Available online at www.sciencedirect.com

ScienceDirect

Cognitive Systems Research 43 (2017) 21–44

Cognitive Systems RESEARCH

www.elsevier.com/locate/cogsys

A socially-based distributed self-organizing algorithm for holonic multi-agent systems: Case study in a task environment

Ahmad Esmaeili [a,*], Nasser Mozayani [a], Mohammad Reza Jahed Motlagh [a], Eric T. Matson [b,c]

[a] *School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran*
[b] *Department of Computer and Information Technology, Purdue Polytechnic Institute, Purdue University, West Lafayette, IN, USA*
[c] *Department of Electrical Engineering, Kyung Hee University, Yongin, South Korea*

Received 15 May 2016; received in revised form 25 October 2016; accepted 3 December 2016
Available online 26 December 2016

Abstract

Holonic multi-agent systems (HMASs) have recently attracted many researches in multi-agent systems community. Inspired from the multi-level and self-similar structures of social and biological system, holonic multi-agent systems have been widely used to model and solve complex real-world problems. The main concern in deploying HMASs is the problem of building the hierarchical holonic structure, called holarchy, and dynamically managing it during its lifetime. The way an HMAS is organized has a great impact on its applicability and performance. This paper proposes a self-organizing algorithm to build and manage the holoic structures in multi-agent systems. This algorithm is based on the local information of the agents about other agents they can communicate with. Using common social concepts, like skills, diversity, social exchange theory, and norms in definition of the algorithm, the outcomes of this research can be used in wide ranges of distributed applications. The proposed model is extensively tested in a task allocation problem; and its performance based on various design parameters is studied. Empirical results show that the proposed model properly increases the performance of the system in terms of effectiveness and efficiency.

Keywords: Multi-agent systems; Holonic multi-agent systems; Self organization; Group theory; Task assignment

1. Introduction

In recent years, intelligent agent technologies have been used pervasively in many applications such as e-commerce, transportation, mobile technologies, graphic design, etc. There are various definitions for intelligent agents, based on their application domains. One of well-known and widely used definitions is introduced in Wooldridge (2009) as follows: An agent is a computer system that is situated in some environment, and that is capable of autonomous action in order to meet its design objectives. A group of such autonomous agents that are interacting each other, directly or indirectly, to move towards common goals is termed Multi-Agent System (MAS). The distributed nature of MAS-based solutions has made them promising alternatives for classic problem solving methods, especially in decomposable large-scale applications.

The success of multi-agent system approaches depends on several key factors. These factors include, but not limited to, the ability of the agents to cooperate on collective goals, the coordination of resources and agents' actions,

* Corresponding author.
E-mail addresses: aesmaeili@iust.ac.ir (A. Esmaeili), mozayani@iust.ac.ir (N. Mozayani), jahedmr@iust.ac.ir (M.R. Jahed Motlagh), ematson@purdue.edu (E.T. Matson).

http://dx.doi.org/10.1016/j.cogsys.2016.12.001
1389-0417/© 2016 Elsevier B.V. All rights reserved.

and negotiation mechanisms, all of which can be handled by means of various organizational models (Ferber, Gutknecht, & Michel, 2004). The organization concept has become commonplace in MAS research community and depending on the diverse backgrounds of researchers, various definitions have been introduced in the literature (Weiss, 2013). An organization consists of the definition of roles, relationships, and control/authority structures. In other words, an organization model specifies the way agents interact with each other over a long term goal, and how these interactions affect resource allocation, data flow, authority levels, and coordination patterns, among many other system characteristics (Carley & Gasser, 1999; Hayden, Carrick, & Yang, 1999). Organization models are essential parts of most multi-agents systems. Being defined implicitly or explicitly in MAS designs, these models play a significant role in the emergence of complex behaviors from simple agents, on one hand, and the overall reduction of the complexity of the system, on the other hand. The contributions of organizational approaches to multi-agent domains include modularity, security of applications, limiting the scope of interactions, management of uncertainty and redundancies, and formalizing high-level goals (Ferber et al., 2004; Horling & Lesser, 2004). COIN (Coordination, Organization, Institutions, and Norms)[1] is an internationally well-known community in multi-agent systems that focuses on the research on organizational structures in MAS and their coordination patterns.

Several organization models, with different characteristics, have been proposed in the literatures, such as hierarchies, holarchies, coalitions, teams, congregation, societies, federations, markets, matrix, and compound organizations, among others (DeLoach & Matson, 2004; Irandoust & Benaskeur, 2008). Holonic organizations are among the successful organizational models that have been introduced in multi-agent systems recently, and resulted in the concept of Holonic Multi-Agent System (HMAS) (Gerber, Siekmann, & Vierke, 1999). Based on its unique structure, a holonic model brings several significant attributes to multi-agent systems, such as self-similarity, reliability, stability, and dynamism, to name a few. These features, as a result, make holonic multi-agent systems be considered as a promising approach in many large scale distributed real-world problems.

In a holonic multi-agent system, the agents are organized in self-similar nested groups, termed holons. Holarchy, on the other hand, is a multi-level structure composed of holons, and allows the modeling of a MAS at several granularity levels (Rodriguez, Gaud, Hilaire, Galland, & Koukam, 2007a). Similar to the other multi-agent organizations, size and shape of the holons, in terms of their compositions, have great impacts on the performance and efficiency of the entire system. Therefore, there is no surprise to expect numerous holonification algorithms proposed in the HMAS literature. Nevertheless, due to the relatively young age of the researches in HMAS, these algorithms are still immature. This paper makes an attempt to contribute to this domain by proposing a distributed self-organization algorithm for holonic multi-agent systems.

In general, the organization of a holonic multi-agent system consists of two primary stages: building the initial holarchy; and controlling its structure against internal and external stimuli during its lifetime. The algorithms proposed for this purpose, can be classified into two main categories: (i) centralized methods, in which a central system is responsible for the whole process; and (ii) decentralized algorithms, in which the holonfication process is managed distributedly by the agents themselves. Each of these classes of methods has its own advantages and disadvantages. In centralized algorithms, the system needs to have all the information about the members at any given time. Although such an assumption may ease the holonification process to a large extend, it limits the application of the holonic multi-agent systems in large and open systems, and also makes them suffer from single point of failure problem. In distributed holonification methods, on the other hand, extendibility and suitability for large-scale open multi-agent system problems are achieved with an expense of high coordination needs, resulting from the local and limited domain of information that is available for the agents.

The proposed algorithm of this paper belongs to the second category of the above-mentioned classification. In other words, in our model, there is no central unit for building and controlling the holarchy, and the whole process is controlled by the member agents, according to their local information about themselves and their neighbors in a multi-agent network. This makes our proposed algorithm more realistic (being close to human social structures) and proper for open system that grow over time.

Our proposed model is presented and assessed in a task environment. Being adopted from the literatures on human organizations and management, a task environment consists of suppliers, distributors, and clients that have a direct impact on the management of the organization and hence the achievement of the organizational goals. In our model for such an environment, a set of tasks with various characteristics and complexities enter to the system. Upon their entrance, the agents/holons of our model work together to complete them in a timely manner. This problem is very close to the extensively studies area of task assignment/distribution in multi-agent systems. Among the related works, we can cite the one reported in Scerri, Farinelli, Okamoto, and Tambe (2005), in which a distributed task allocation algorithm, namely LA-DCOP, is introduce for large-scale multi-agent teams; and the work conducted in Dos Santos and Bazzan (2012), where a distributed agent clustering method, based on swarm intelligence, is introduced.

[1] http://www.pcs.usp.br/coin/.

There are numerous organization models for multi-agent systems, such as AGR (Agent, Group, Role) (Ferber, Gutknecht, & Michel, 2003), MOISE (Hannoun, Boissier, Sichman, & Sayettat, 2000), MOISE+ (Hübner, Sichman, & Boissier, 2002), Electronic Institutions (Esteva, Rodriguez-Aguilar, Sierra, Garcia, & Arcos, 2001), MaSE (DeLoach, Wood, & Sparkman, 2001), and the works in Irandoust and Benaskeur (2008), Gaston and DesJardins (2005), Sims, Corkill, and Lesser (2008), Dignum, Meyer, Weigand, and Dignum (2002) and Kota, Gibbins, and Jennings (2012). Here, we refer to those proposed for the holonic structures of HMAS. As mentioned before, the first challenge in the holonification process is the appropriate selection of agents to reside in holons. One of the earliest works on this problem is the one proposed by Zhang (Zhang & Norrie, 1999) based on Product-Resource-Order-Staff Architecture (PROSA) (Van Brussel, Wyns, Valckenaers, Bongaerts, & Peeters, 1998). In their model, two sets of holons, namely static and mediator holons, are used to organize the holonic structure. As their model has been proposed in manufacturing domain, the static holons of their model correspond to physical and information objects in the environment, such as devices, plans, conveyors, etc. As new tasks arrive, a Dynamic Mediator Holon (DMH) is made to tie different static holons together to perform the tasks. Upon the completion of the tasks, the corresponding DMHs are destroyed. Being based on a specific domain and making use of static predetermined holons, this model is inappropriate to be used as a general purpose method in MAS problems.

In the work by Stefanoiu (Stefanoiu, Ulieru, & Norrie, 2000), the authors have used a fuzzy entropy minimization method to build the holarchy as sets of clusters. For this purpose, a collection of holons, with potential assignments of them to clusters and the probability of the occurrence of those clusters, are assumed. Then, the agents are chosen to work inside holons such that the entropy of the assignments are minimized. Ulieru further extended this model by making use of a genetic algorithm to search for possible clustering assignments (Ulieru, 2002).

One of the most successful and generic frameworks reported for the self-organization of holonic multi-agent systems, is the work by Rodriguez in Hilaire, Koukam, and Rodriguez (2008). In their framework, the authors have proposed an organizational approach by extending the RIO model (Hilaire, Koukam, Gruer, & Müller, 2000). More specifically, the organization of their model, in an abstract level, is made up of roles, as first class entities, and of holons that play those roles, in the concrete level. Inspired by artificial immune systems, the self-organization is controlled through the merging and dissolving operations that are defined based on the concepts of satisfactions and affinity. The primary drive of self-organization in their methodology is based on capability matching of holons and predetermined roles and interactions among those roles. This, as a result, makes the organization structure be static and inapt for dynamic task environments.

In another research by Ciufudean (Ciufudean & Filote, 2011), the authors have developed a class of Petri nets to propose a Holonic Social System (HoSS). In their model, physical resources and information flows are controlled and managed by Petri nets, towards the creation of flexible production systems. In this work, a centralized unit is needed to pre-determine and analyze all aspects of the system using Petri nets in order to build the holons. Such restrictions make this framework an unsuitable option for open and dynamic systems where a distributed approach is needed. In another work (Jie, Wei-Ming, Bao-Xin, & Zhong, 2011), the authors have reported an organization model for HMAS. In their model, it has been assumed that there are predefined types of agents, namely head and function agents, and the holarchy is built based on the negotiation between the heads. Their model seem to be generic and applicable in task environments, however, it lacks any precise algorithm for building the holarchy and reorganizing it. Furthermore, it is assumed that the agents can communicate with every other member to build the holarchy.

