Self-regulated Multi-robot Task Allocation: A Taxonomy and Comparison of Centralized and Local Communication Strategies

Md Omar Faruque Sarker and Torbjørn S. Dahl[a]

[a]*Cognitive Robotics Research Centre*
Newport Business School, Allt-yr-yn Campus
Newport, NP20 5DA, United Kingdom.
Mdomarfaruque.Sarker |Torbjorn.Dahl@newport.ac.uk

Abstract

This paper proposes to solve the MRTA problem using a set of previously published generic rules for 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), provides sufficient abstraction to local communication and sensing which is uncommon in existing MRTA solutions. We have validated the effectiveness of AFM to address 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 under a peer-to-peer local communication scheme. They are applied in a manufacturing shop-floor scenario using 16 e-puck robots. A flexible multi-robot control architecture, *hybrid event-driven architecture on D-Bus*, has been outlined which uses the state-of-the-art D-Bus interprocess communication. Based-on the organization of task-allocation, communication and interaction among robots, a novel taxonomy of MRTA solutions has been proposed to remove the ambiguities found in existing MRTA solutions. Besides, a set of domain-independent metrics, e.g., plasticity, task-specialization and energy usage, has been formalized to compare the performances of the above two strategies.

1. Introduction

Multi-robot systems can provide improved performance, fault-tolerance and robustness in complex and distributed tasks through parallelism and redundancy [1, 2]. However in order to get potential benefits of multi-robot systems, we need to answer a common research question. *How can we allocate tasks among multiple robots* *dynamically?* Traditionally, this issue has been identified as the *multi-robot task allocation* (MRTA) [3]. This issue can be treated as the *division of labour* (DOL) among robots, analogous to the DOL in biological and human social systems[1] [4].

MRTA is generally identified as the question of assigning tasks in an appropriate time to the appropriate robots considering the changes of the environment and/or the performance of other team members [5]. This is a *NP-hard* optimal assignment problem where optimum solutions can not be found quickly for large and complex problems [6].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, generally robots have limited capabilities to sense, to communicate and to interact locally. None of them has the complete knowledge of the past, present or future actions of other robots.

Early research on predefined task-allocation was dominated by intentional coordination [6], use of dynamic role assignment [7] and market-based bidding approach [8]. Under these approaches, robots use direct task-allocation method, often to communicate with group members for negotiating on tasks. These approaches are intuitive, comparatively straight forward to design and implement and can be analysed formally. However, these approaches typically works well only when the number of robots are small (≤ 10) [9].

On the other hand, self-organized task-allocation approach relies on the emergent group behaviours, such as emergent cooperation [10], adaptation rules [11] etc. They are more robust and scalable to large team sizes. However, most of the robotic researchers found that self-organized task-allocation approach is difficult to design, to analyse (formally) and to implement in real robots. The solutions from these systems are also sub-optimal. It is also difficult to predict exact behaviours of robots and overall system performance.

Within the context of the Engineering and Physical Sciences Research Council (EPSRC) project, "Defying the Rules: How Self-regulatory Systems Work", we have proposed to solve the above mentioned self-regulated DOL problem in an alternate way [12]. Our approach is inspired from the studies of emergence of task-allocation in both biological and human social systems. We have proposed four generic rules to explain self-regulation in those social systems. These four rules are: *continuous flow of information*, *concurrency*, *learning* and *forgetting*, all of them will be explained later. Primarily these rules deal with the issue of deriving local control laws for

[1]Although the term "division of labour" is often used in biological literature and the term "task-allocation" is primarily used in multi-agent literature, in this paper we have used these terms interchangeably.

regulating an individual's task-allocation behaviour that can facilitate the DOL in the entire group. In order to employ these rules in the individual level, we have developed a formal model of self-regulated DOL, called the *attractive field model* (AFM).

In biological social systems, communications among the group members, as well as sensing the task-in-progress, are two key components of self-organized DOL. In robotics, existing self-organized task-allocation methods rely heavily upon local sensing and local communication of individuals for achieving self-organized task-allocation. However, AFM differs significantly in this point by avoiding the strong dependence on the local communications and interactions found in many existing approaches to MRTA. AFM provides a rich abstraction to this requirement through a system-wide continuous flow of information about tasks, agent states etc. This flow of information can be achieved by using both centralized and decentralized communication modes under explicit and implicit communication strategies.

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* [13]. Depending on the group size, these species follow different strategies for communication and sensing of tasks. Polistes wasps are called the *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. On the other hand, polybia wasps are called the *swarm founders* where a swarm of workers and queens initiate colonies consisting of several hundreds to millions of individuals.

The most notable difference in the organization of work of these two social wasps is: 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. On the other hand, the work mode of swarm founders can be treated as *local sensing - local communication (LSLC)* where the individuals can only sense tasks locally due to large colony-size and they can communicate locally to exchange information, e.g. task-requirements (although their exact mechanism is unknown).In this study, we have used these two sensing and communication strategies to compare the performance of the self-regulated DOL of our robots under AFM.

The main contributions of this paper are as follows:

- Interpretation of AFM, an inter-disciplinary generic model of division of labour, as a basic mechanism of self-regulated MRTA.

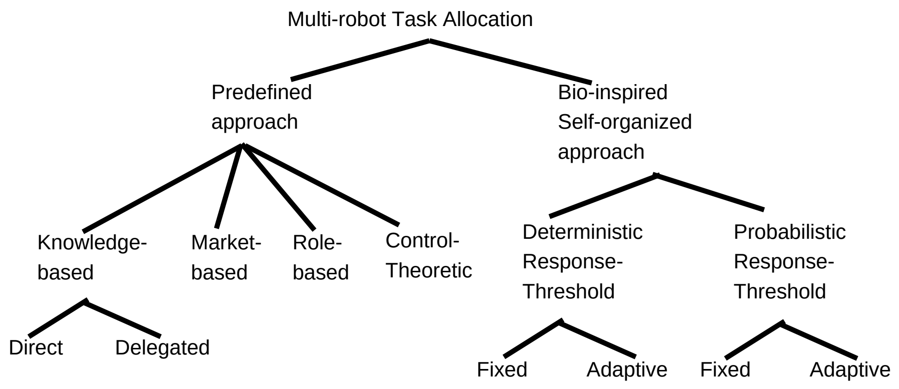
Multi-robot Task Allocation
Predefined approach
Bio-inspired Self-organized approach
Knowledge-based
Market-based
Role-based
Control-Theoretic
Deterministic Response-Threshold
Probabilistic Response-Threshold
Direct
Delegated
Fixed
Adaptive
Fixed
Adaptive

Figure 1: Classification of MRTA solutions.

- Validation of the model through experiments with reasonably large number of real robots i.e., 16 e-puck robots.
- Comparisons of the performances of two bio-inspired sensing and communication strategies in achieving self-regulated MRTA.
- Development of a flexible multi-robot control architecture using D-Bus interprocess communication technology.
- Classification of MRTA solutions based on three major axes: organization of task-allocation, interaction and communication.

