasks and navigate to tasks autonomously. Moreover, they also require necessary hardware to do the actual task, e.g. gripper for picking up objects. However, in this study, we are particularly interested to find the suitable communication schemes that can effectively spread the attractive fields (task-urgencies) among robots. So we have simplified the complexities of a full-fledged implementation by using a centralized communication system that effectively makes up the limitations of our e-puck robots. For example, our robots are not capable of sensing a task to estimate its urgencies, instead our centralized TPS broadcast task information, e.g. task-urgencies, locations etc. to robots in certain time intervals. In our current implementation, instead of doing any real work with powerful robots, we emulate a mock manufacturing shop-floor scenario that requires the robot only to travel among tasks. As our robots do not have on-board CPU, they need a host PC to control them using BTCom Bluetooth communication protocol over the wireless link [28]. Thus our host-PC runs one RCC for each physical robot. These RCCs also rely upon SwisTrack multi-robot tracking system for updating their real-time pose. So although our MRTA solution is distributed by design, we primarily used a centralized approach to implement it due to the limitations of our robots and the convenience of implementation. Table **??** list the D-Bus communication interfaces among our software components. Fig. **??** outlines the placements of various software components of our multi-robot control architecture based on their functional characteristics and processing requirements [29].

1) RCC Modules: As shown in Fig. 4, each e-puck robot is controlled by a corresponding RCC. A RCC sends commands to the robot's firmware using BTCom protocol. A RCC consists of several Python modules. Each module represents a sub-process under a main process that ties all of them together by using Python's Multiprocessing module[2]. Below we have described briefly the implementation of these Python modules. All code are publicly accessible through GitHub repository[3]. In the implementation of RCC, we have employed a data and event management process called `DataManager` that acts as a data warehouse and event management centre for RCC. Table **??** list the major data structures and corresponding event channels of `DataManager`. Here `mRobotID` represents the e-puck robot's marker ID (converted to decimal number from the binary code) which uniquely identifies a robot from others. We have used Python *dictionary* type data structure for all other data structures that are managed by the Python Multiprocessing's

[2]http://docs.python.org/library/multiprocessing.html

[3]git://@github.com/roboshepherd/EpuckCentralizedClient.git, *hash:*7e3af8902e64db3fa59e

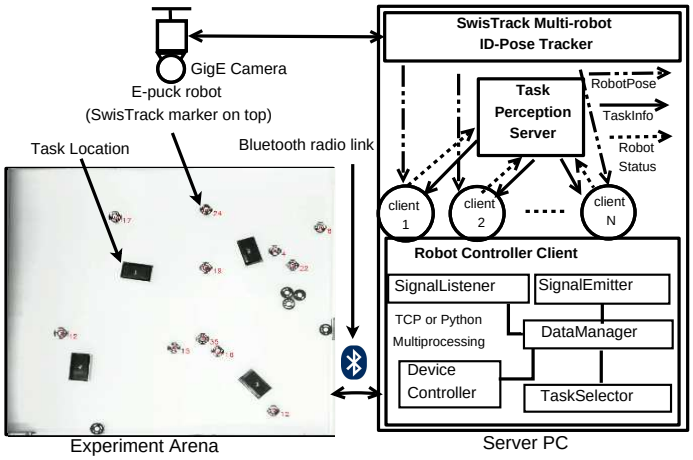


Fig. 4: Hardware and software setup for series A & B experiments

Manager() object. `DataManager` object also runs a tiny TCP/IP server process powered by the Multiprocessing's *RemoteManager()* interface. So, if necessary, any other process can access all of the DataManger's data structures and event channels remotely by instantiating a proxy client that connects to this TCP/IP server process.

We have employed two D-Bus communication modules in RCC: `SignalListener` and `SignalEmitter`, that are responsible for listening and emitting D-Bus signals respectively. `SignalListener`, has subscribed to D-Bus session bus for listening to D-Bus `RobotPose` and `TaskInfo` signals. `SignalEmitter` makes use of Python *sched* module that enables it to check periodically `mSelectedTaskStarted` event and emit a robot's task status containing signal, `RobotStatus`. `TaskSelector` works in the application layer of our multi-robot control architecture. It plugs the AFM algorithms (Sec. **??**) into RCC to select task based-on `DataManger`'s event notifications.