The latest work on holonification is presented in Barbosa (2015). Their proposed model is based on ADAptive holonic COntrol aRchitecture (ADACOR) (Leitão & Restivo, 2006) architecture and gets inspirations from biological systems like swarms in order to self-organize the holarchy. This model, called ADACOR2, performs self-organization at two levels: behavioral or micro self-organization of the individual agents; and structural or macro self-organization to deal with the relationships between the holons of the holarchy. In spite of being distributed, this work differs from our proposed model in various aspects. Unlike ADACOR2, our model is not limited to a particular application and hence, does not have any predefined sets of holons. Furthermore, ADACOR2 assumes that all system entities have information about the others, while in our model, according to the local knowledge of agents, there is no such assumption to confine it to be inappropriate for open systems.

Our proposed model in this article is based on the interaction networks among the agents. That is, without any loss of generality, it assumes that initial system is a multi-agent network. Among the holonification works on networked multi-agent systems, we can refer to Abdoos, Esmaeili, and Mozayani (2012) and Esmaeili, Mozayani, and Jahed Motlagh (2014). Both of these researches are merely on building the holarchy, and not re-organizing them during its lifetime. Moreover, through a central unit, the holarchy is built based on the connections of the agents and before the system is deployed for solving any problem.

The rest of this paper is organized as follows. In Section 2, a brief introduction to holonic multi-agent systems and related terminologies are given. Section 3 presents our model of self-organization in HMAS in details. In Section 4, the proposed model is assessed through various experiments; and a comprehensive discussion over the results is given. Finally, Section 5 concludes this paper and presents some ideas for future works.

2. Holonic multi-agent systems

The word "holon" is derived from the Greek word "holos", meaning whole, and the suffix "on", meaning part. This concept has roots in biology and sociology, as it was introduced for the first time by a Hungarian philosopher, named "Arthur Koestler", to describe the recursive and self-similar structures in biological and sociological entities (Koestler, 1968). The history about this concept goes back to the time when Koestler was studying the hierarchical structures in both living organism and social organizations. According to his findings, there are no absolute "wholes" and "parts" in the structures of biological and sociological entities. In other words, all the components of these systems can only be understood by recognizing the sub-parts they are composed of and the super-components they are subordinate of. According to Koestler, holons are stable and coherent self-similar structures that simultaneously consist of several sub-structures, called sub-holons, and are part of a greater structure, called super-holon. The first notable use of holonic structures was in the domains of Manufacturing (Leitao & Restivo, 2008), and Business Process Management (BPM) (Clegg, 2007). Nevertheless, they have found their way into other disciplines like Artificial Intelligence.

The use of holonic concepts in definition and building of organizations has lead to the creation of hybrid, recursive, and multi-level structures that benefit key characteristics such as stability, autonomy, and cooperation capacity (Rodriguez, Hilaire, & Koukam, 2007b). In the multi-agent community, such a hierarchical structure is called holarchy, and can be modeled by means of whole-part relationships among the system components.

According to Koestler, holonic systems, under the term of Open Hierarchical Systems, are defined using series of rules as follows (Adam & Mandiau, 2003):

- A holonic system can be represented in a tree-like structure.
- A holon can be considered as a part of another holon, and as a whole that contains other holons.
- A holons is flexible in adopting required strategies or obeying precise principles.
- As we move towards the higher levels of holarchy, the actions of the holons become complex, while simple and reactive actions are performed by the holons in the lower levels.
- The communications are limited to the same level members of a holon, and the holons at different levels of a holarchy through heads.

Organizing the agents in a multi-level structure leads to Holonic Multi-Agent Systems (HMAS), composed of holons in super- and sub-holon relationships. A super-holon is a composite holon that contains a nonempty subset of the lower-level holons; and similarly, sub-holons are atomic/comosite holons that are members of an upper-level composite holon. According to the previously mentioned rules, a holonic multi-agent system needs to have a flexible hierarchical structure. The flexibility of the structure is derived from the flexible autonomy of the holons and their ability to change the intra-holon arrangements. According to flexible autonomy, a manager, called head, sets the goals of its holon, while the members themselves choose the strategy to use, based on their local knowledge.

In a HMAS, the holons of any level can interact only with the ones in the same level, and the ones in their immediate upper or lower levels through the heads of the holons. It should be noted that, at any level, the interactions between any holon and the one that does not belong to the same super-holon as the first one does - this is called an alien holon to the first holon - are needed to be managed by the head holons. Fig. 1 shows a four-level holarchy, with intra-holon communications. In this holarchy the heads of the holons are shown in red, and the levels are numbered from bottom to top. As depicted, the level of abstraction decreases as we move towards the bottom of holarchy, i.e., the lowest level holons are responsible for the real actions in the environment.

It has been proven that holonic solutions are effective in problems that can be modeled with hierarchical and self-organizing structures (Rodriguez, Hilaire, & Koukam, 2005). This has led to successful application of holonic multi-agent systems in a broad range of complex real world problems, including flexible manufacturing systems (Barbosa, Leitão, Adam, & Trentesaux, 2013), transportation (Abdoos, Mozayani, & Bazzan, 2013), health organizations (Ulieru & Geras, 2002), and oil industry (Marcellino & Sichman, 2010).

As mentioned earlier, one of the main steps in tackling the real world complex problems with HMAS is to construct the holarchy and handling it over time. In this paper, we get inspirations from sociology and social networks to devise a general-purpose self-organization methodology for holonic multi-agent systems. This is done by preventing the issues of most related works and applying more realistic assumptions.

3. Proposed model

In this section, we present our model of self-organization for holonic multi-agent systems. This model uses a bottom up distributed approach to form the holons and the holarchy. The distributed nature of the proposed method not only makes our model more realistic and close to human societies, but also makes it appropriate for large scale open multi-agent applications, where new agents can be added over time. Furthermore, since no central unit is used to control the holonification process, this model does not suffer from the single point of failure problem. The pro-

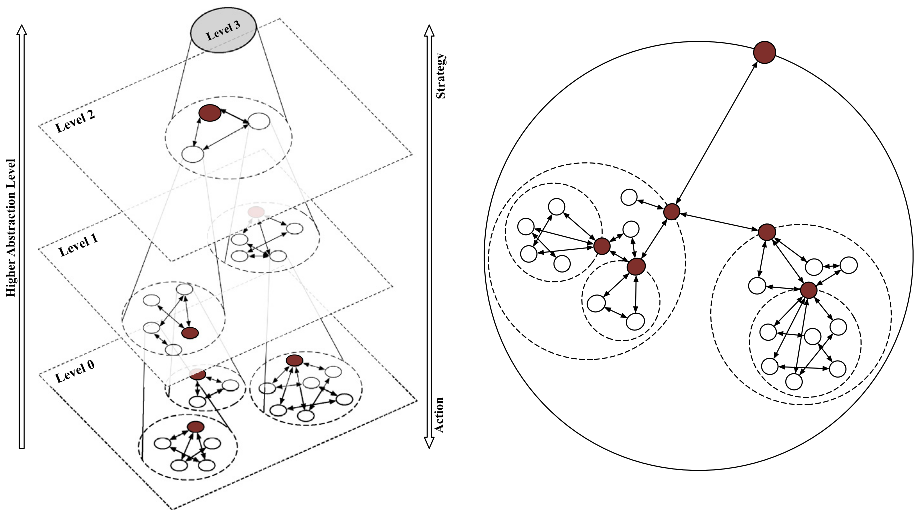


Fig. 1. A holarchy composed of four holarchical levels, in two views, hierarchical (left), flat (right). The red circles in every level are the heads. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

posed self-organization framework assumes an interaction network among the agents and uses machine-learning techniques together with social concept of norms to structure the holarchy. In order to provide better understanding of the model, we apply them in a task environment.

3.1. Multi-agent network

As mentioned before, one of the essential characteristics of multi-agent systems is the interaction and communication between its agents. In other words, the intelligent agents, as the member of the multi-agent system, need to interact in order to cooperate and coordinate towards a collective goal. When the size of a MAS is small, it can be assumed that all the agents can be connected to each other in order to interact. However, when the size of a MAS increases, in terms of the number of agents, such an assumption is not feasible due to limitations in resources and coordination complexities. As a consequence, in almost all multi-agent systems, there are networks on which the agents interact and communicate each other. Such networks help the agents keep track of only their neighbors rather than all other agents, and hence reduce the overall coordination complication. These interaction networks are either predefined by the system designers or implicitly used by the agents themselves according to conditions like homophily, friendship, the proximity of agents, etc., just similar to human societies.

Our method assumes the presence of an interaction network among the agent/holons before it starts. This interaction network is modeled and represented by a graph in which the nodes are the agents and the edges represent the interaction between the corresponding agents. Various properties of a networked MAS can be represented by different types of graphs. For instance, a directed weighted graph can be used to model a multi-agent system, in which the communications between the agents are unilateral and with different qualities. The interaction network used in our model is a graph of undirected unweighted type, which is formally denoted by tuple $MAN\langle A, I\rangle$, where A and I are respectively agents set and interaction links between the agents. This network is represented by symmetric matrix Λ, in which item $\Lambda(a_i, a_j)$ indicates the presence of any bilateral interaction link between a_i and a_j. That is:

$$\Lambda = \begin{cases} 1 & \text{if } (a_i, a_j) \in I \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Fig. 2 illustrates an example of a multi-agent network composed of 8 agents and 16 interaction links.

3.2. Holons and holarchy

Having a multi-agent network as defined before, the self-organization algorithm tries to build the holarchy according to the tasks that are introduced to the system over time. The holarchy is defined in terms of holons and holonic interactions at different levels. Formally, a holarchy with h holarichal levels is denoted by set H and is defined as in Eq. (2), where H^i, is the set of all holons in level i of the holarchy.

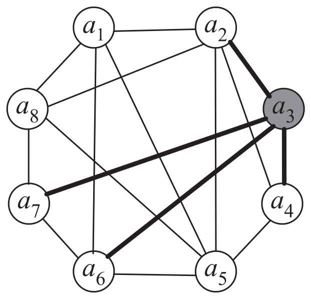
$$H = \{H^0, H^1, H^2, \ldots, H^{h-1}\} \tag{2}$$

Similarly, the set of all holons in level i of the holarchy is defined as in Eq. (3), where H^i_j denotes the j-th composite holon in level i of the holarchy, and n_i equals the number of composite holons in the i-th level.

$$H^i = \{H^i_1, H^i_2, H^i_3, \ldots, H^i_{n_i}\} \tag{3}$$

Please note that, in this definition, the Holon of the h-th level of the holarchy, i.e., H^h is a single composite holon such that:

$$H^h = \{H^h_1\}; \quad \forall H^{h-1}_j \in H^{h-1} : H^{h-1}_j \in H^h_1 \tag{4}$$



$$A = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8\}$$

$$\begin{aligned} I = \{&(a_1,a_2),(a_1,a_5),(a_1,a_6),(a_1,a_8),(a_2,a_3),\\ &(a_2,a_4),(a_2,a_5),(a_2,a_8),(a_3,a_4),(a_3,a_6),\\ &(a_3,a_7),(a_4,a_5),(a_5,a_6),(a_5,a_8),(a_6,a_7),\\ &(a_7,a_8)\} \end{aligned}$$

$$\Lambda = \begin{array}{c|cccccccc} & a_1 & a_2 & \mathbf{a_3} & a_4 & a_5 & a_6 & a_7 & a_8 \\ \hline a_1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ a_2 & 1 & 0 & \mathbf{1} & 1 & 1 & 0 & 0 & 1 \\ \mathbf{a_3} & 0 & \mathbf{1} & 0 & \mathbf{1} & 0 & \mathbf{1} & \mathbf{1} & 0 \\ a_4 & 0 & 0 & \mathbf{1} & 0 & 1 & 0 & 0 & 0 \\ a_5 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ a_6 & 1 & 0 & \mathbf{1} & 0 & 1 & 0 & 1 & 0 \\ a_7 & 0 & 0 & \mathbf{1} & 0 & 0 & 1 & 0 & 1 \\ a_8 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{array}$$

Fig. 2. An example multi-agent network with 8 agents and 16 links.