2. Related work

Since 90s, MRTA many robot control architectures have been solely designed to address MRTA issue from different perspectives. Based-on the high-level design of those solutions, here we have classified them into two major categories: 1) predefined or intentional task-allocation and 2) bio-inspired self-organized task-allocation. Fig. 1 illustrates our classification.

In most of the traditional multi-robot system, task allocation is done using well-defined models of tasks and environments. Here it is assumed that the system designer has the precise knowledge about tasks, robot-capabilities etc. Many flavours of the type of task-allocation can be found in the literature. Knowledge-based and multi-agent based approaches use various knowledge-based techniques to represent tasks, robot capabilities etc. One of the early well-known MRTA architecture of this category was ALLIANCE in which each robot models the ability of team-members to perform tasks by observing their current task performances and collecting relevant task quality statistics e.g. time to complete tasks [14]. Robots use these models to select a task that benefit the group as a whole.

Similar to ALLIANCE, multi-agent based task allocation also use both centralized and decentralized approaches for allocating tasks among its peers. [15] presented a detailed categorization where in a multi-agent system task allocation can be done by using various agents ranging from a central supervising agent or a few mediator agents to all independent agents.

As a feasible alternative to the above common multi-agent based task-allocation techniques, many researchers have been following the market-based bidding approach [8]. Originated from the Contract-Net Protocol, market-based approach can be implemented as a centralized auctioning system or as a combination of *a few auctioneers – all bidders* or, independently *all auctioneers – all bidders*. For example, in a completely distributed system, when a robot needs to perform a task for which it does not have necessary expertise or resources, it broadcasts a task-announcement message, often with a expiry time of that message. Robots, that received the message and can perform that task, return a bid message. The initiating robot or *manager* selects one (or more) bidder, called as *contractor*, and offers the opportunity to complete the task. The choice of contractor is done by the manager with a mutual agreement with contractor that maximizes the individual profits.

Under role or value-based task-allocation scheme, each role assumes several specific tasks and each robot selects roles that best suit their individual skills and capabilities [7]. In this case, robots are typically heterogeneous, each one having variety of different sensing, computation and effector capabilities. Here robot-robot or robot-environment interactions are designed as a part of the organization. In multi-robot soccer [16], positions played by different robots are often defined as roles, e.g. goalkeeper, left/right defender, left/right forwarder etc.

Under control-theoretic approaches, a model of the system is usually developed that converts the task specification into an objective function to be optimized. This model typically uses the rigid body dynamics of the robots assuming the masses and other parameters well-known. Control laws of individual robots are derived either by analytically or by run-time iterations. Unlike most other approaches where task-allocation problem is taken as discrete, control-theoretic approaches can produce continuous solutions. The formalisms of these systems allow system designer to check the system's controllability, stability and related other properties. These systems typically use some degree of centralization, e.g. choosing a leader robot. Example of control-theoretic approach include: multi-robot formation control [17], multi-robot box-pushing [18] etc.

Predefined task-allocation through few other approaches are also present in the literature. For example, inspired by the vacancy chain phenomena in nature, [19] proposed a vacancy chain scheduling algorithm for a restricted class of MRTA problems in spatially classifiable domains.

Task performance in self-organized approaches relies on the collective behaviours resulted from the local interactions of many simple and mostly homogeneous (or interchangeable) agents. Robots choose their tasks independently using the principles of self-organization, e.g. positive and negative feedback mechanisms, randomness. Moreover interaction among individuals and their environment are modulated by the stigmergic, local and broadcast communications. Among many variants of self-organized task-allocation, most common type is threshold-based task-allocation [20]. In this approach, a robot's decision to select a particular task depends largely on its perception of stimulus (demand for a task) and its corresponding response threshold for that task.

Under deterministic response-threshold approach, each robot has a fixed or deterministic activation threshold for each task that needs to be performed. It continuously perceives or monitors the stimulus of all tasks that reflect the relative urgencies of tasks. When a particular task-stimuli exceeds a predefined threshold the robot starts working on that task and gradually decreases this 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 [21, 11], aggregation [22]. This fixed response-threshold can initially be same for all robots [23], or they can be different according robot capabilities or configuration of the system [21]. Adaptive response threshold model changes or adapts the threshold over time. Response-threshold decreases often due to performance of a task and this enables a robot to select that particular task more frequently or in other words it learns about that task [20, 22].

Unlike deterministic approach, where robots always respond to a task-stimuli that has a largest stimulus above the threshold, probabilistic approach offers a selection process based-on a probability distribution. Robots always have small nonzero probabilities for all tasks.

Most predefined task-allocation solutions are proposed within the context of a known or controlled environment where the modelling of tasks, robots, environments etc. becomes feasible. Note that here tasks can be arbitrarily complex that often require relatively higher sensory and processing abilities of robots. Robot-team can be consists of homogeneous or heterogeneous individuals, having different capabilities based on the variations in their hardware, software etc. But the uncertainty of the environment is assumed to be minimum.

On the other hand, bio-inspired self-organized MRTA solutions are free from extensive modelling of environment, tasks or robot capabilities. Most of the existing research considers very simple form of one global task e.g. foraging, area cleaning, box-pushing etc. This is due to the fact that major focus of this approach is limited mainly to design individual robot controllers in such a way that a few simple or *specific* tasks can be accomplished. More research is needed to verify the capabilities of self-organized approach in doing multiple complex tasks. At this moment, the bottom line remains as "select simple robots for simple tasks (self-organized approach) and complex robots for complex tasks (predefined approach)".

Both of the above task-allocation approaches expose their relative strengths and weaknesses when they are put under real-time experiments with variable number of robots and dynamic tasks. In an arbitrary event handling domain, [24] compared between self-organized and predefined market-based task-allocation, where they found that predefined task-allocation was more efficient when the information was accurate, but threshold-based approach offered similar quality of allocation at a fraction of cost under noisy environment.

[5] presented a comparative study of the complexity and optimality of key architectures, e.g. ALLIANCE [14], BLE [25], M+ [26], MURDOCH [27], First piece auctions [28] and Dynamic role assignment [7], all of them relied upon predefined task-allocation methods. The computational and communication requirements of these MRTA solutions were expressed in terms of number of robots and tasks. Although this study does not explicitly measures the scalability of those key architectures, it clearly shows us that many predefined task-allocation solutions will fail to scale well in challenging environments when the number of robots and tasks will increase, under the given limited overall communication bandwidth and processing power of individual robots.

From above discussions we can see that, self-organized task-allocation methods are advantageous as they can provide fully distributed, scalable and robust MRTA solutions through redundancy and parallelism in task-executions. Moreover, the interaction and communication requirements of robots can also be kept under a minimum limit. So for large multi-robot system, self-organized task-allocation methods can potentially be selected, if the complex tasks can be divided into simple pieces that can be carried out by multiple simple robots in parallel with limited communication and interaction requirements.