In order to realize LPCM strategy, in the implementation of RCC we have added three new D-Bus signal interfaces. They are briefly specified in Table **??**. Our base implementation of RCC's task-allocation (i.e. `TaskSelector`) is almost unchanged in this local implementation. The only thing we need to extend is the D-Bus signal reception and emission interfaces that catches the above signals and put the signal payload in `DataManager`. For the sake of simplicity, we have merges the perceived `TaskInfo` with communicated `LocalTaskInfo` into one so that `TaskAllocator` always gets all available task information. We have not made any major change in other modules of RCC, i.e. `DeviceController`. The full Python implementation of this RCC is available for download from the GitHub repository[4].

2) TPS modules: In our GSNC strategy based implementation of TPS periodically broadcasts `TaskInfo` signals to all robots using similar D-Bus signal emitting and listening modules. Task information has been updated by a separate Python module following the TPS algorithm described before. In the local version, TPS only emits `TaskInfo` signal when a robot is within a task's perception range, r_{task} based-on the `TaskNeighbour` signal from SwisTrack. For this additional interface, we have extended D-Bus `SignalListener` to catch this additional D-Bus signal. No major changes are done in the other parts of TPS implementation. The full Python implementation of TPS is available for download from

[4]http://github.com/roboshepherd/EpuckDistributedClient

the authors' GitHub centralized[5] and distribued [6] repositories.

VIII. Results

In this section we have presented our experimental results. We ran those experiments for about 40 minutes duration. They are averaged over five iterations for Series A and B experiments and three iterations for Series C and D experiments.

A. Shop-floor work-load history

In our experiments we have defined shop-floor work-load in terms of task urgencies. For example, Eq. 1 shows how we have calculated initial production work-load of our manufacturing shop-floor scenario. Fig. **??** show the dynamic changes in task-urgencies for the single iteration of each experiment. The fluctuations in these plots are resulted from the different levels of task-performance of our robots. In case of Series D, we can see that an unattended task, *Task4*, was not served by any robot for a long period and later it was picked up by some of the robots. In order to measure the task-related work-loads on our system we have summed up the changes in all task-urgencies over time. We call this as *shop-floor work-load history* and formalized as follows. Let $\phi_{j,q}$ be the urgency of a task j at q^{th} step and $\phi_{j,q+1}$ be the task urgency of $(q+1)^{th}$ step. We can calculate the sum of changes in urgencies of all M tasks at $(q+1)^{th}$ step:

$$\Delta\Phi_{j,q+1} = \sum_{j=1}^{M} (\phi_{j,q+1} - \phi_{j,q}) \qquad (10)$$

Fig. **??** shows the dynamic shop-floor workload for all four series of experiments. From these plots, we can see that initially the sum of changes of task urgencies (shop-floor workload) is going towards negative direction. This implies that tasks are being served by a high number of robots.

B. Ratio of active workers

From Fig. **??**, we can see that in production stage, when work-load is high, many robots are active in tasks. Here active workers ratio is the ratio of those robots that work on tasks to the total number of robots N of a particular experiment. Here we can see that this ratio varies according to the shop-floor work-load changes.