According to this definition, base holon set H^0 contains the real agents of the system, which we call atomic holons. Formally, $H^0 = \{a_1, a_2, a_3, \ldots, a_N\}$ or $\forall a_i \in A : H^0_i = a_i$. Thus, throughout this paper, wherever we mention atomic/base holons, we mean the corresponding agents, and vice versa. As an example, Fig. 3 depicts the formal representation of the holarchy of Fig. 1.

Furthermore, we also assume that the holons of every level of holarchy do not overlap each other, that is:

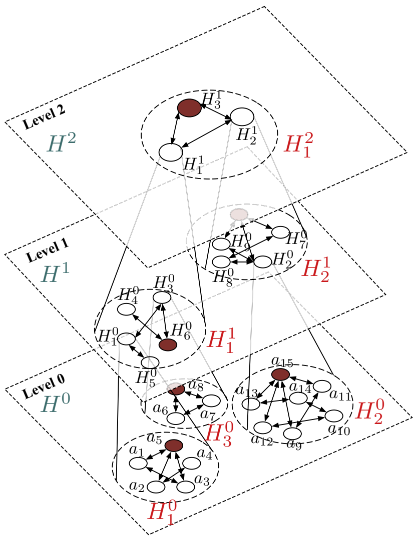
$$\forall i,j,k \in \mathbb{N}; i \in [1, h-1]; j,k \in [1, n_i]; \quad j \neq k \\ : H^i_j \cap H^i_k = \varnothing \tag{5}$$

In addition to the holon sets, there are interaction networks connecting holons inside every level of holarchy. The set of all holonic interaction networks at different holarchical levels is denoted by HI and is defined as in Eq. (6), where h denotes the height of the holarchy and HI^i is the set containing all the interaction links in level i of the holarchy.

$$HI = \{HI^0, HI^1, HI^2, \ldots, HI^{h-1}\} \tag{6}$$

Considering a symmetric adjacency matrix for every level, similar to the one in Fig. 2, this set is defined as in Eq. (7), where $\Lambda_i(H^i_j, H^i_k)$ is the item at the corresponding row and column of the adjacency matrix of level i, Λ_i.

$$HI^i = \{(H^i_j, H^i_k) | \Lambda_i(H^i_j, H^i_k) = 1; 1 \leqslant j \neq k \leqslant n_i\} \tag{7}$$



$$H^2_1 = \{H^1_1, H^1_2, H^1_3\}$$

$$H^2 = \{H^2_1\}$$

$$H^1_1 = \{H^0_1, H^0_3, H^0_4, H^0_5, H^0_6\}$$
$$H^1_2 = \{H^0_2, H^0_7, H^0_8, H^0_9, H^0_{10}\}$$

$$H^1 = \{H^1_1, H^1_2\}$$

$$H^0_1 = \{a_1, a_2, a_3, a_4, a_5\}$$
$$H^0_2 = \{a_9, a_{10}, a_{11}, a_{12}, a_{13}, a_{14}, a_{15}\}$$
$$H^0_3 = \{a_6, a_7, a_8\}$$

$$H^0 = \{H^0_1, H^0_2, H^0_3\}$$

Fig. 3. Formal representation of the holarchy given in Fig. 1.

In other words, HI^i contains all pairs of the holons in level i such that there is a link between them. The holonic interaction networks are built during the self-organization algorithm. Please note that, in this representation, there is no interaction between different levels of the holarchy. Similar to the holon sets, the base holonic interactions are the same interactions between the agents as defined for MAN, i.e., $HI^0 = I$. Defining the holons as above, we denote the entire holarchy by tuple $Holarchy\ \langle H, HI \rangle$.

As stated before, all holons of the holarchy, except the base holons, have a head to manage the intra-holon activities and facilitate inter-holon communications. In our model, the heads are holonic members with specific characteristics. The detailed metric used for selecting the head is given later in the holonification algorithm.

3.3. Tasks and holonic skills

We assess our proposed model in a task environment. In this environment, various task are introduced to the system at specific intervals. Detailed definition of tasks is based on the concept of holonic skills. These skills specify how capable the holons are and what types of tasks they can accomplish. Every agent (atomic holon) introduced to the system contains a limited number of capabilities or skills. Considering set $S = \{s_1, s_2, \ldots, s_\sigma\}$ as the set of all σ skills defined in the system, we denote the skill lists of atomic holon $H_i^0 \in A$ by set HS_i^0, such that $HS_i^0 \subseteq S$ and $HS_i^0 \neq \emptyset$. In case of the upper level holons, we just simply define the holonic skills as the union of the skill sets of their members, as defined in Eq. (8), where h is the height of holarchy; n_i is the number of holons at level i of the holarchy; and finally, H_j^i specifies the j-th holon at level i of the holarchy, as defined in previous section.

$$\forall i,j,k \in \mathbb{N}; i \in [1,h]; j \in [1,n_i]; k \in [1,n_{i-1}] : HS_j^i = \bigcup_{H_k^{i-1} \in H_j^i} HS_k^{i-1} \tag{8}$$

According to this definition of skills, we define each task t_i with tuple $\langle id_i, TS_i, pt_i, ta_i, py_i \rangle$, where id_i is a unique identifier for the task; non-empty set $TS_i \subseteq S$ is the list of skills required to fulfill the task; pt_i is the processing time of the task, i.e., the number of time steps needed to perform the task; ta_i specifies the task age, i.e., the number of time steps the task will be valid; and py_i is the payment for fulfilling the task. In our model, we calculate the payment in terms the number of skills it need and its processing time as $py_i = |TS_i| \times pt_i$. It should also be noted that, in our proposed model we assume that the tasks are divisible and additive. As an example, a task with required skill list $TS = \{s_1, s_2, s_3\}$ can be split in several sub-parts like $ts_1 = \{s_1, s_2\}$ and $ts_2 = \{s_3\}$ (divisible) each of which can be performed by an individual holon. On the other hand, after they are completed by the committed holons, the total output is resulted by cumulatively combining the outputs from performing aforementioned subtasks (additive).

3.4. Holonification

In this section we present the detailed algorithm for constructing the holarchy. As mentioned before, the holonification process starts with a connected multi-agent network (MAN) and is distributedly carried out by the agents. Unlike many other holonification algorithms that are performed at design time, in our model, holarchy is built, as the systems is deployed to perform the introduced tasks. The building block of the proposed algorithm is shown in the pseudocode of Algorithm 1.

Algorithm 1. Distributed Holonic Organization algorithm

```
1: procedure MainFunc($\langle H^0, HI^0 \rangle$)
2:     $SA \leftarrow H^0$          ▷ Stand Alone Holons
3:     $T_i \leftarrow$ GetNewTask ()
4:     while $T_i \neq \emptyset$ do        ▷ There is still a task to perform
5:         $H_j^l \leftarrow$ Initiator($SA, T_i$)
6:         $Result \leftarrow$ Initiate($H_j^l, T_i$)
7:         Process($Result$)
8:         Update($SA$)
9:         $T_i \leftarrow$ GetNewTask()
10:    end while
11: end procedure
```

As it can be seen, this algorithm starts with the initial multi-agent network. At the start of the algorithm, the agents of the initial population do not belong to any super-holon. Therefore, they are assigned to the list of standalone holons SA (line 2 of Algorithm 1). As long as there is a new task to perform, a proper holon is found to initiate the task. This is done using function Initiator in line 5 of Algorithm 1, which searches among all standalone holons and returns the proper one according to their skills and neighbors. It should be noted that this searching process is done by broadcasting the request for performing the task, and collecting their responses and possible proposals. The function that is used by the standalone holons to calculate and report their fitness for a specific task is given in Algorithm 2.

Selecting a proper holon from the pool of accepted holons can vary based on the application they are used. Throughout this paper the first holon that sends an ACCEPT message is chosen. Considering the j-th holon of level l, H_j^l, to be the selected initiator, function Initiate of line 6 of Algorithm 1 asks H_j^l to start executing task T_i. According to the result of the execution, system properly pays the initiator using function Process. Finally, the list of standalone holons is updated in line 8 of Algorithm 1. This update is done due to the fact that any newly created holon is going to be the representative of its standalone members in the list.

Algorithm 2. Fitness calculation algorithm

```
1: function FITNESS($T_i$)    ▷ This function is run by standalones
2:     if IDLE $Skills \cap TS_i \neq \emptyset$
3:         return REJECT
4:     else if $rand \leqslant P_{init}$ and $status \neq BUSY$ then
5:         return ACCEPT
6:     else
7:         return REJECT
8:     end if
9: end function
```

In Algorithm 2, a standalone holon computes its fitness based on its skills. In case of not being already busy on another task, the more fitting the holon is, the most probable it accepts to perform the task. Formally, fitness, in term of an initiation probability P_{init}, is defined as in Eq. (9), where $|\dots|$ denotes the cardinality of a given set, and NHS_j^l is the set of all skills provided by the neighbors of holon H_j^l in the holonic interaction network of HI^l, as defined in Eq. (10).

$$P_{init} = \begin{cases} 0 & \text{if } \{TS_i \cap HS_j^l\} = \emptyset \\ \frac{|\{HS_j^l \cup NHS_j^l\} \cap TS_i|}{|TS_i|} & \text{otherwise} \end{cases} \tag{9}$$

$$NHS_j^l = \left\{ \bigcup HS_k^l \;\middle|\; (H_j^l, H_k^l) \in HI^l \right\} \tag{10}$$

The above-mentioned equation for calculating P_{init} ensures that the holon does not take part in initiating any task that is not possibly going to be beneficial. Furthermore, status variable of every holon keeps track of its standing in the task environment. In our model, at any time, holons have one of the following three values as their statuses:

- IDLE: when the holon is not processing any task. This is the default value for any new holon introduced to the system.
- COMMITTED: when the holon has a skill appropriate for a task, and thus is committed to be part of the collectives working on that task.
- BUSY: holons in this status are actively working on a task. A holon goes from status COMMITTED to status BUSY as soon as all the necessary skills for accomplishing the task are provided. Please note that, changing to state BUSY requires the task be still active (according to its age). The holons remain in BUSY state as long as the processing time of the task is not over.