3. A three-axes taxonomy of MRTA solutions

In order to characterize both predefined and self-organized approaches in terms of their deployment, we propose three distinct axes: 1) organization of task-allocation (X), 2) degree of interaction (Y) and 2) degree of communication (Z). Fig. 2 depicts these axes with a reference point O. These axes can be used to measure the

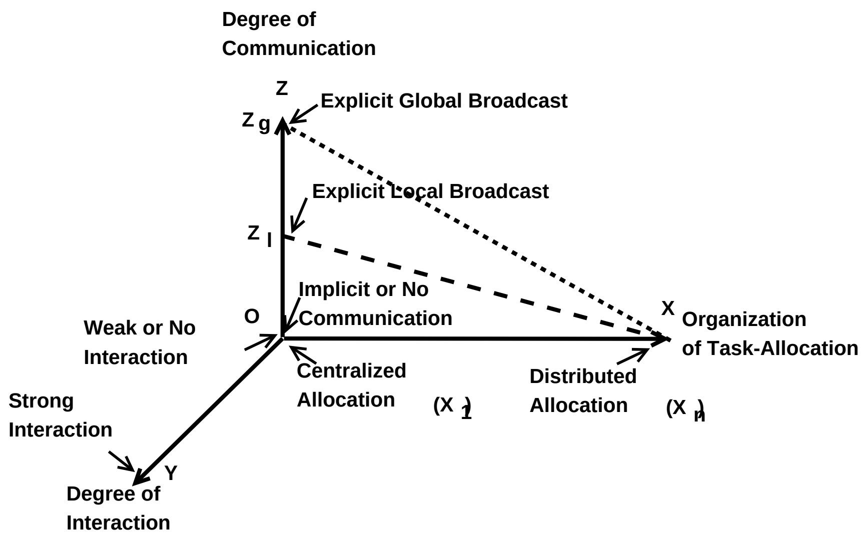

Figure 2: Three major axes of complexities in MRTA

complexities involved in various kinds of MRTA problems and the design of their solutions.

In Fig. 2, X axis represents the number of active nodes that provides the task-allocation to the group. For example, in any predefined task-allocation approach, we can use one external centralized entity or one of the robots (aka leader) to manage the task-allocation. In many predefined methods, e.g. in market-based systems, multiple nodes can act as mediators or task-allocators that we have discussed before. 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 discussed before in ALLIANCE architecture).

Most of the self-organized task-allocation methods are fully distributed, i.e. they allocate their tasks independently without the help of a centralized entity. However, they might be dependent on external entities for getting status or descriptions of tasks. Recent studies on swarm-robotic flocking by [29] show that a swarm can be guided to a target by a few informed individuals (or leaders) while maintaining the self-organizing principles of task-allocation. Task-allocation of a swarm of robots just by one central entity may be rare since one of the major spirits of swarm robotic system is to become fully distributed.

In Fig. 2, Y axis corresponds to the level of robot-robot interaction present in the system. Group-level interaction can be classified into various levels: collective, cooperative, coordinative and collaborative [6]. 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.

Y axis can also be used to refer to the degree of coupling present in the system. In case of collective interaction, robots merely co-exist, i.e. they may not be aware of each other except treating others as obstacles. Many other multi-robot systems are loosely-coupled where robots can indirectly infer some states of the environment from their team-mates' actions. But in many cases, e.g. in co-operative transport, robots not only recognize others as their team-mates, but also they coordinate their actions. Thus they form a tightly coupled system. This level of interaction and coupling also gives us the information about potential side-effects of failure of an individual robot. Tightly coupled systems with high degrees of interactions among the robots suffer from the performance loss if some of the robots removed from the system.

The Z axis of Fig. 2 represents the communication overhead of the system. This can be the result of the interactions of robots under a given task-allocation method. As we have discussed before various task-allocation methods rely upon variable degrees of robot-robot communications. On the other hand, the communication capabilities of individual robots can limit (or expand) the level of interaction 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. And in other way, the communication capabilities of robots can guide a system designer to design interaction rules of robot teams, e.g. the specification of robot's on-board camera can determine the degree of possible visual interactions among robots. The suitable trade-offs between these two axes: communication and interaction can give us a balanced design of our MRTA solutions.

The central issue of this paper is to determine the role of communication and sensing strategies under an adaptive response-threshold task-allocation method. So we have focused to examine the benefits of traversing along the various axes of Fig. 2. In this paper, we are interested on two distinct lines: 1) distributed task-allocation, with no direct robot-robot interaction and communication, say line OX_n (n being the number of robots) and 2) distributed task-allocation, with no direct robot-robot interaction, but varying degrees of local communications, say line X_nZ_l (Z_l being a local broadcast communication strategy that involves l number of peers in communication).

4. Robotic interpretation of the Attractive Field Model

The construction of AFM has been achieved through a series of collaborative interactions among our project partners. From the biological experiments of ants colonies *Temnothorax albipennis* we inferred the bottom-up rules and roles of feedback in collective performance of ants brood-sorting and nest construction after emigration to a new site. We also analysed the observational data from the self-organized infrastructural development of an *eco-village* by an open community of volunteers resided in Ireland. These studies helped us to formalize the generic rules into a concrete model and to validate this model by deploying it in our robot controllers within the context of a manufacturing shop-floor scenario. In this section, we have described the robotic validation of AFM, with a brief presentation on interpretations of AFM from different social perspectives.

4.1. Generic framework

Inspired from the DOL in ants, humans and robots, we have proposed the following four rules that are the potential ingredients to obtain self-regulation in any social system. In this dissertation, these rules are mentioned as the *generic rules of self-regulation.*

Rule 1: Continuous flow of information. Self-regulatory social systems establish the continuous minimum flow of information over the period of time when self-regulation can be defined. This should help to maintain at least two states of an agent: 1) receiving information about task(s) and 2) ignoring information or doing no task. The up-to-date information should reflect the changes of the system i.e. it encodes the necessary feedback for the agents. Thus, this property will act as the basis of the state switching of agents, between these two minimum states or, among multiple states e.g. in case of multiple tasks or many sub-states of a single task.

At the individual level, information is processed differently by each individual, and is certainly not constant nor continuous. In addition, there can be lower and upper thresholds on the amount of info necessary for DOL to take place. The time scale at the individual level is very small compared to the system level's time scale.We can approximate the propagation of information at this macro time scale as the continuous flow of information. In the model, emphasize is given to whether the information is used e.g. stimulation to perform a task, or unused e.g. random walk.

Rule 2: Sensitization. Self-regulatory social systems allow the differentiation in the use of (or access to) information, e.g. through sensitization or learning of some tasks. This differentiation is regulated by the characteristics of the system, e.g. the ability of the agents to learn tasks that are repeatedly performed.