C. Shop-task performance

In our manufacturing shop-floor scenario, we have calculated the APCD and APMW as shown in Fig. **??**. For Series A we have got average production completion time 111 time-steps (555s) where sample size is (5 x 2) = 10 tasks, SD = 10 time-steps (50s). According to Eq. 5, our theoretical minimum production completion time is 50 time-steps (250s) assuming the non-stop task performance of all 8 robots with an initial task urgency of 0.5 for 2 tasks and task urgency decrease rate $\Delta\Phi_{DEC}$ = 0.0025 per robot per time-step. Hence, Eq. 6 gives us APCD, ζ = 1.22 which means that in Series A experiments, it took 1.22 times more time (305s) than the estimated minimum production completion time (250s). For Series B, we have got average production completion time 165 time-steps (825s) where sample size is (5 x 4) = 20 tasks, SD = 72 time-steps (360s). Hence, Eq. 6 gives us APCD, ζ = 2.3. For Series C we have got average production completion time 121 time-steps (605s) with SD = 36 time-steps (180s). For Series D, average production completion time is 123 time-steps (615s) with SD = 40 time-steps (200s). According to Eq. 5, our theoretical minimum production completion time is 50 time-steps (250s). The values of APCD are as follows. For Series C, ζ = 1.42 and for Series D, ζ = 1.46. For both series of experiments APCD values are very close.

For APMW, Series A experiments give us an average time length of 369 time-steps (1845s). In this period we have calculated APMW and it is 1 time-step with SD = 1 time-step (5s) and $\Delta\Phi_{INC}$ = 0.005 per task per time-step. This shows a very low APMW (χ = 0.000235) implying a very high robustness of our system. For Series B experiments, from the average 315 time-steps (1575s) maintenance activity of our robots per experiment run, we have got APMW, χ = 0.012756 which corresponds to the pending work of 3 time-steps (15s) where

[5]http://github.com/roboshepherd/CentralizedTaskServer
[6]http://github.com/roboshepherd/DitributedTaskServer

SD = 13 time-steps (65s). This also tells us the robust task performance of our robots which can return to an abandoned task within a minute or so. The APMW of Series C experiments give us an average time length of 359 time-steps (1795s). In this period we calculated APMW and it is 5 time-steps with SD = 17 time-steps and χ = 0.023420. For Series D experiments, from the average 357 time-steps (1575s) of maintenance activity of our robots per experiment run, we have got APMW, χ = 0.005359 which corresponds to the pending work of 2 time-steps (10s) where SD = 7 time-steps.

D. Task specializations

We have measured the task-specialization of the robots based-on their peak value of sensitization. This maximum value represents how long a robot has repeatedly been selecting a particular task. Since tasks are homogeneous we have considered the maximum sensitization value of a robot among all tasks during an experiment run. This value is then averaged for all robots using the following equation.

$$K^G_{avg} = \frac{1}{N}\sum_{i=1}^{N} \max_{j=1}^{M} \left(k^i_{j,q}\right) \tag{11}$$

If a robot r_i has the peak sensitization value k^i_j on task j ($j \in M$) at q^{th} time-step, Eq. 11 calculates the average of the peak task-specialization values of all robots for a certain iteration of our experiments. We have also averaged the time-step values (q) taken to reach those peak values for all robots using the following equation.

$$Q^G_{avg} = \frac{1}{N}\sum_{i=1}^{N} q^i_{k=k_{max}} \tag{12}$$

In Eq. 12, $q^i_{k=k_{max}}$ represents the time-step of robot r_i where its sensitization value k reaches the peak k_{max} as discussed above. By averaging this peak time-step values of all robots we can have an overall idea of how many task-execution cycles are spent to reach the maximum task-specialization value K^G_{avg}. Table **??** and Table **??** show the peak sensitization values of Series A and Series B experiments respectively. Based on Eq. 11 and Eq. 12, we have got the peak task-sensitization K^G_{avg} values: 0.40 (SD=0.08) and 0.30 (SD=0.03), and their respective time-step Q^G_{avg} values: 38 (SD=13) and 18 (SD=5) time-step. They are shown in Fig. **??**. Here we can see that the robots in Series A had higher chances of task-specialization than that of Series B experiments.

E. Robot motions

We have aggregated the changes in translation motion of all robots over time based on their travelled distances in Euclidean metric. Let $u_{i,q}$ and $u_{i,q+1}$ be the translations of a robot i in two consecutive steps. If the difference between these two translations be δu_i, we can find the sum of changes of translations of all robots in $(q+1)^{th}$ step using the following equation.

$$\Delta U_{q+1} = \sum_{i=1}^{N} \delta u_{i,q+1} \tag{13}$$

The sum of translations of robots from our experiments are plotted in Fig. **??**. In this plot we can see that robot translations also vary over varying task requirements of tasks.