The proposed model for organizing holonic multi-agent systems defines several social norms that all holons should agree upon. These norms are as follows:

- **Joining norm:** a new member joins an existing holon only if both sides agree on the join. In case of inviting a member, the invitee accepts the invitation only if it is not member of any other super holon.
- **Leaving norm:** an existing member of a holon can leave the holon at any time it wants, without any need for permission from the head. On the other hand, head can also fire any member whenever it desires. Furthermore, any member holon with status COMMITTED cannot leave its super holon nor can it be fired.
- **Task performing norm:** when a non-head holon receives a task, it accepts the task if it has been passed by a holonmate (a holon of the same super holon) and if the holons have idle skills for the task.
- **Task initiation norm:** holons can only initiate a new task if they are heads or if they are IDLE and standalone (a holon without any super holon).
- **Leadership norm:** head of a holon, can accept and pass arriving tasks regardless of its status. This right is exceptionally reserved for head to prevent blocking the members from performing tasks. Besides, head of a holon is the last member that can leave the holon.
- **Interaction norm:** members of a holon can directly interact and communicate their holonmates if and only if there is a direct link between them. In case of outer holon interactions, non-head holons can not interact with holons of other levels. However, they can receive or send outer holon messages to alien holons (holons belonging to other superholons) of the same level, unless, for security reasons and depending on the application they are explicitly prohibited.

It should be noted that our model, being based on the definition of holonic multi-agent systems presented in Gerber et al. (1999), does not hinder the autonomy of holons. In fact, these norms define the way holons give up a part of their autonomy to their super-holons and their corresponding objectives.

As mentioned in Algorithm 1, after finding a proper holon to initiate a given task, the system, as a universal holon holding all components, assigns the task to that holon to initiate. Please note that this assignment process is not in contrast to our self-organization model, since it does not dictate any external decision. During the initiation, based on the given task and skills of the initiator holon, any of the following cases might occur:

1. The initiator creates a new holon and invites others to perform the task.
2. The initiator holon joins any neighbor super-holon.
3. The initiator holon has enough idle skills to perform the task by itself.

The detailed initiation algorithm is provided in Algorithm 3. Lines 2 and 3 of this algorithm divides the needed skills list of given task T_i into two groups: the ones the performer holon can provide and the ones that holon needs some help from others. Next, the initiator holon checks whether it needs any help at all or if it is a member of another superholon (in case the all agent are holonified, there would be no standalone holon in the system except the universal holon). In either case, it assigns the task to the head. If the result of this assignment is the set of all skills required by the task, it means that the head holon, with the help from any possible neighbors, can perform the task and therefore it is commanded to do so. The result of function Perform is returned as the result of initiation. If the result of assignment cannot satisfy all the required skills, the age of the task is checked and if it is over, the head is asked to release all committed skills for the task and then returns FAIL as the result of initiation. Otherwise, the age of the task is reduced one unit; and rescheduling the task for another assignment, it returns PENDING as the result of initiation. The pending case is used to wait another time step to see if new skills, committed or busy on other tasks, are going to be freed. This is going to be repeated for the task until there are enough skills to accomplish the task or the task's age is over.

Algorithm 3. Task initiation algorithm

```
1:  function InitiateTask($T_i$)    ▷ This function is run by initiator holon
2:    $Mine_i \leftarrow TS_i \cap$ Idle($Skills$)    ▷ Skills I can provide
3:    $Others_i \leftarrow TS_i - Mine_i$    ▷ Skills for others to provide
4:    if $Others_i = \varnothing$ or $Superholon \neq \varnothing$ then
5:      $result(id_i) \leftarrow Head.$ AssignTask ($TS_i, id_i$)
6:      if $TS_i - result(id_i) = \varnothing$ then
7:        $PerfRes \leftarrow Head.$ Perform($TS_i, id_i$)
8:        return $PerfRes$
9:      else if $ta_i = 0$ then
10:       $Head.$ ReleaseSkillsFor ($id_i$)
11:       return FAIL
12:     else
13:       $ta_i \leftarrow ta_i - 1$
14:       Schedule($T_i$)
15:       return PENDING
16:     end if
17:   else if $Superholon = \varnothing$ then
18:     $SAN \leftarrow$ Neighbors ($HI^l, \varnothing$)    ▷ Standalone neighbors
19:     $EMN \leftarrow$ Neighbors($HI^l, S \neq \varnothing$)    ▷ Employed neighbors
20:     $H_k^{l+1} \leftarrow$ CreateHolon($this$)
21:     $Needs \leftarrow Others_i$
22:     for all $H_m^l \in SAN$ do
23:       if $Needs \neq \varnothing$ then
24:         $InvRes, Sk \leftarrow$ InviteToJoin ($H_m^l, Needs$)
25:         if $InvRes =$ ACCEPT then
26:           AddMember ($H_m^l, H_k^{l+1}$)
27:           $Needs \leftarrow Needs - Sk$
28:           Introduce ($H_m^l.EMN, H_k^{l+1}$)
29:         end if
30:       else
31:         break    ▷ Getting out of For loop
32:       end if
33:     end for
34:     if $Needs = \varnothing$ and $TS_i - \{$ AssignTask ($Mine_i, id_i) \cup Others_i\} = \varnothing$ then
35:       $PerfRes \leftarrow this.$ Perform($TS_i, id_i$)
36:       Register($H_k^{l+1}$)
37:       Introduce($EMN, H_k^{l+1}$)
38:       return $PerfRes$
39:     else if $Others - Needs \neq \varnothing$ then
40:       Register($H_k^{l+1}$)
41:       Introduce($EMN, H_k^{l+1}$)
42:       $this.$ ReleaseSkillsFor ($id_i$)
43:       return PENDING
44:     else
45:       Delete($H_k^{l+1}$)
46:       $Superholon \leftarrow \varnothing$
47:       for all $H_m^l \in EMN$ do
48:         if $HS_m^l \cap Others \neq \varnothing$ then
49:           $JoinRes \leftarrow$ JoinRequest ($H_m^l.Superholon$)
50:           if $JoinRes =$ACCEPT
51:             Join ($H_m^l.Superholon$)
52:             Introduce ($EMN, H_m^l.Superholon$)
53:             Schedule ($T_i$)
54:             return PENDING
55:           end if
56:         end if
57:       end for
58:       return FAIL
59:     end if
60:   end if
61: end function
```

In case that above conditions do not hold, the holon checks to see if it is standalone. Being standalone provides proper settings for creating a new holon. For this purpose, the holon needs to find and classify its neighbors based on their standing (being standalone or employed in another holon) in the system, according to the holonic interaction network of its own level (lines 18 and 19 of Algorithm 3). Please note that in this algorithm, it is assumed holon H_j^l is running the initiation function, therefore, the holonic interaction network is HI^l. In line 20 of Algorithm 3, the holon creates new holon H_k^{l+1}, containing himself as the only member and head. When a standalone holon initiates the task and has the option to choose between a neighboring composite holon and another neighboring standalone holon, it prefers the latter one by creating a new superholon. This way, not only does it get the opportunity to be head, the size of the holons are kept small, inspiring from human groups that almost tend to be small in size (Mullen, 1987). This idea is implemented in lines 22 though 33 of Algorithm 3. The inviting function in line 24 of Algorithm 3 has two return values, one for the result of invitation (ACCEPT or REJECT) and one for the skills that the invited holon can provide. As new members are added to the newly created holon, its holonic interaction should also be built. This is done by means of procedure Introduce in line 28. Specifically, by running this procedure, the runner holon informs superholons of all the employed neighbors of the added member about the newly created holon, so they can establish an interaction link with it. The recruiting process continues until either all standalone neighbors have been processed, or there is no need to recruit more standalone holons. Therefore, depending on which case occurs, the head deploys the following processes:

1. In case the collective of the head and recruited members can perform the task (line 34 of Algorithm 3), performing command is issued, newly created holon is registered as a standalone holon in the system, and finally the interaction links of the new holon is built using the similar process mentioned before (lines 35–38 of Algorithm 3). It should be noted that during the registration process, the standing of the members of new holon, which once were standalone but now are employed, are updated in the system as well.
2. If the recruiting algorithm ends without providing the needed resources for performing the task, the head returns FAIL message, however, the newly created holon continues to exist for the future assignments of the same or similar tasks, which would hopefully be performed in companion with upper level neighbors. Please be noted that as soon as the holon fails its committed members become IDLE again and hence they will not be hindered taking part in performing future tasks.

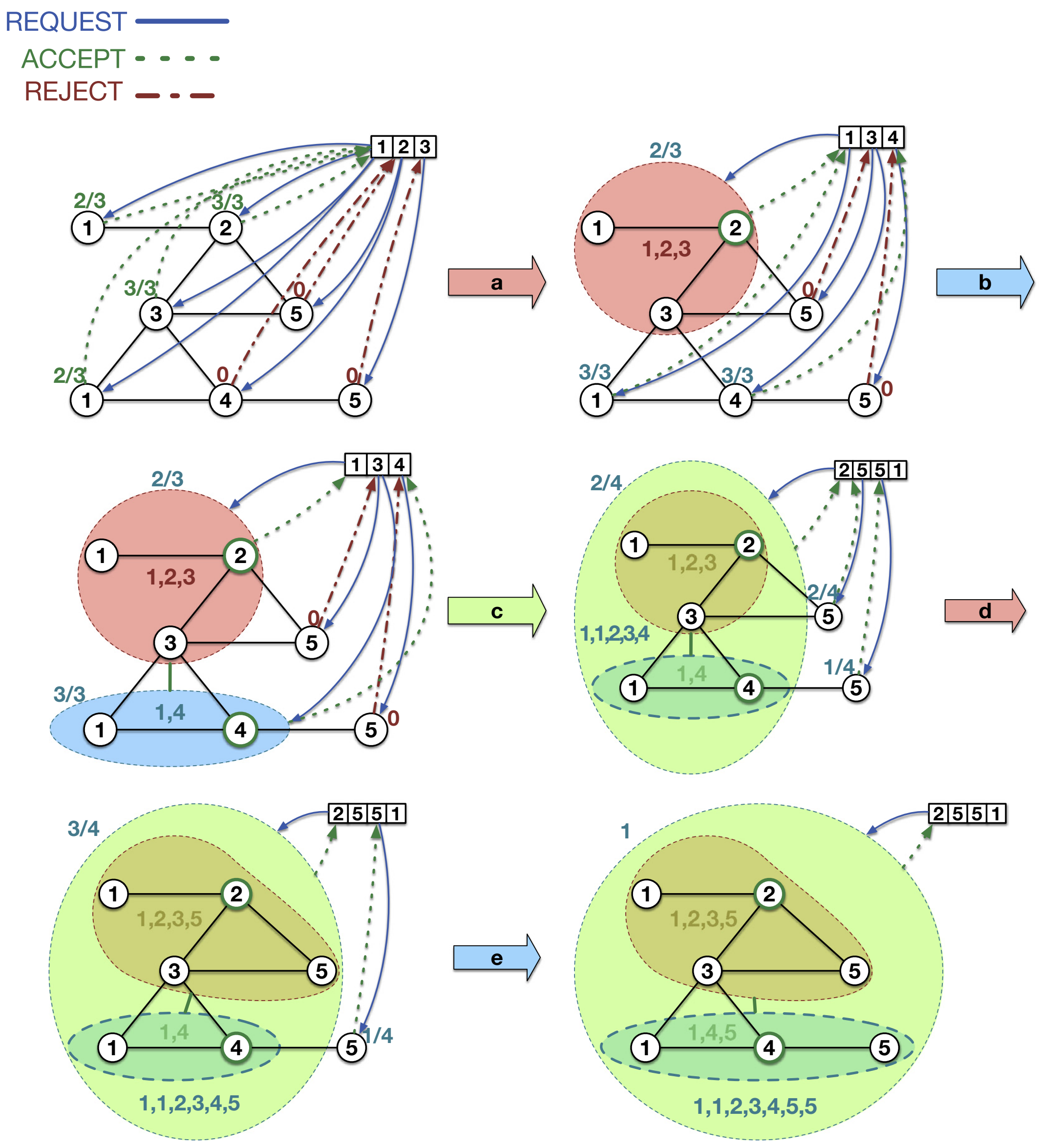


Fig. 4. Illustrative example for the proposed holonification algorithm.