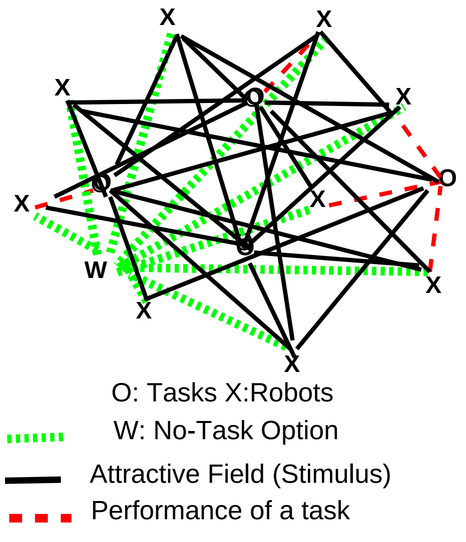

Figure 3: The attractive filed model (AFM)

Rule 3: Concurrence. Self-regulatory social systems include concurrent access to information from different spatial positions with certain preferences. This preference is not fixed and can change with the dynamics of the system.

Rule 4: Forgetting. Self-regulatory social systems include forgetting, e.g. the ability of the agents to diminish information over time, if not used. The system determines the amount of information being released, and this changes over time. For example, specialists might have to attend an emergency situation and switch tasks that contributes to the forgetting of old task experiences. This is considered as crucial to allow flexibility in the system. Having this general framework of self-regulatory social systems, we can now formalize AFM that will describe the properties of individual agents and the system as a whole. In terms of networks, the model is a bipartite network, i.e. there are two different types of nodes. One set of nodes describes the sources of the attractive fields and the other set describes the agents. Links only exist between different types of nodes and they encode the flow of information so that, even if there is no direct link between two agents, their interaction is taken into account in the information flow. This is an instance of *weak* interaction. The strength of the field primarily depends upon the distance between the task and the agent. This relationship is represented using weighted links. Besides, there is a permanent field that represents the *no-task* or option for ignoring information. The model can be mapped to a network as shown in Fig. 3. The correspondence is given below:

1. Source nodes (o) are tasks that can be divided between a number of agents.
2. Agent nodes (x) an be ants, human, robots etc.
3. The attractive fields correspond to stimuli to perform a task, and these are given by the black solid lines.
4. When an agent performs a task, the link becomes different, and this is denoted in the figure by a dashed line. Agents linked to a source by a red line are the agents currently doing that task.
5. The field of ignoring the information (w) corresponds to the stimulus to random walk, i.e. the no-task option, and this is denoted by the dotted lines in the graph.
6. Each of the links is weighted. The value of this weight describes the strength of the stimulus that the agent experiences. In a spatial representation of the model, it is easy to see that the strength of the field depends on the physical distance of the agent to the source. Moreover, the strength can be increased through sensitisation of the agent via experience (learning). This distance is not depicted in the network, it is represented through the weights of the links. In the figure of the network (Fig. 3), the nodes have arbitrary places. Note that even though the distance is physical in this case, the distance in the model applied to other systems, needs not to be physical. It can represent the accessibility to the information, the time the information takes to reach the receiver, etc.

In summary, from the above diagram of the network, we can see that each of the agents is connected to each of the fields. This means that even if an agent is currently involved in a task, the probability that it stops doing it in order to pursue a different task, or to random walk, is always non-zero. The weighted links express the probability of an agent to be attracted to each of the fields.

4.2. Relationship of AFM with self-organization

It is interesting to note that our proposed four generic rules can be considered as the four major foundations of a self-organized system (Fig. 4). Self-organized systems exhibit four distinct perspectives known as so-called ingredients or properties of self-organization [**?**]. However, it is not clear how those properties can come into existence. Here, we have described the four underlying mechanisms that explains how self-organization can be realized in different social systems using our generic framework of self-regulation. From our understanding, we can explain it in the following ways.

Firstly, multiple interactions become meaningful when *continuous flow of information* occurs by exchanging signals or cues among agents or their environment

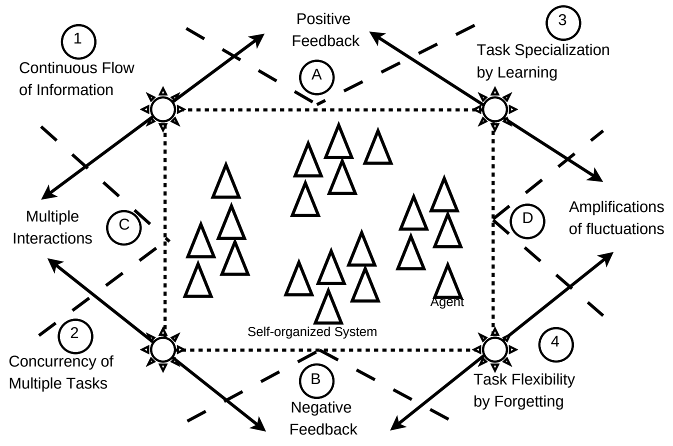
1
Continuous Flow of Information
Positive Feedback
A
3
Task Specialization by Learning
Multiple Interactions
C
D
Amplifications of fluctuations
Agent
Self-organized System
2
Concurrency of Multiple Tasks
B
Negative Feedback
4
Task Flexibility by Forgetting

Figure 4: Four generic rules establish the self-regulated DOL in social systems

that regulates their behaviours. This, in turn, contribute to the task-allocation and task-switching in the social level. In swarm intelligence literature, multiple interactions are often described as an essential ingredient of self-organization. However, interactions without definite purposes may not contribute to the self-organization.

Secondly, in swarm intelligence, positive feedback has been attributed as another mechanism of self-organization. But it is not easy to understand what creates positive feedback in a social system. Possible answers might be the characteristic of the environment e.g. ants select shorter path since density of pheromones becomes higher and thus more ants becomes attracted in that path, the gradual decrease of response-threshold of individuals which increases the probability of selecting a task etc. To make the answer more concrete, we have explicitly attributed *sensitisation* or learning as a mechanism of positive feedback. There might exist other mechanisms too. But clearly sensitisation will be one of the reliable mechanisms for achieving positive feedback.

Thirdly, similar to positive feedback, we have proposed *forgetting* that contributes to provide negative feedback about a task or decreasing the probability to select it. Other negative feedback mechanisms can be implemented by assigning a saturation level to each task which is also present in our model, for details see [12].

Finally, creating artificial amplification of fluctuations or stochastic events is not a straight-forward issue. It throws many open questions. Does a system designer intentionally impose irregularity in task-performance of agents? Is random movement enough for simulating randomness in a system? Since emergencies do not always pop-up on request, we provide the rule of *concurrency* that enables agents to maintain even a small amount of probability of selecting a low-priority, or less sensitized or distant task. This concurrency mechanism provides a high-degree of robustness in the system such that all tasks can be attended even if specialization of agents delays them in switching to some of the tasks.