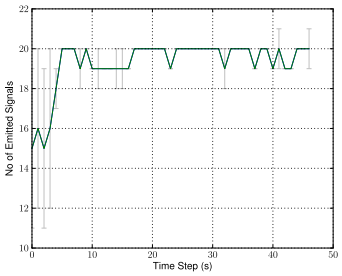
F. Communication load

Fig. 9 (a) and (b) shows the number of received `TaskInfo` signals by each robot in Series A and Series B experiments. Since the duration of each time-step is 50s long and TPS emits signal in every 2.5s, there is an average of 20 signals in each time-step. The overall P2P task information signals of both of these local modes are plotted in Fig. 9 (c) and (d). As an example of P2P signal reception of a robot, Fig. **??** show the number of received signals by *Robot12* in two local experiments. It states the relative difference of peers over time in two different cases.

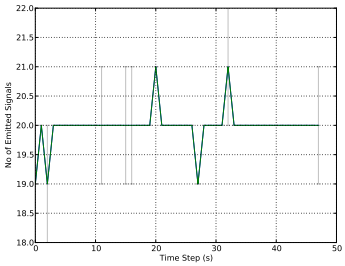
IX. Discussions

A. Self-regulated MRTA

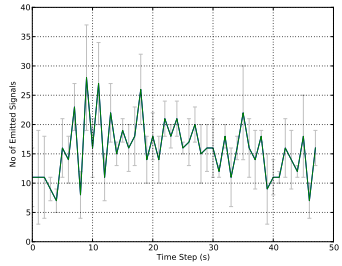
From our experimental results, we have noted several aspects of self-regulated MRTA that expose the effectiveness of AFM. As we have pointed



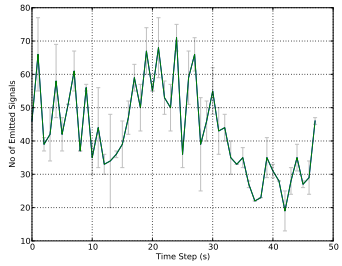
(a) Series A



(b) Series B



(c) Series C



(d) Series D

Fig. 9: Frequency of `TaskInfo` signalling of (a) and (b) by TPS and (c) and (d) by local peers.

out that this self-regulated MRTA, as observed in biological and human social systems, needs to satisfy several important characteristics, particularly plasticity and task-specialization. In addition to satisfying those basic qualities, AFM has demonstrated many other aspects. Our self-regulated robots, driven by AFM, effectively handle the dynamic work-load in our manufacturing shop-floor. They can dynamically support the need to work on demanding tasks, if there any. The variations of active worker ratio shows this.

From the self-organized worker allocations of AFM, it is clear to us that although in larger system (Series B) the degree of variations of active-worker ratio can show us significantly unpredictable patterns, nevertheless the self-regulated rules drive the robots to respond to the dynamic needs of the system. This means that AFM can sufficiently produce the plasticity of DOL in order to meet the dynamic work-load of the system.

B. Learning and forgetting

From the individual and group-level task-specialization, we can see that robots can maintain both task-specialization and flexibility. In a self-organized system, it is very common that only a few individuals specialize on tasks and others generally do not. From two sample data sets provided in Table **??** and Table **??**, we can see that in particular runs of Series A and Series B experiments, task-sensitization values of only 2-3 robots reach above the group-level average score. Thus in both types of experiments, robots exhibit similar task-specialization behaviours.

C. Concurrency and robustness

As a consequence of fewer robots specializing in tasks, we can also see that robots can concurrently consider different tasks without being biased to a particular task all the time. Our experiments also show us the robust MRTA as in case of both high and low work-loads present in the system. This is evident from the manufacturing shop-floor task performance during PMM and MOM. For example, in case of Series B experiments APMW was 13 time-steps (65s) which corresponded to pending work-load of 0.065 unit for a single robot. Thus, on an average, before the work-load exceeded by about 13 percent of initial work-load, robots were able to respond to a task.