Consequently, through the lines 40–42 of Algorithm 3, the new holon is registered and the interaction network of upper level holons are updated.

3. The holon creation process may be finished without neither finding proper standalones to recruit nor providing all the necessary skills. In this situation, the newly created singleton holon is deleted and H_j^l resets its superholon indicator (lines 45–46 of Algorithm 3). Then it looks for any composite holons in its vicinity (according to H^l) to join. This is done by sending join requests to the super-holons of its employed neighbors. The host holon may take several conditions into its consideration in order to accept or reject the join request. In our model, host holons accept/reject the requests based on the changes that would occure to their miscellany of capabilities. These miscellanies are caused by the differences in the capabilities, knowledge, and behaviors of the holons and are called social diversity (Esmaeili, Mozayani, Motlagh, & Matson, 2016). Formally, consider $\Delta_{pre}(H_o^{l+1})$ and $\Delta_{post}(H_o^{l+1})$ be the social diversities of host holon H_o^{l+1} respectively before and after accepting the join request. We have assumed that the host would accept the request with probability 0.6 if $\Delta_{pre}(H_o^{l+1}) \leqslant \Delta_{post}(H_o^{l+1})$ and with probability 0.4 if $\Delta_{pre}(H_o^{l+1}) > \Delta_{post}(H_o^{l+1})$. The value of 0.4 is chosen based on the research reported in Esmaeili et al. (2016). According to the research, diversity of 0.6 (in a scale from 0 to 1) inside a holon yields a relatively better performance in networked based holarchies. As soon as it finds a holon to join, the process is over with rescheduling the task, and then issuing message PENDING (lines47–58 of Algorithm 3). The social diversity of a holon is computed based on the skills that its members provide. In this paper, we have used Shannon's information entropy (Shannon, 1949) as in Eq. (11), where $frq_{HS_o^l}(s_i)$ is the frequency of skill s_i in skill set HS_o^l, and $|\ldots|$ denotes the cardinality of a given set.

$$\Delta(H_o^l) = -\sum_{s_i \in HS_o^l} \left(\frac{frq_{HS_o^l}(s_i)}{|HS_o^l|} \log_2 \frac{frq_{HS_o^l}(s_i)}{|HS_o^l|} \right) \tag{11}$$

Please be noted that, the rescheduling of the tasks that has led to PENDING message, is managed by the initiator holon, while rescheduling of the FAILed ones, in case their age is not over, is controlled by the system.

As it can be seen above, as soon as a new super-holon is built, its creator becomes the head of that holon. This should be a good choice at first when the holon is small and the creator is identified as a fitting member. However, as the system grows and its structure and interaction links are altered during time, the initial head may not be a proper choice anymore and should thus be reselected again. The detailed head selection algorithm we have used in our model is presented in Section 3.5.

Among the functions that were mentioned before, almost all have straightforward processes, except function AssignTask, which needs to be discussed in more details. This function is implemented as shown in the pseudocode of Algorithm 4.

Algorithm 4. Task assignment algorithm

```
1: function AssignTask($TS_i, id_i$)    ▷ This function is run by holons
2:   if $role \neq Head$ and $status = BUSY$ then
3:     return $\varnothing$
4:   else
5:     $IntNeighs \leftarrow$ Neighbors ($HI^l$, $Superholon$)
6:     $CommittedList \leftarrow \varnothing$    ▷ Skills that have been committed
7:     $CapableNeighs \leftarrow \varnothing$    ▷ Capable neighbors of the holon
8:     $IdlSks \leftarrow$ Idle (Skills)    ▷ Idle skills of the holon
9:     $MatchedSks \leftarrow IdlSks \cap TS_i$    ▷ Idle skills proper for the task
10:    if $MatchedSks \neq \varnothing$ then
11:      if $Subholon = \varnothing$ then
12:        SetCommitted($MatchedSks$)
13:        $CommittedList \leftarrow CommittedList \cup MatchedSks$
14:      else
15:        $SubAssigneds \leftarrow Head$. AssignTask ($MatchedSks, id_i$)
16:        SetCommitted($SubAssigneds$)
17:        $CommittedList \leftarrow CommittedList \cup SubAssigneds$
18:      end if
19:    end if
20:    for all $H_n^l \in IntNeighs$ do
21:      if !$H_n^l$.assigned($id_i$) then
22:        $Result \leftarrow H_n^l$.assignTask($TS_i - CommittedList, id_i$)
23:        if $Result \neq \varnothing$ then
24:          $CapableNeighs(id_i) \leftarrow CapableNeighs(id_i) \cup H_n^l$
25:          $CommittedList \leftarrow CommittedList \cup Result$
26:        end if
27:      end if
28:    end for
29:  end if
30:  return $CommittedList$
31: end function
```

The purpose of this algorithm is to properly assign the required skill of a task among the neighbors and subordinate holons. For this purpose, the caller holon, first checks to see if it plays the role of head inside the containing holon. If it does not, it simply rejects the assignment request by returning an empty set, just in case its current status is BUSY (lines 2–3 of Algorithm 4). Otherwise, it continues the assignment process. In the first step of the process, it distinguishes all its neighbors that are employed in the same containing holon that holds itself, called internal neighbors (line 5 of Algorithm 4). Then, through the lines from 6 to 9, the holon divides the assigned skills into two subsets: the ones that it can provide based on its idle skills, and the ones it should ask for help from the neighbors. If the self-assigned skills list is not empty (there is at least a skill that can be provided by the holon), the holon checks its holonic type. In case that it is an atomic one (one without any subholons), it reserves the self-assigned skills as COMMITTED, and stores them for future report. However, if it is a composite holon (a holon with subordinated holons as members), the holon recursively assigns the self-assigned skills to its subordinates through its head (lines 10–19 Algorithm 4). In the next step, the holon contacts its internal neighbors that have not been assigned for that specific task before, and recursively assigns them the skills that have not been provided by the holon or any of its previously assigned neighbors (lines 20–29 Algorithm 4). Finally, it returns the set of all skills that can be provided by the caller and its proper neighbors.

For better understanding of the holonification algorithm, a simple example is illustrated in Fig. 4. For the sake of simplicity, it is assumed that the agents only provide one skill, and tasks introduction continues until its age is over or it is accomplished. Furthermore, the initiation and assignments are done in such a way that there is at least a change at every step, and preventing wasting the space of the paper by duplicate figures.

As it can be seen in this figure, holonification starts with a multi-agent network consisting of 7 agents and 8 interaction links. The skills of the agents are printed inside of the nodes that represent the agents. As a task with $TS_1 = \{1,2,3\}$ enters the system, it is introduced to all standalone agents to initiate (represented by blue dashed lines). Computing the initiation probability (as written above the agent nodes), the standalones properly reply with ACCEPT (shown by green dashed lines) or REJECT (shown by red dashed lines) messages. For instance the agent with skill 2 together with its neighbors can provide skill set $\{1,2,3,5\}$, and using Eq. (9) its fitness is computed as $\frac{3}{3} = 1$. Through step ⟨a⟩, the agent with skill 2 is chosen as the initiator and it forms a new holon based on its own and its neighbors' skills. The newly created holon is depicted by a red circle, in which the initiator is the head. The formed holon successfully accomplishes the task since it owns all the skills needed by the task. As new task $TS_2 = \{1,3,4\}$ is introduced, the previous steps are repeated to find the initiator. According to the initiation probability, either of the neighboring agents with skills 1 or 4 can be chosen as the initiator. Here we assume the task is initiated by the agent with skill 4. Through step ⟨b⟩ a new holon is built (it is shown in blue color) and the upper level interactions are updated (new link is depicted by a green line connecting two new holons), however because of the lack of required skills, the newly created holon fails to accomplish the task. In the next round the task is reassigned to the standalones again. This time the blue holon initiates the task and forming a new holon (depicted in light green) through step ⟨c⟩, it succeeds to accomplish the task. Please note that the skills of the newly created holons are listed inside the ovel that represents them. Again a new task with skills list $TS_3 = \{2,5,5,1\}$ enters the system and hence, it is introduced to all standalone holons, as depicted. The standalone holon with skill 5 (the upper one) initiates the task. However, since it does not have any standalone neighbor, it joins the super-holon of one of its employed neighbors (we assumed that host holon accept the joining request) through step ⟨d⟩. Since the last initiation of task failed, it is introduced to the standalones again. We assume the standalone agent with skill 5 initiates the task (it has initiation probability of 0.25, so it is possible that it initiates the task, though after several steps). Again this initiator does not have any standalone neighbor to construct a new holon, therefore it joins the super-holon of its neighbor (in step ⟨e⟩). Finally, being able to provide all needed skills, the only standalone holon of the system (the green one) succeeds to perform the task.

3.5. Head selection algorithm

In human societies, leaders play a critical role in groups communications, decision-makings, and coordinating the members in order to augment the entire performance of the group. As a result, they are selected based on their social and personal powers, which can be defined in terms of their successful experiences, communication skills, personality traits, etc. (Forsyth, 2009). As it was explained in previous section, the heads of the newly created holons are the task initiators, by default. During the early ages of the holons, this choice may be proper, however, as the holons grow in size and their composition and interaction links change over time, the heading role should be assigned to more appropriate member of the holon.

In our model, we measure the power of the holons based on their immediate interactions with the other holons of the same level. More specifically, the more interactions a holon has (degree centrality of the corresponding node in the holonic network), the more information and resources the holon can provide and as a result, the more power it holds. On the other hand, because of the distributed nature of our model, the head selection algorithm should also be distributed. That is, the members of the holons choose among themselves for any appropriate candidate to be the head. In cases that the holons do not have exceptional features or capabilities in comparison with the others, the number of their interaction links can be considered as a good local measure for their power in the network.

The algorithm that is proposed in this paper can be run by the current head of holon at specific times during the holons' life or whenever a special event occurs. Algorithm 5 shows how the selection process is conducted.