4.3. Interpretation of AFM in multi-robot systems

The interpretation of AFM in a multi-robot system also follows almost exactly as in generic interpretation. However, in order to make the interpretation more concrete, let us consider a manufacturing shop floor scenario, where N number of autonomous mobile robots are required to attend J number of shop tasks spread over a fixed area A. Let these tasks be represented by a set of small rectangular boxes resembling to manufacturing machines.

Let R be the set of robots $r_1, r_2, ..., r_n$. Let a task j has an associated task-urgency ϕ_j indicating its relative importance over time. If a robot attends a task j in the x^{th} time-step, the value of ϕ_j will decrease by an amount $\delta_{\phi_{INC}}$ in the $(x+1)^{th}$ time-step. On the other hand, if a task has not been served by any robot in the x^{th} time-step, ϕ_j will increase by another amount $\delta_{\phi_{DEC}}$ in $(x+1)^{th}$ time-step. Thus urgency of a task is updated by the following rules.

$$If\ the\ task\ is\ not\ being\ done:\ \phi_j \rightarrow \phi_j\ + \delta_{\phi_{INC}} \tag{1}$$

$$If\ the\ task\ is\ being\ done:\ \phi_j \rightarrow \phi_j\ - n\,\delta_{\phi_{DEC}} \tag{2}$$

Eq. 1 refers to a case where no robot attend to task j and Eq. 2 refers to another case where n robots are concurrently performing the task j.

In order to complete a task j, a robot r_i needs to be within a fixed boundary D_j. If a robot completes a task j it learns about it and this will influence r_i's likelihood of selecting that task in future, say through increasing its sensitization to j by a small amount, k_{INC}. Here, the variable affinity of a robot r_i to task j is called as its *sensitization* k^i_j. If a robot i does not do a task j for some time, it forgets about j and k^i_j is decreased, by another small amount, say k_{DEC} . Thus a robot's task-sensitization update follows these rules.

$$If\ task\ is\ done:\ k^i_j \rightarrow k^i_j\ +\ k_{INC} \tag{3}$$

$$If\ task\ is\ not\ done:\ k^i_j \rightarrow k^i_j\ -\ k_{DEC} \tag{4}$$

According to AFM, all robots will establish attractive fields to all tasks due to the presence of a system-wide continuous flow of information. The strength of these attractive fields will vary according to the dynamic distances between robots and tasks, task-urgencies and corresponding sensitizations of robots. Simplifying the generic implementation of AFM from [12], we can formally encode this stimuli of attractive field as follows.

$$S_j^i = tanh\{\frac{k_j^i}{d_{ij} + \delta}\phi_j\} \tag{5}$$

$$S_{RW}^i = tanh\left\{1 - \frac{\sum_{j=1}^{J} S_j^i}{J+1}\right\} \tag{6}$$

$$P_j^i = \frac{S_j^i}{\sum_{j=0}^{J} S_j^i} \quad where, \quad S_0^i = S_{RW}^i \tag{7}$$

Eq. 5 states that the stimuli of a robot r_i to a particular task j, S_j^i depends on r_i's spatial distance to j (d_{ij}), level of sensitization to j (k_j^i), and perceived urgency of that task (ϕ_j). In Eq. 5, we have used a very small constant value δ to avoid division by zero, in the case when a robot has reached to a task. Since S_j^i is a probability function, it is chosen as a *tanh* in order to keep the values between 0 and 1. Eq. 6 suggests us how we can estimate the stimuli of random walk or no-task option. This stimuli of random walk depends on the sum of stimulus of J real tasks. Here, random-walk is also considered as a task. Thus the total number of tasks become $J + 1$. The probability of selecting each task has been determined by a probabilistic method outlined in Eq. 7 which states that the probability of choosing a task j by robot r_i is directly proportional to its calculated stimuli S_j^i. Finally, let T_a be the allocated time to accomplish a task. If a robot can enter inside the task boundary within T_a time it waits there until T_a elapsed. Otherwise it will select a different task.

5. AFM based task-allocation solution

We have designed a set of manufacturing shop-floor scenario experiments for validating the effectiveness of our AFM in producing self-regulated MRTA. The overall aim of this design is to analyse the various properties of task-allocation and related other issues. In this section, we have described our manufacturing shop-floor scenario and our task-allocation solution for robot-controllers.

5.1. A manufacturing shop-floor scenario

By extending our interpretation of AFM in multi-robot system, we can set-up manufacturing shop-floor scenario. Here, 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

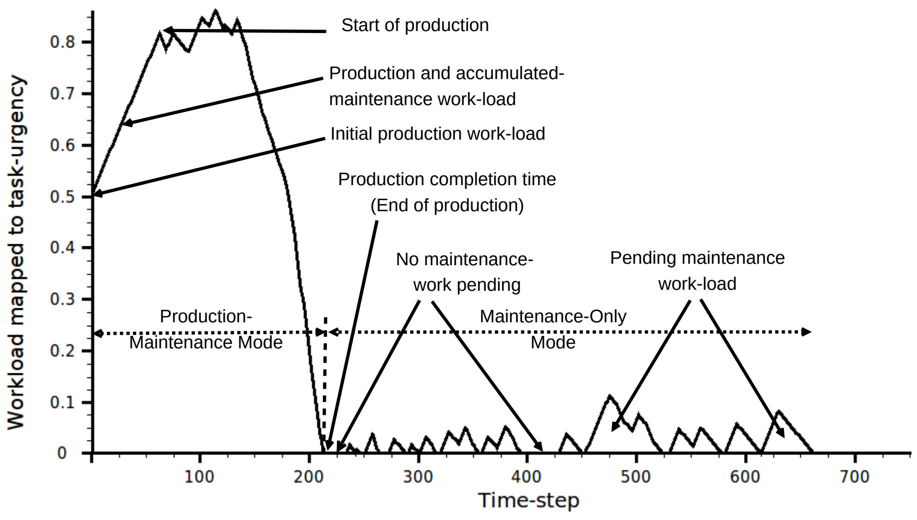

Figure 5: A manufacturing shop-floor production and maintenance cycle

material parts that can be 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, called hereafter as *minimum robots per machine (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. These robots, along with their production jobs, can do necessary maintenance works concurrently. For the sake of simplicity, here we consider $\mu = 1$.

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: 1) *production and maintenance mode* (PMM), and 2) *maintenance only mode* (MOM). Initially a machine starts working in PMM and does production and maintenance works concurrently. When there is no production work left, then it enters into MOM. Fig. 5 illustrates this scenario for a single machine.