D. Communication load

In GSNC strategy based experiments we used a centralized communication system (source of attractive fields) that serves the robots with necessary task-perception information. Our centralized communication system has the advantage of minimising the communication load and the disadvantage of a single point of failure as well as a single point of load. Under LSLC strategy based implementation, we present how the task perception can be decentralized by P2P communications among RCCs.

E. Scaling-up

We have observed the effect of scaling-up the robot team size. The system size of Series B is double of that of Series A in terms of robots, tasks and experiment arena. Keeping a fixed ratio of robot-to-task and task-to-arena we have intended to see the scaling effects in our experiments. Here we see that both systems can show sufficient self-regulated DOL, but task-performance of both systems varies significantly. For example, the value of APCD in Series B is higher by 1.08. This means that performance is decreased in Series B experiments despite having the resources in same proportion in both systems. This occurs partly due to the greater stochastic effects found in task-allocation in a larger system, e.g. presence of more tasks produce higher stochastic behaviours in robot's task selection.

Similarly we can see that in larger system robots have less chances to specialize on tasks, as the Series B experiments show us that the overall average task-specialization of the group K^G_{avg} is lower by 0.10 and it lasts for significantly less time (the difference of Q^G_{avg} of both systems is 20 time-steps). Thus, in a large group, robots are more likely to switch among tasks more frequently and this produces more translation motions which cost more energy (e.g. battery power) in task-performance.

F. Comparisons between GSNC and LSLC strategies

Results from Series C and Series D experiments show many similarities and differences with respect to the results of Series A and Series B experiments. As shown in Table II both Series C and Series D experiments show similar production delay (APCD) values: 605s and 615s respectively, which are less than Series B experiment result (825s) and are close to Series A experiment result (555s). Note that all statistical t-test results are obtained with respect to Series C data and it shows that there is no significant difference in task-performance between Series B and Series C results. Besides, in terms of task-specialization as shown in Table III, the overall task-specialization of group in Series C (K^G_{avg} = 0.4) is closer to that of Series A experiments (K^G_{avg} = 0.39) and interestingly, the value of Series D (K^G_{avg} = 0.27) is much closer to that of Series B experiments (K^G_{avg} = 0.30). So task-specialization in large group under LSLC strategy is not significantly different than their GSNC counter part.

Table IV summarizes the average translations done by robots in all four series of experiments. From these robot motion profiles, it is found that under LSLC strategy, robot translations have been reduced significantly. From this table it is seen than Series C and Series D show about 2.8 times less translation than that in Series B experiments. The translation of 16 robots in Series C and Series D experiments are approximately double (1.89 times) than that of Series A experiments with 8 robots. Thus the energy-efficiency under LSLC strategy seems to be higher than that under GSNC strategy.

From the above results one can see that large group robots achieve similar or better MRTA under LSLC strategy. The local sensing of tasks prevents them to attend a far-reaching task which may be more common under global sensing strategy. However, as it is seen in Fig. **??** some tasks can be left unattended for a long period of time due to the failure to discover it by any robot. For that reason it is seen that the values of pending maintenance load is slightly higher under LSLC strategy. But this trade-off is worth as LSLC strategy provides superior self-regulated MRTA in terms of task-performance, task-specialization and energy-efficiency.

X. Conclusion and future works

The benefits of deploying a large number of robots in dynamic multi-tasking environment can not be fully realized without adopting an effective communication and sensing strategy for a given multi-robot system. This study has focused on comparing two bio-inspired communication and sensing strategies in producing self-regulated MRTA by an interdisciplinary model of DOL, AFM. Under the GSNC strategy, AFM has produced the desired self-regulated MRTA among a group of 8 and 16 robots. This gives us the evidence that AFM can successfully solve the MRTA issue of a complex multi-tasking environment like a manufacturing shop-floor. Under the LSLC strategy, AFM can also produce the desired self-regulated MRTA for 16 robots with different communication and sensing ranges.