Algorithm 5. Head selection algorithm

```
 1: function SELECTHEAD($Power, HolonID$)    ▷ This function is run by holons during the selection process.
 2:   $Holonmates \leftarrow$ NEIGHBORS($HI^l, Superholon$)
 3:   $HighestPower \leftarrow Power$
 4:   $HeadCandidate \leftarrow HolonID$
 5:   for all $H^l_k \in Holonmates$ do
 6:     if $!H^l_k$.REPORTEDBEFORE() then
 7:       $(NeighsHighestPow, Cand) \leftarrow H^l_k$. SELECTHEAD ($Power, HolonID$)
 8:       if $NeighsHighestPow > HighestPower$ then
 9:         $HighestPower \leftarrow NeighsHighestPow$
10:         $HeadCandid \leftarrow Cand$
11:       end if
12:     end if
13:   end for
14:   if $|$NEIGHBORS($HI^l$)$| > HighestPower$ then
15:     $HighestPower \leftarrow |$NEIGHBORS($HI^l$)$|$
16:     $HeadCandidate \leftarrow MyID$
17:   end if
18:   return ($HighestPower, HeadCandidate$)
19: end function
```

The head selection function is a recursive function that inputs the latest power value and the id of the latest most powerful holon as its arguments. Then the caller holon recursively searches among its holonmates that have not taken part in the selection process, for any better candidate for leadership. That is, the holon that belongs to the same super-holon and has more interaction links than the previous candidate (lines 5–13 of Algorithm 5). Having received the candidates of its neighboring holonmates, the holon then compares his own power to the candidate's and reports the winner of this comparison (lines 14–18 of Algorithm 5). It should be noted that, during this process, the power values of the holons are computed based on all interactions of that holon, regardless of the super-holon of the peers, while the selection process only happens among the members of the same holon. In order to give a better understanding of this algorithm, a simple illustrative example is provided in Fig. 5.

As it can be seen in this example, the holon with $ID = a$, as the former head of blue composite holon, initiates the head selection process by sending out its $(degree, ID)$ pair to holon with $ID = b$. Then holon b continues to pass the pair to its own unsent neighboring holonmate. This sending process continues until the pair reaches the holon with $ID = f$. Since this final holon does not have any unsent neighbors, it compares the last version of the message with its own power and then corrects the pair accordingly. The newly created pair goes the same path it has traveled, in reverse order, until it reaches the former head a. Head sends this updated pair to his other unsent neighboring holonmate neighbor, c. Since c does not have more power than f, it resends the message back to a. Finally the selection process is completed by choosing holon f as the new head of the holon. The order of all messages is printed before the pairs in this figure.

3.6. Profit distribution

Recalling the definition of tasks in the previous section, the profit of a holon is defined in terms of its success in performing a task and the amount of the payment that task provides. Due to the responsibility of the head holons in coordinating the others and their high load of interactions, we assume that they are paid more than the body holons in our proposed method. In addition, all the non-head members of the holon that have taken part in performing the task are paid equally. Paying only to those agents that have taken part in performing a specific task, and not all the members, reduces the presence of free riders. Formally, let H^k_j be the responsible holon - a holon that succeeds in initiating a task - for performing task $t_i \langle id_i, TS_i, pt_i, ta_i, py_i \rangle$, and $\eta^{j,k}_i$ be the number of its members, including

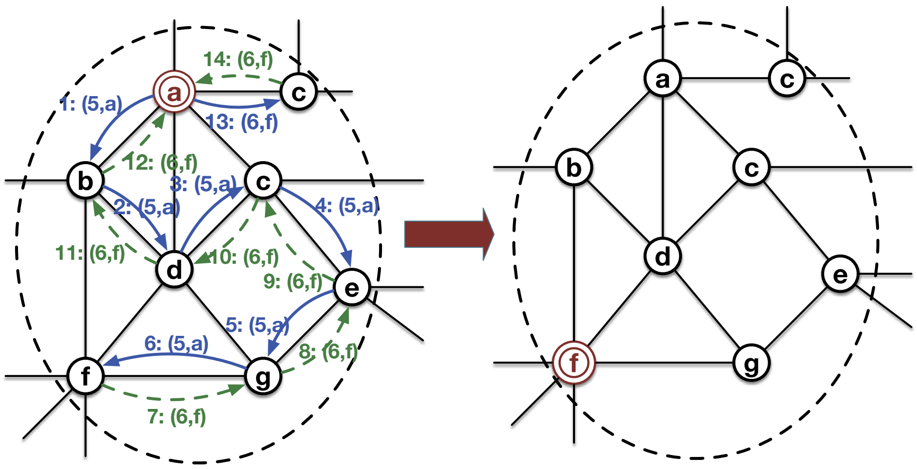


Fig. 5. Illustrative example for the head selection algorithm.

the head, that have had roles in performing task t_i. The profits of its head and each of non-head members are denoted by $\rho_{head_j^k}$ and $\rho_{\overline{head}_j^k}$, and defined as in Eqs. (12) and (13) respectively, where $\gamma \geqslant 1$ is the ratio of the head's payment to the payment of each non-head member.

$$\rho_{head_j^k} = \frac{\gamma \cdot py_i}{\eta_i^{j,k} + \gamma - 1} \tag{12}$$

$$\rho_{\overline{head}_j^k} = \frac{py_i}{\eta_i^{j,k} + \gamma - 1} \tag{13}$$

The profits of lower level holons are defined similarly using the profit of its super-holon. For instance, in order to calculate the profit of non-head holon H_l^{k-1}, at level $k-1$, which is the member of head holon H_j^k, at level k, one needs to compute the profit of the head holon, i.e., $\rho_{\overline{head}_j^k}$ using Eq. (13) first, and then use the result in Eq. (12) as $\rho_{head_j^{k-1}}$. The process is shown in Eq. (14).

$$\begin{aligned}\rho_{head_l^{k-1}} &= \frac{\gamma \cdot \rho_{\overline{head}_j^k}}{\eta_i^{j,k-1} + \gamma - 1} = \frac{\gamma \cdot \frac{py_i}{\eta_i^{j,k}+\gamma-1}}{\eta_i^{l,k-1} + \gamma - 1} \\ &= \frac{\gamma \cdot py_i}{\left(\eta_i^{j,k} + \gamma - 1\right)\left(\eta_i^{l,k-1} + \gamma - 1\right)}\end{aligned} \tag{14}$$

Using the same recursive process and $\gamma \geqslant 1$, one can easily find the range of the profit of every base holon H_x^0. Let us assume that holon H_j^k, the ancestor of holons H_x^0, is the successful initiator of task $t_i\langle id_i, TS_i, pt_i, ta_i, py_i\rangle$. The minimum (maximum) possible value for the profit of the base holon happens when holon H_j^k and all the super-holons on the path to holon H_x^0 are non-head (head) holons. Therefore the range is computed as in Eq. (15), where $m \in \{head, \overline{head}\}$.

$$\begin{aligned}py_i \cdot \left(\prod_{\lambda=0,y}^{k}\left(\eta_i^{y,\lambda} + \gamma - 1\right)\right)^{-1} &\leqslant \rho_{m_x^0} \\ &\leqslant \gamma^{k+1} \cdot py_i \cdot \left(\prod_{\lambda=0,y}^{k}\left(\eta_i^{y,\lambda} + \gamma - 1\right)\right)^{-1}\end{aligned} \tag{15}$$

Furthermore, since all the holons of our holarchy are disjoint at each level, in the sense that they do not share any member between each other, it can be inferred that for every performed task t_i we have:

$$\sum_x \rho_{m_x^0} = py_i \tag{16}$$

3.7. Learning-based reorganization

The ability to learn is essential for all intelligent systems. In multi-agent systems, two extremes for implementing this ability are the use of complete centralized and decentralized techniques. In centralized approaches, only one agent or a special controlling unit learns, based on detailed information about every parts of the system; while in decentralized models, all the agents of the system try to learn according to their experiences in interaction with the other agents in the environment. Although centralized methods are usually easier to be implemented and they may outperform the decentralized approaches in efficiency, they often fail to be used in large-scale open systems, especially where environment is partially observable. Among many learning algorithms in the artificial intelligence literature, Reinforcement Learning is one of the most common mechanisms in the design and deployment of multi-agent systems (Sen & Weiss, 1999). In this mechanism, learner agents receive rewards or punishments as feedbacks of their actions and behaviors in an environment.

In our work, we utilize a specific RL algorithm, namely stateless Q-learning (Claus & Boutilier, 1998), to enable the holons to adapt their structures dynamically at appropriate times. In other words, during their run time, the holons learn when to reorganize themselves, in terms of relationships and membership, and adapt to the changes occur in the system. The proposed learning-based reorganization process begins after the holonification step, when the initial holarchy is built distributedly as discussed before. During this process, the holons at every level of the holarchy monitor their performances in conducting the tasks, and change their interactions accordingly. Before, continuing to the proposed learning mechanism, detailed definition of terms "performance" is needed.

3.7.1. Performance

According to the concept of social facilitation in sociology, the performance of individuals is enhanced when they work in presence of others in both cooperative and competitive environments (Forsyth, 2009). The way this performance is measured depends on the domain the individuals are working in attendance of the others. For instance, in car racing domain the performance is measured in terms of speed, while in soccer the performance is often defined in terms of the number of scores.

Inspired from the concept of performance in organization theory, we asses the performance of the holarchy and holons in terms of effectiveness and efficiency (Bartuš evičienė & Šakalytė, 2013). In the context of our model for holonic multi-agent systems, these concepts are defined as follows:

3.7.1.1. Effectiveness. measures how successful a holon is in performing the introduced tasks. Formally, the effectiveness of holon $H_j^k, j > 0$ is defined as in Eq. (17), where $\left|T^{j,k}\right|$ denotes the total number of tasks assigned to holon H_j^k; and $\left|T_s^{j,k}\right|$ is the number of tasks that has been successfully completed by holon H_j^k. Please note that, $T_s^{j,k} \subseteq T^{j,k}$.

$$E_{\text{effectiveness}}(H_j^k) = \frac{\left|T_s^{j,k}\right|}{\left|T^{j,k}\right|} \tag{17}$$

Similarly, the effectiveness of entire holarchy $\langle H, HI\rangle$ can be defined as in Eq. (18), where $|T|$ and $|T_S|$ are respectively the number of all tasks introduced to the system, and the number of tasks that has been completed successfully by the holonic multi-agent system.

$$E_{\text{effectiveness}}(\langle H, HI\rangle) = \frac{|T_S|}{|T|} \tag{18}$$

3.7.1.2. Efficiency. Measures how a holon performs in completing the tasks, i.e., using the resources properly in performing the tasks. In other words, efficiency is about handling the costs of performing a task. In the proposed model, the cost is quantified in terms of the number of communications the holon and its sub-holons are involved to perform tasks. Let $T^{j,k}$ denote the set of all tasks assigned to holon H_j^k, $Com_t^{H_j^k}$ be the set of all communications made by H_j^k and its subordinate holons to perform tasks $t \in T^{j,k}$, and $|\ldots|$ be the set cardinality operator. The per-task efficiency of holon H_j^k is defined as follows:

$$E_{\text{efficiency}}(H_j^k) = \left[\frac{1}{|T^{j,k}|}\sum_{t\in T^{j,k}}\left|Com_t^{H_j^k}\right|\right]^{-1} \tag{19}$$

Similarly, for the efficiency of holarchy $\langle H, HI\rangle$ we have:

$$E_{\text{efficiency}}(\langle H, HI\rangle) = \left[\frac{1}{|T|}\sum_{t\in T}\left|Com_t^{H_{res}^h}\right|\right]^{-1} \tag{20}$$

where H_{res}^h is the responsible holon for task t in the top most level h of the holarchy.