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. 8.

$$\Phi_{j,INIT}^{PMM} = \Omega_j^p \times \Delta\phi_{INC} + \phi_j^{m0} \tag{8}$$

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. 9.

$$\Phi_{j,k}^{PMM} = \Phi_{j,INIT}^{PMM} + k \times \Delta\phi_{INC} \tag{9}$$

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 $\mu = 1$. 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:

$$\Phi_{j,k+1}^{PMM} = \Phi_{j,k}^{PMM} - \nu_k \times \Delta\phi_{DEC} \tag{10}$$

At a particular machine j, once $\Phi_{j,k}^{PMM}$ 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_j^{PMM}. Average production time-steps of a shop-floor with M machines can be calculated by the following simple equation.

$$T_{avg}^{PMM} = \frac{1}{M}\sum_{j=1}^{M} T_j^{PMM} \tag{11}$$

T_{avg}^{PMM} can be compared with the minimum number of time-steps necessary to finish production works, T_{min}^{PMM}. This can only happen in an ideal case where all robots work for production without any random walking or failure. We can get T_{min}^{PMM} from the total amount of work load and maximum possible inputs from all robots. If there are M machines and N robots, each machine has Φ_{INIT}^{PMM} task-urgency, and each time-step robots can decrease N $\times$ $\Delta\phi_{DEC}$ task-urgencies, then the theoretical T_{min}^{PMM} can be found from the following Eq. 12.

$$T_{min}^{PMM} = \frac{M \times \Phi_{INIT}^{PMM}}{N \times \Delta\phi_{DEC}} \tag{12}$$

$$\zeta_{avg}^{PMM} = \frac{T_{avg}^{PMM} - T_{min}^{PMM}}{T_{min}^{PMM}} \tag{13}$$

Thus we can define ζ_{avg}^{PMM}, average production completion delay (APCD) by following Eq. 13: When a machine enters into MOM, only μ robots are required to do its maintenance works in each time step. So, in such cases, if no robot serves a machine, the growth of task-urgency will follow Eq. 9. 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:

$$\Phi_{j,k+1}^{MOM} = \Phi_{j,k}^{MOM} - (\nu_k - \mu) \times \Delta\phi_{DEC} \tag{14}$$

By considering $\mu = 1$, Eq. 14 will reduces to Eq. 10. Here, $\Phi_{j,k+1}^{MOM}$ 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 a no-task option (random-walking). Interestingly PMW will indicate the robustness of this system since higher PMW value will indicate the delay in attending maintenance works by robots. We can find the *average pending maintenance work-load* (APMW) per time-step per machine, χ_j^{MOM} (Eq. 15) and average PMW per machine per time-step, χ_{avg}^{MOM} (Eq. 16).

$$\chi_j^{MOM} = \frac{1}{K}\sum_{k=1}^{K} \Phi_{j,k}^{MOM} \tag{15}$$

$$\chi_{avg}^{MOM} = \frac{1}{M}\sum_{j=1}^{M} \chi_j^{MOM} \tag{16}$$

5.2. Robot-controller algorithms

Task-allocation algorithm:

Stage 1. In the beginning of our task-allocation algorithm, we count the total number of tasks and initialize the sum of the stimuli all tasks to zero. Then for each task, we extract its pose, urgency, sensitisation from input data.
CalculateTaskDistance() function calculates the Euclidian distance between a task and a the robot by taking the current robot-pose and static task-pose as the input. These values are enough to get a task's stimuli based on Eq. 5. In this loop finally we update *TaskRecords* for using it in next task-allocation cycle. We get the random-walk task's stimuli from Eq. 6. It requires total number of shop-tasks and sum of their stimulus.

Algorithm 4.1: Self-regulated task-allocation based on AFM

```
1: Input: $AllTaskInfo, RobotPose, TaskRecords, DeltaDistance$
```

```
Output: $SelectedTaskID$
comment: Stage 1: Get each task's individual stimuli and sum all stimulus
$TotalTasks \leftarrow$ Total number of tasks $\in AllTaskInfo$
$TaskStimuliSum \leftarrow 0$
for all $Task \in AllTaskInfo$ do
    $TaskPose \leftarrow$ Task-position $\in Task$
    $TaskUrgency \leftarrow$ Task-urgency $\in Task$
    $TaskSensitization \leftarrow$ Task-sensitization $\in Task$
    $DistanceToTask \leftarrow$ CalculateTaskDistance($RobotPose, TaskPose$)
    $TaskStimuli \leftarrow$ CalculateTaskStimuli($DistanceToTask,$
                        $TaskSensitization, TaskUrgency, DeltaDistance$)
    $TaskStimuliSum \leftarrow TaskStimuliSum + TaskStimuli$
    $TaskRecords \leftarrow$ UpdateTaskRecords($TaskStimuli,$
                        $DistanceToTask, TaskUrgency, TaskSensitization$)
end for
$RandomWalkStimuli \leftarrow$ CalculateRandomWalkStimuli($TotalTasks,$
                        $TaskStimuliSum$)
$AllStimuliSum \leftarrow TaskStimuliSum + RandomWalkStimuli$
comment: Stage 2: Find probability of each task based on its stimuli
$TaskID \leftarrow 0$
while $TaskID \leq TotalTasks$ do
    $TaskStimuli \leftarrow$ Task-stimuli $\in TaskRecords(TaskID)$
    $TaskProbability \leftarrow$ GetTaskProbability($TaskStimuli, AllStimuliSum$)
    $TaskProbabilityRange \leftarrow$ ConvertTaskProbabilityIntoRange(
                        $TaskProbability$)
    $TaskRecords \leftarrow$ UpdateTaskRecords($TaskProbability$)
end while
comment: Stage 3: Draw a random-number to match with TaskID
$RandomNum \leftarrow$ GetRandomNumber(0, Max($TaskProbabilityRange$))
while $TaskID \leq TotalTasks$ do
    $RangeStart \leftarrow$ Min($TaskProbabilityRange(TaskID)$)
    $RangeEnd \leftarrow$ Max($TaskProbabilityRange(TaskID)$)
    if $RandomNum \geq RangeStart$ && $RandomNum \leq RangeEnd$ then
        $SelectedTaskID \leftarrow TaskID$
    end if
end while
```

Stage 2. In the second stage of our task-allocation algorithm, we find the probability of each task (including random-walk) based on Eq. 7. Then this probability value of each task, between 0 and 1, is rounded to the closest two-digit fractions and multiplied by 100 and put into a linear scale. For example, if two tasks probability values are: 0.15 and 0.25, the probability range of first task becomes 0 to 15 and for second task, it becomes 16 to 40.

Stage 3. After converting each task's probability values into a linear range, we use a random-number generator that draws a random-number between 0 and the highest value of task-probability range, say 40 for the previous example. Then this random number is compared against the task probability range of each task. For the above example, if the random-number becomes 37, our task-probability range checking function selects *Task2* since 37 falls between 16 to 40. Under this probabilistic method, the task with larger probability range has a higher chance to be selected, but low probability tasks are also got selected time-to-time.