From our comparative results, we note that for large group of robots, degradation in task-performance and task-specialization of robots are likely to occur under GSNC strategy that relies upon a centralized communication system. Thus GSNC strategy can give us better performance when the number of tasks and robots are relatively small. This confirms us the assertions made by some biologists that self-regulated DOL among small group of individuals can happen without any significant amount of local communications and interactions. However, our findings suggest that task-specialization can still be beneficial among the individuals of a small group which contradicts the claim that small groups only posses the generalist workers, but not the specialists.

On the other hand, LSLC strategy is more suitable for large group of individuals that are likely to be unable to perform global sensing and global communications with all individuals of the group. The design of communication and sensing range has still remained as a critical research issue. However, our results suggest that the idea of maximizing information gain is not appropriate under a stochastic task-allocation process. Here more information tends to cause more task-switching behaviours that

lowers the level of task-specialization of the group. This might not be the case under a deterministic task-allocation scheme where more information may lead to better and optimum allocations in a small group of individuals. Nevertheless, despite having the limited communication and sensing range, LSLC strategy has helped the robots to produce comparatively better task-allocation with increased task-specialization and significantly reduced motions or savings in energy consumption.

REFERENCES

[1] L. E. Parker and F. Tang, "Building multirobot coalitions through automated task solution synthesis," *Proceedings of the IEEE*, vol. 94, pp. 1289–1305, 2006.

[2] B. P. Gerkey and M. J. Mataric, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 23, p. 939, 2004.

[3] A. B. Sendova-Franks and N. R. Franks, "Self-assembly, self-organization and division of labour'," *Philosophical Transactions of the Royal Society of London B*, vol. 354, pp. 1395–1405, 1999.

[4] B. Gerkey and M. Mataric, "Multi-robot task allocation: analyzing the complexity and optimality of key architectures," *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 3, 2003.

[5] E. Arcaute, K. Christensen, A. Sendova-Franks, T. Dahl, A. Espinosa, and H. J. Jensen, "Division of labour in ant colonies in terms of attractive fields," *Ecol. Complexity*, 2008.

[6] R. Jeanne, "Group size, productivity, and information flow in social wasps," *Information processing in social insects*, pp. 3–30, 1999.

[7] L. E. Parker, "Distributed intelligence: Overview of the field and its application in multi-robot systems," *Journal of Physical Agents, special issue on multi-robot systems*, vol. vol. 2, no. no. 2, pp. 5–14, 2008.

[8] L. Chaimowicz, M. F. M. Campos, and V. Kumar, "Dynamic role assignment for cooperative robots," *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 1, 2002.

[9] M. B. Dias, R. M. Zlot, N. Kalra, and A. Stentz, "Market-based multirobot coordination: A survey and analysis," *Proceedings of the IEEE*, vol. 94, pp. 1257–1270, 2006.

[10] K. Lerman, C. Jones, A. Galstyan, and M. J. Mataric, "Analysis of dynamic task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 25, p. 225, 2006.

[11] S. Camazine, N. Franks, J. Sneyd, E. Bonabeau, J. Deneubourg, and G. Theraula, *Self-organization in biological systems*. Princeton, N.J. : Princeton University Press, c2001., 2001, what is self-organization? – How self-organization works – Characteristics of self-organizing systems – Alternatives to self-organization –.

[12] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm intelligence: from natural to artificial systems*. Oxford University Press, 1999.

[13] W. Liu, A. F. T. Winfield, J. Sa, J. Chen, and L. Dou, "Towards energy optimization: Emergent task allocation in a swarm of foraging robots," *Adaptive Behavior*, vol. 15, no. 3, pp. 289–305, 2007.

[14] M. Krieger and J. Billeter, "The call of duty: Self-organised task allocation in a population of up to twelve mobile robots," *Robotics and Autonomous Systems*, vol. 30, no. 1-2, pp. 65–84, 2000.

[15] W. Agassounon and A. Martinoli, "Efficiency and robustness of threshold-based distributed allocation algorithms in multi-agent systems," in *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 3*. ACM, 2002, p. 1097.