Table 1 demonstrates the impact of each of these measures in our holonic multi-agent system model.

As it can be inferred, efficiency and effectiveness of a holonic multi-agent system, while being exclusive, influence each other at the same time. Defining performance based on these two measures, we have:

$$Performance = Effectivenes \times Efficiency \tag{21}$$

More precisely, the performance of any holon H_j^k and entire holarchy $\langle H, HI\rangle$ can be written respectively as:

$$P(H_j^k) = |T_s^{j,k}|\left[\sum_{t\in T^{j,k}}\left|Com_t^{H_j^k}\right|\right]^{-1} \tag{22}$$

and

$$P(\langle H, HI\rangle) = |T_S|\left[\sum_{t\in T}\left|Com_t^{H_{res}^k}\right|\right]^{-1} \tag{23}$$

3.7.2. Learning

The learning process of the proposed model is aimed at enabling the holarchy to adapt its structure dynamically over time. This adaptation is carried out through changes in holonic interactions, and hence holonic memberships, as new tasks arrive and holons receive feedbacks about their connections and standings in the holarchy.

In our HMAS model, we assume the holons as independent learners who ignore the existence of the other holons and employ classic Q-learning to learn the values of their own actions. As stated before, our specific algorithm for the learner holons is based on stateless Q-learning (Claus & Boutilier, 1998) method in which each holon H_j^k updates the Q-values of its actions over time using Eq. (24), where a denotes the action that the holon is updating the Q-value for; λ is the learning rate of the algorithm such that $0 \leqslant \lambda \leqslant 1$; and r is the reward that the holon receives for performing action a. In this algorithm, the experience of the holons can be encoded by the values of $\langle a, r\rangle$ over time.

$$Q^{j,k}(a) \leftarrow Q^{j,k}(a) + \lambda\left(r - Q^{j,k}(a)\right) \tag{24}$$

The above-mentioned learning method needs the set of actions that holons can play and the way their corresponding rewards are estimated, be clearly specified. In our model, holons at any time can perform any of actions in $\{change, preserve\}$. By performing action preserve, holons maintain their current memberships and interactions as they are. On the other hand, by action change, holons try to change their position in the holonic network and possibly their holonic memberships. For this purpose, holons perform three sub-actions $\{break, make, decide\}$, in the order of starting from break and ending to decide. By sub-action break, a holon breaks its relationship with one of its immediate members, by removing the corresponding holonic link. This breaking action is followed by making a new relationship and starting a new interaction link. Finally based on the new relationships the holon decides about changing its current membership. It should be noted that during the learning process, holon learns merely the aforementioned two main actions, without considering any Q-values for the sub-actions.

Regarding reward parameter r, whenever holon H_j^k updates it Q-values, it computes the reward based on the

Table 1
Effectiveness vs. efficiency in the proposed model.

	Efficient	Inefficient
Effective	HMAS succeeds to complete a **large** number of introduced tasks, with the **lowest** per-task cost	HMAS succeeds to complete a **large** number of introduced tasks, however the per-task cost is **high**
Ineffective	HMAS succeeds to complete a **small** number of introduced tasks, with the **lowest** per-task cost	HMAS succeeds to complete a **small** number of introduced tasks, and the per-task cost is **high**

change in its effectiveness and efficiency as presented in Eq. (25), where $\Delta E_{Efficiency}(H_j^k)$ and $\Delta E_{Effectivness}(H_j^k)$ are the changes occurred in holon H_j^k's efficiency and effectiveness (Eqs. (17) and (19)) because of performing the most recent action.

$$r = \Delta E_{Efficiency}(H_j^k) + \Delta E_{Effectivness}(H_j^k) \tag{25}$$

As for the learning rate, we have employed the *Win or Lose Fast (WoLF)* method introduced in Bowling and Veloso (2002) to make the holons learn fast when their situations are not beneficial (changes in either effectiveness or efficiency of the holon is negative), and learn slowly otherwise. The learning process for the action values of holon H_j^k is used in every pre-specified episodes.

3.7.3. Reorganization

The reorganization process is triggered according to the learned Q-values for the actions. In other words, the reorganization process of every holon H_j^k is triggered when $Q(change) > Q(preserve)$. The detailed procedure is demonstrated in Algorithm 6.

Algorithm 6. Reorganization algorithm

```
1: procedure REORGANIZE    ▷ This function is run by an individual holon
2:     $action \leftarrow \arg\max_a Q(a)$
3:     if $action = change$ then
4:         $mvn \leftarrow$ MINVALUEDNEIGHBOR()
5:         $prob \leftarrow \frac{|\text{SKILLS}(self) \cap \text{SKILLS}(mvn)|}{|\text{SKILLS}(self)|}$
6:         $nn \leftarrow mvn$.MAXVALUEDNEIGHBOR() with probability $prob$ or $nn \leftarrow mvn$.MINVALUEDNEIGHBOR() with probability $(1 - prob)$
7:         if MAKE($nn$)= $SUCCESSFUL$ then
8:             BREAK($mvn$)
9:     end if
10:    $cl \leftarrow$ CL(), $cla \leftarrow$ CLALT()
11:    if max($cla$) > $cl$ then
12:            $ns \leftarrow \arg\max_{superholon} cla$
13:            if APPLYTO($ns$) = $ADMITTED$ then
14:                CHECKOUT($superholon$)
15:                CHECKIN($ns$)
16:            end if
17:        end if
18:    end if
19: end procedure
```

The proposed reorganization algorithm is inspired from the social exchange theory in social psychology (Emerson, 1976). According to this theory, social relationships of humans are formed and altered according to a subjective cost-benefit analysis (CBA). Base on line 2 of Algorithm 6, a reorganizing holon chooses the action that has the highest Q-value, learned so far. If the selected action is "change" then the holon tries to make changes in its relationships. This is done through choosing the most inferior neighbor and substituting the interaction with it, with one of its referrals. The most inferior neighbor, is the immediate connected holon that has directed beneficial task assignment requests.The heuristic that we use to select the inferior neighbor is based on the value that holons assign to their interaction links during time. Based on social exchange theory, holon H_i^k updates the value of its interaction with neighbor holon H_j^k as in Eq. (26), where v_{ij}^k denotes the interaction value; $0 \leqslant \alpha_m \leqslant 1$ is the impact coefficient to control the influence of inner $(m = inner)$ and outer holon $(m = outer)$ relationships; and $benefit_{ij}^k$ and $cost_{ij}^k$ are the quantitative benefits and costs of keeping the relationship respectively.

$$v_{ij}^k \leftarrow \alpha_m \left(benefit_{ij}^k - cost_{ij}^k\right) \tag{26}$$

The precise definitions of these parameters depend on the application domain in which the model is used. In this article, we define benefit based on the positive interactions (interactions that possibly result in the completion of a task) between two holons and cost based on the negative relationships (relationships that waste the resources). Lines 4 of Algorithm 6 finds the most inferior neighbor as the one connected with the lowest interaction value. The new candidate holon to make relationship is the one referred by that previously selected least valued neighbor. The referred neighbor is chosen based on the portion of skills the reorganizing holon and its inferior neighbor share (lines 5 and 6 of Algorithm 6). This way the reorganizing holons chance to get beneficial relationships is icreased. Finally, if the new connection is made successfully the least valued link is broken. It should be noted that the break operation disconnects the relationship with a holon merely if this disconnections does not breaks the reorganizing holons relation with its current superholon.

Change of holonic membership is defined base on the concepts of Comparison Level (CL) and Comparison Level for alternatives (CLalt) in social exchange theory (Forsyth, 2009). CL is the standard by which a holon evaluates membership to other holons (if any), and CLalt is the standard by which holons evaluate the quality of other holons that they can join. Again, these standards should be defined according to the application domain. In the task environment of this paper, we simply use the values a holon assigns to its current relationships. If a holon finds any new superholon for witch $CLalt > CL$, it tries to change his super holon. If the new super-holon accepts the new membership, holon implements the transfer procedure through $Check - in$ and $Check - out$ procedures (lines 11–15 of Algorithm 6). Please note that, all these procedures are run in compliance with the social norms we defined in the holonification section.

In addition to changing relationships and memberships, the composite holon may also dissolve the holon through a decomposition procedure. This case happens when the

Table 2
General configuration of the experimented multi-agent networks.

Property	$\lvert A\rvert = \lvert H^0\rvert$	$\lvert HS_i^0\rvert$	σ	$\lvert TS_i\rvert$	$\lvert T\rvert$
Value	150	1	15	15	15,000
Property	γ	λ_{reward}	$\lambda_{punishment}$	α_{inner}	α_{outer}
Value	1.5	0.1	0.6	0.5	0.5

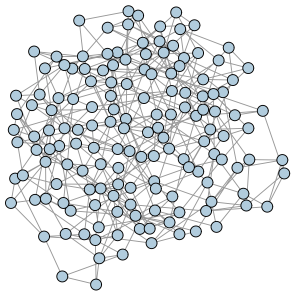
performance of the holon continues to decrease for a specific number of reorganization process. When a holon is dissolved, all of its members leave their container holons and become standalone again.

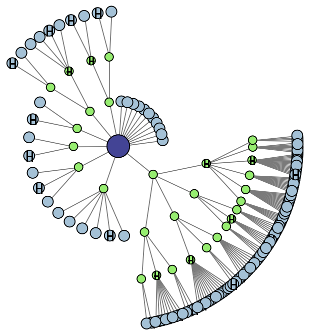
4. Results and discussion

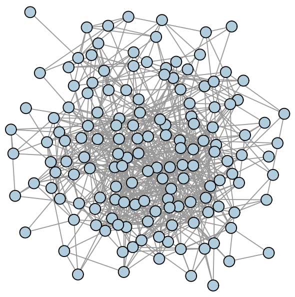
In order to assess our algorithm, we applied it to a set of multi-agent networks generated according to Kleinberg's small-world network model (Easley & Kleinberg, 2010). These networks consist of 150 agents. The general configuration of the MAS systems, used in evaluations, is given in Table 2. As stated in the table, each agent has a single skill ($|HS_i^0| = 1$), and there are 15 distinct skills defined in the system ($\sigma = 15$); the total number of tasks that enter the system is 15,000 ($|T| = 15000$), each of which has a required skill set of size 15 ($|TS_i| = 15$); the ratio of the head's payment to the payment of each non-head member is 1.5 ($\gamma = 1.5$); the learning rates for reward and punishment are respectively 0.1 ($\lambda_{reward} = 0.1$) and 0.6 ($\lambda_{punishment} = 0.6$); and finally, the impact coefficient for both inner and outer holon relationships is 0.5 ($\alpha_{inner} = 0.5$ and $\alpha_{outer} = 0.5$).