Algorithm 4.2: Robot's learning and forgetting of tasks based on AFM

```
1: Input: $TaskRecords, LeranRate, ForgetRate, SelectedTaskID$
2: Output: Updated $TaskSensitisation \in TaskRecords$
3: for all $Task \in TaskRecords$ do
4:     $TaskID \leftarrow$ ID $\in Task$
5:     $TaskSensitization \leftarrow$ Sensitisation $\in Task$
6:     if $TaskID \equiv SelectedTaskID$ then
7:         $TaskSensitization \leftarrow \mathbf{Max}(1, (TaskSensitization + LeanRate))$
8:     else
9:         $TaskSensitization \leftarrow \mathbf{Min}(0, (TaskSensitization - ForgetRate))$
10:    end if
11: end for
```

Robot learning and forgetting algorithm:
Algorithm 4.2 lists the pseudo-code for updating the robot's task-sensitization values. Along with previously mentioned *TaskRecords*, and *SelectedTaskID*, it takes two other inputs: *LeranRate* (Δk_{INC}) and *ForgetRate* (Δk_{INC}) as outlined in Eq. eqn:k-inc and eqn:k-dec respectively. In addition to that it also keep the value of task-sensitization between the fixed limit of 0 and 1.

6. Experiments

In this section, we have described the design of our MRTA experiments within the context of our manufacturing shop-floor scenario.

6.1. Observables

Plasticity: Self-regulated DOL can be characterised by plasticity and task-specialization, in both macroscopic and microscopic levels. Within manufacturing shop-floor context, plasticity refers to the collective ability of the robots to switch

from doing no-task option (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) during PMM. Similarity, when there would be low workload in case of MOM only a few robots would do the task, rest of them would either be idle (not doing any task) or perform a random-walk. The changes of task-urgencies and the ratio of robots engaged in tasks can be good metrics to observe plasticity in MRTA.
Task-specialization: Under heavy work-load most of the robots should attend to tasks. But self-regulated DOL is always accompanied with task-specializations of agents. That means that few robots will be more active than others. From 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. If a few robots specializes on a particular task that will help to reduce traffic near the task and improves overall efficiency of the system. Thus, at the end of our production cycle in manufacturing shop-floor scenario, we can count the percentage of robots specializes on each task in our experiments.
Quality of task-performance: As discussed in Sec. 5.1 we can measure the quality of MRTA from the APCD. It first calculates the ideal minimum production time and then finds the delay in production process from the actual production completion data. Thus this will indicate how much more time is spent in the production process due to the self-regulation of robots in this distributed task-allocation scheme. In order to calculate APCD, we can find the production completion time for each task from the raw log of task-urgency and make an average from them.
Robustness: In order to see if our system can respond to the gradually increasing workloads, we can measure APMW within the context of our manufacturing shop-floor scenario. This can show the robustness of our system where a task can be unattended for long time. 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.
Flexibility: 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.
Energy-efficiency: 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

Table 1: Experimental parameters of Series A & B experiments

Parameter	Series A | Series B
Total number of robots (N)	8 | 16
Total number of tasks (M)	2 | 4
Experiment area (A)	2 m^2 | 4 m^2
Initial production work-load/machine (Ω_j^p)	100 unit
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

by our robots.
Information flow: Since AFM requires a system-wide continuous flow of information, we can measure the communication load to bench-mark our implementation of communication system. This bench-mark data can be used to compare among various communication strategies. Here we can measure how much task-related information, i.e. task-urgency, location etc. are sent to the robots at each time step. This amount of information or communication load can be constant or variable depending on the design of the communication system. **Scalability:** In order to see the effects of scaling on MRTA, we have designed two series 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, 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. All of the above metrics of Set A and Set B can be compared to find those scalability effects.

Thus, in order to observe the above properties of self-regulated MRTA , we have designed our experiments to record the following observables in each time-step.

1. Task-urgency of each task (ϕ).
2. Number of robots engaged in each task.
3. Task-sensitizations (k) of robots.
4. Pose data of robots.
5. Communication of task-information message with robots.

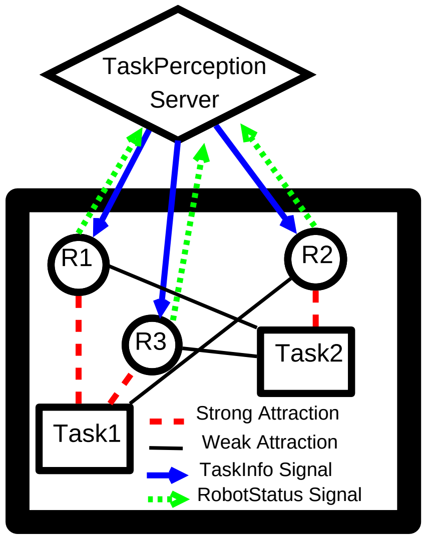

Figure 6: A centralized communication scheme

6.2. Parameters

Table 1 lists a set of essential parameters of our experiments. We intend to have a set-up that is relatively complex, i.e., with a high number of robots and tasks in a large area. The diameter of the marker of our e-puck robot is 0.08m. So, if we put 4 robots in an area of one square meter, this will give us a robot-occupied-space to free-space ratio of about 1:49 per square meter. This ratio reasonable in order to allow the robots to move at a speed of 5 cm/sec without much interference to each other.

The initial values of task urgencies correspond to 100 units of production work-load without any maintenance work-load as outlined in Eq. 8. 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. We choose a default sensitization value of 0.1 for all tasks. The following relationships are maintained for selecting task-urgency and sensitization parameters.