[16] C. Jones and M. Mataric, "Adaptive division of labor in large-scale minimalist multi-robot systems," in *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings*, 2003.

[17] N. Kalra and A. Martinoli, "A comparative study of market-based and threshold-based task allocation," *Distributed Autonomous Robotic Systems 7*, pp. 91–101, 2007.

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

[19] B. B. Werger and M. J. Mataric, "Broadcast of local eligibility for multi-target observation," *Distributed Autonomous Robotic Systems*, vol. 4, p. 347356, 2001.

[20] S. Botelho, R. Alami, and T. LAAS-CNRS, "M+: a scheme for multi-robot cooperation through negotiated taskallocation and achievement," *Proceedings IEEE International Conference on Robotics and Automation*, vol. 2, 1999.

[21] B. P. Gerkey and M. J. Mataric, "Sold!: Auction methods for multirobot coordination," *IEEE Transaction on Robotics and Automation*, vol. 18, 2002.

[22] R. Zlot, A. Stentz, M. Dias, and S. Thayer, "Multi-robot exploration controlled by a market economy," *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 3, 2002.

[23] H. Çelikkanat, A. Turgut, and E. Şahin, "Guiding a robot flock via informed robots," *Distributed Autonomous Robotic Systems 8*, pp. 215–225, 2008.

[24] M. Sarker and T. Dahl, "A Robotic Validation of the Attractive Field Model: An Inter-disciplinary Model of Self-regulatory Social Systems," in *Swarm Intelligence: Proc. of 7th International Conference, ANTS 2010, Brussels, Belgium, September 8-10, 2010*. Springer-Verlag GmbH, 2010, p. 24.

[25] B. Holldobler and E. Wilson, *The ants*. Belknap Press, 1990.

[26] E. Yoshida and T. Arai, "Performance analysis of local communication by cooperating mobile robots," *IEICE Transactions on Communications*, vol. 83, no. 5, pp. 1048–1059, 2000.

[27] B. Gerkey and M. Mataric, "Principled communication for dynamic multi-robot task allocation," *Experimental Robotics VII*, pp. 353–362, 2001.

[28] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J. Zufferey, D. Floreano, and A. Martinoli, "The e-puck, a robot designed for education

in engineering," in *Proc. of the 9th Conf. on Autonomous Robot Systems and Competitions*, 2009.

[29] M. Sarker and T. Dahl, "Flexible Communication in Multi-robotic Control System Using HEAD: Hybrid Event-driven Architecture on D-Bus," in *In Proc. of the UKACC International Conference on Control 2010 (CONTROL 2010), to appear*, 2010.

TABLE II: Shop-floor production and maintenance task performance

Experiment Series	*Production delay (SD) s*	*p-value* 1-tailed t-test (confidence)	*Pending maintenance time (SD) s*	*p-value* 1-tailed t-test
A	555 (50)	0.0	5 (5)	0.0
B	825 (360)	0.2 (60%)	15 (65)	0.0
C	605 (180)	N/A	25 (85)	N/A
D	615 (200)	0.0	10 (35)	0.0

TABLE III: Task-specialization values of the robots

Experiment Series	K^G_{avg} (SD)	*p-value* 1-tailed t-test (confidence)	Q^G_{avg} (SD)	*p-value* 1-tailed t-test (confidence)
A	0.40 (0.08)	0.0	38 (13)	0.001 (99.8%)
B	0.30 (0.03)	0.2 (60%)	18 (5)	0.2 (60%)
C	0.39 (0.17)	N/A	13 (7)	N/A
D	0.27 (0.1)	0.0	11 (5)	0.0

TABLE IV: Sum of translations of robots in all experiments.

Experiment Series	Average translation (SD) m	*p-value* 1-tailed t-test (confidence)
A	2.631 (0.804)	N/A
B	13.882 (3.099)	0.001 (99.8%)
C	4.907 (1.678)	N/A
D	4.854 (1.592)	0.0