$$\Delta\phi_{INC} = \frac{\Delta\phi_{DEC} \times N}{2 \times M} \tag{17}$$

$$\Delta k_{DEC} = \frac{\Delta k_{INC}}{M - 1} \tag{18}$$

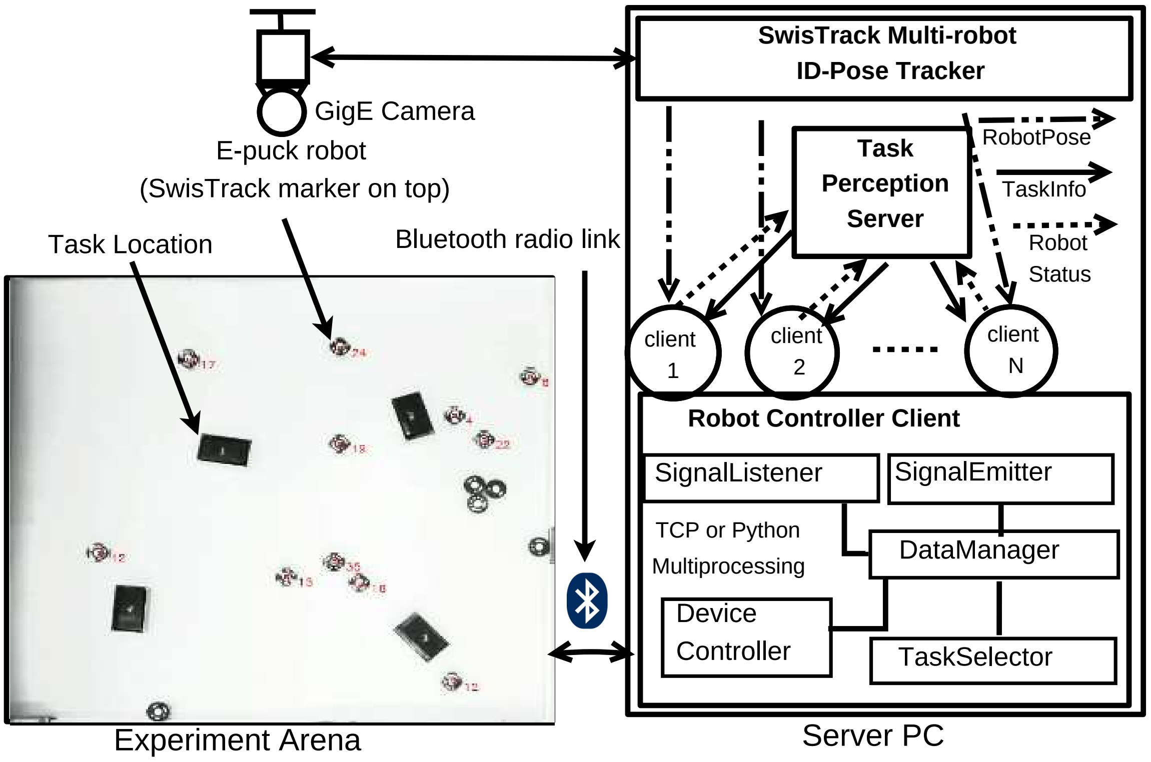

Figure 7: Hardware and software setup for series A & B experiments

Eq. 17 establishes the fact that task urgency will increase at a higher rate than that of its decrease. As we do not like to keep a task left unattended for a long time we choose a higher rate of increase of task urgency. This difference is set on the basis of our assumption that at least half of the expected number of robots (ratio of number of robots to tasks) would be available to work on a task. So they would produce similar types of increase and decrease behaviours in task urgencies.

Eq. 18 suggests that the learning will happen much faster than the forgetting. The difference in these two rates is based on the fact that faster leaning gives a robot more chances to select a task in next time-step and thus it becomes more specialized on it.

6.3. Implementation

Ideally, AFM can be implementation 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 pick-up tasks.

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 robots. For example, our robots are not capable of sensing a task to estimate its urgencies, instead our centralized *task-perception server (TPS)* broadcast task information, e.g. task-urgencies, locations etc. to robots in certain time intervals. Within this interval, if some robots work on a task, they independently send their status, i.e. which task they are currently doing. From this status message, TPS can re-calculate the task-urgencies and send them in next broadcast.

Fig. 6 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.

This centralized communication system can be converted into a decentralized one where robots can use local observation and communication with peers about tasks to estimate task-urgencies. In Chapter **??**, we present an emulation of this scenario where robots do not depend entirely on TPS for estimating task-urgencies, instead they get task information from TPS when they are very close to a task (inside a pre-defined task-boundary) or from local peers who know about a task via TPS.

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 discussed in Sec. **??**, our robots do not have on-board CPU, they need a host PC to control them using BTCom communication protocol. 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. Below we describe the actual implementations of these components, i.e. D-Bus communication interfaces and detail implementations of TPS and RCC.

7. Results

8. Discussions

9. Conclusions

References

[1] R. C. Arkin, Behavior-based robotics, Cambridge, Mass. : MIT Press, c1998., 1998, includes bibliographical references (p. [445]-476) and indexes.

[2] L. E. Parker, F. Tang, Building multirobot coalitions through automated task solution synthesis, Proceedings of the IEEE 94 (2006) 1289–1305.

[3] B. P. Gerkey, M. J. Mataric, A formal analysis and taxonomy of task allocation in multi-robot systems, The International Journal of Robotics Research 23 (2004) 939.

[4] A. B. Sendova-Franks, N. R. Franks, Self-assembly, self-organization and division of labour', Philosophical Transactions of the Royal Society of London B 354 (1999) 1395–1405.

[5] B. Gerkey, 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 3.

[6] 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. 2 (no. 2) (2008) 5–14.

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

[8] M. B. Dias, R. M. Zlot, N. Kalra, A. Stentz, Market-based multirobot coordination: A survey and analysis, Proceedings of the IEEE 94 (2006) 1257–1270.

[9] K. Lerman, C. Jones, A. Galstyan, M. J. Mataric, Analysis of dynamic task allocation in multi-robot systems, The International Journal of Robotics Research 25 (2006) 225.

[10] C. R. Kube, H. Zhang, Collective robotics: From social insects to robots, Adaptive Behavior 2 (1993) 189.

[11] W. Liu, A. F. T. Winfield, J. Sa, J. Chen, L. Dou, Towards energy optimization: Emergent task allocation in a swarm of foraging robots, Adaptive Behavior 15 (3) (2007) 289–305.

[12] E. Arcaute, K. Christensen, A. Sendova-Franks, T. Dahl, A. Espinosa, H. J. Jensen, Division of labour in ant colonies in terms of attractive fields.

[13] R. Jeanne, Group size, productivity, and information flow in social wasps, Information processing in social insects (1999) 3–30.

[14] L. E. Parker, Alliance: an architecture for fault tolerant multirobot cooperation, Robotics and Automation, IEEE Transactions on 14 (1998) 220–240.

[15] W. Shen, D. H. Norrie, J.-P. Barthes, Multi-agent systems for concurrent intelligent design and manufacturing, Taylor & Francis, London, 2001.

[16] P. Stone, M. Veloso, Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork, Artificial Intelligence 110 (1999) 241–273.

[17] C. Belta, V. Kumar, Abstraction and control for groups of robots, IEEE Transactions on Robotics 20 (5) (2004) 865–875.

[18] G. Pereira, V. Kumar, M. Campos, Decentralized algorithms for multirobot manipulation via caging, Algorithmic Foundations of Robotics V (2003) 257–274.

[19] T. Dahl, M. Mataric, G. Sukhatme, Distributed multi-robot task allocation through vacancy chains, Autonomous Robots.

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

[21] M. Krieger, J. Billeter, The call of duty: Self-organised task allocation in a population of up to twelve mobile robots, Robotics and Autonomous Systems 30 (1-2) (2000) 65–84.

[22] W. Agassounon, 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.

[23] C. Jones, 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, Vol. 2, 2003.

[24] N. Kalra, A. Martinoli, A comparative study of market-based and threshold-based task allocation, Distributed Autonomous Robotic Systems 7 (2007) 91–101.

[25] B. B. Werger, M. J. Mataric, Broadcast of local eligibility for multi-target observation, Distributed Autonomous Robotic Systems 4 (2001) 347356.

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

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

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

[29] H. Çelikkanat, A. Turgut, E. Şahin, Guiding a robot flock via informed robots, Distributed Autonomous Robotic Systems 8 (2008) 215–225.