Furthermore, we have assumed that the lengths of tasks, in terms of their required skills, are fixed; the processing time of the tasks are chosen randomly from set $\{3, 4, 5\}$; and finally the minimum and maximum age that the tasks can have are 1 and 2 respectively. This choice of task ages creates a heavily loaded task environment for our experiments. All data presented in this study are averaged over 30 experiments, each consists of the creation of a new multi-agent network and performing the proposed algorithms on it, based on the configurations of Table 2.

The framework is developed in Java programming language, and the holonic networks together with the built holarchy are carefully observed during the experiments. Fig. 6 demonstrates a multi-agent network together with its holonic structure after applying the initial holonification algorithm, in one of the experiments conducted in this study. It also shows the final states of the corresponding holonic network, and the holarchy at the end of the simulation.


(a) Initial multi-agent network.


(b) Initial hierarchical view of the built holarchy.


(c) Multi-agent network at the end of simulation.

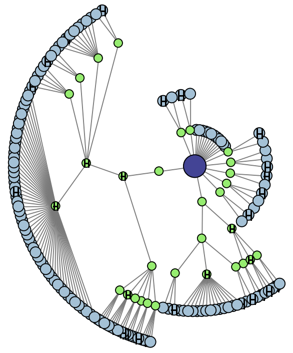

(d) Holarchy at the end of dynamic reorganization.

Fig. 6. A holonification experiment.

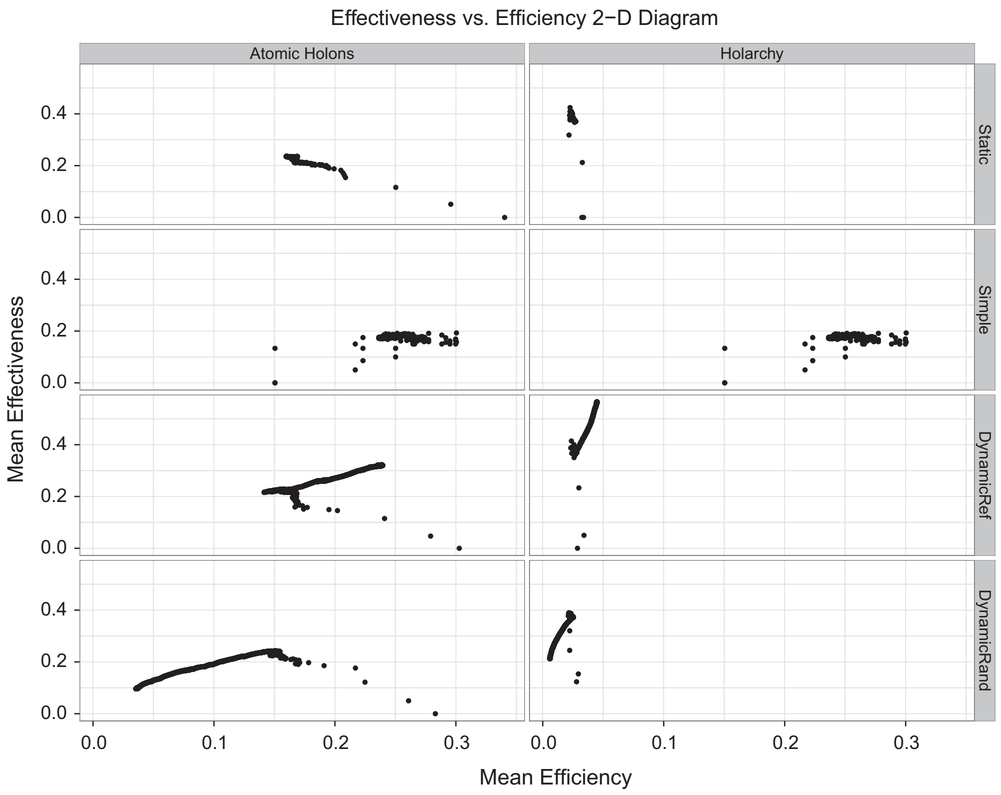


Fig. 7. Effectiveness vs. efficiency of various algorithms in two-dimensional view.

For simplicity and clarity, the holonic structure is demonstrated in radial tree like layout. In this holarchy, the leaves are atomic holons (agents) and the intermediate light green nodes are the composite holons of higher levels. Further more, the head holons of every composite holon, are specified by letter "H" inside the corresponding node. Please note that, the intra-level interaction links are omitted in this view of the holarchy, and only inter-level links are shown.

The empirical results presented in this section, show how the proposed framework behaves at different stages. For this purpose, in addition to our proposed distributed dynamic holonification algorithm (DynamicRef), we have implemented and tested three other algorithms: Dynamic Random holonification (DynamicRand), Static Holonification (Static), and Simple Non-Holonic (Simple) algorithms. In the dynamic random algorithm the holarchy is built using the same holonification algorithm as the proposed model and it benefits the same learning procedure to adapt the holonic network connections. However, instead of using a referral method to change the connections, the holons randomly choose the peers to disconnect from and the ones to connect to. The static holonification algorithm lacks the dynamic reorganization part of our proposed model and is used to assess the effect of dynamic adaption of the structure. Finally, in order to prove the capabilities of holonification and holonic structures, we have made use of a simple non-holonic algorithm. In this algorithm the agents try to perform the tasks based on their own capabilities and the ones of their immediate neighbors.

The empirical results presented in this section are based on the values obtained from 30 different runs over 30 different networks. In order to provide a fair condition for the comparisons, the above-mentioned algorithms are evaluated on same multi-agent network and same task sets introduced in same order in each run.

The first set of experiments concentrates on effectiveness and efficiency measures as defined in previous section. In order to demonstrate how these factors change together, we have plotted the values of effectiveness vs. the values of efficiency. In addition, in order to see how these parameters change over time, we have provided a 3-dimensional view for the plots. Please note that each point in the plots represents the average value of the corresponding measures in 30 different runs. The 2-D and 3-D view of the effectiveness vs. efficiency plots for the atomic holons and the entire holarchy, under various algorithms, are depicted in Figs. 7 and 8 respectively.

According to the demonstrated results, the average efficiency of the atomic holons is significantly higher (about two times) than that of the entire holarchy, in each experimented framework. This is due to the additional costs of inter-level communications in the holarchy. In addition, among all four algorithms, the simple non-holonic one provided the best efficiency, thanks to its non-hierarchical structure. However, the cost of such high efficiency is the

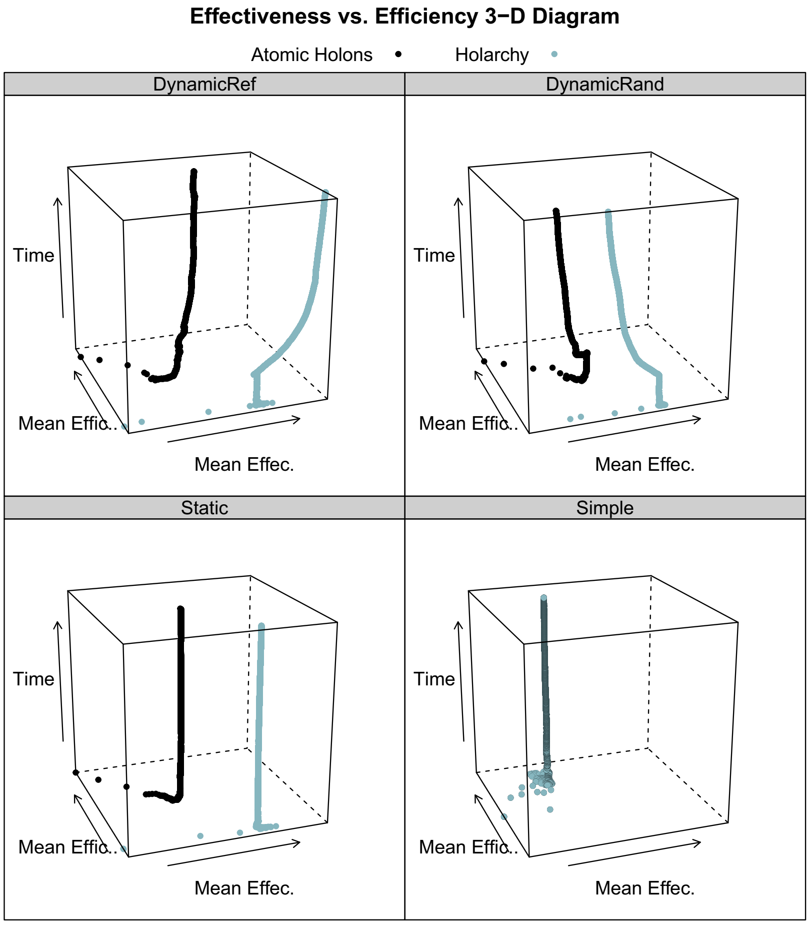


Fig. 8. Effectiveness vs. efficiency of the algorithms during run time in three-dimensional view.

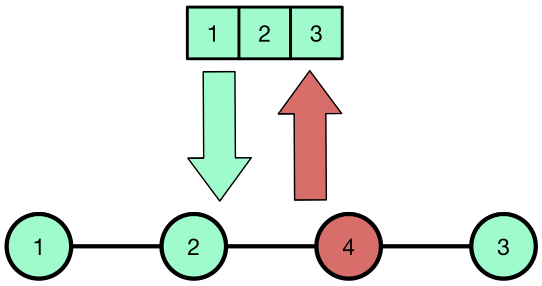


Fig. 9. A simple example illustrating situations that simple non-holonic algorithm fails to perform an arrived task.

low level of its effectiveness in comparison to almost all other algorithms, which is due to its incapability in utilizing the skills of agents who are not in proper vicinity of each other. This incapability is shown through the simple illustrative example of Fig. 9. As it can bee seen, when a task requiring skills $\{1,2,3\}$ arrives, though the system own all the needed skills, it fails to perform the task because of the lack of any link between agents 3 and any of agents 1 or 2.

As it can be seen in Figs. 7 and 8, all holonic approaches managed to overcome the effectiveness problem through their multi-level structure. The gained improvements in the holonification phases of the algorithms were as much as 120% in effectiveness. However, this improvement has been accompanied by a decrease in efficiency of the systems (almost 33% decrease in mean efficiency of the atomic holons and 75% in that of entire holarchy). It should be noted that the similarity in the behaviors of three holonic approaches, in terms of effectiveness and efficiency measures, during the holonification stage, is due to the fact that they benefit the same holonification algorithm until the reorganization step starts. The empirical results prove that reorganization of the holarchy plays an important role in its performance. As with our proposed referral based reorganization algorithm, we have witnessed an even more increase in average effectiveness (33% for holarchy and 37% for atomic holons) in comparison to static method. On the other hand, we observed that, applying a random approach in rewiring the holonic network of the system (as used in *DyanmicRand* method), downgraded both effectiveness and efficiency of system in comparison to the static holonic approach. In terms of effectiveness this decrease was 59% for atomic holons and 70% for the entire holarchy. Furthermore, since improper reconnection of the holons results in more unsuccessful communications, a significant decrease in efficiency was also observed. This is while the proposed referral besed dynamic algorithm has

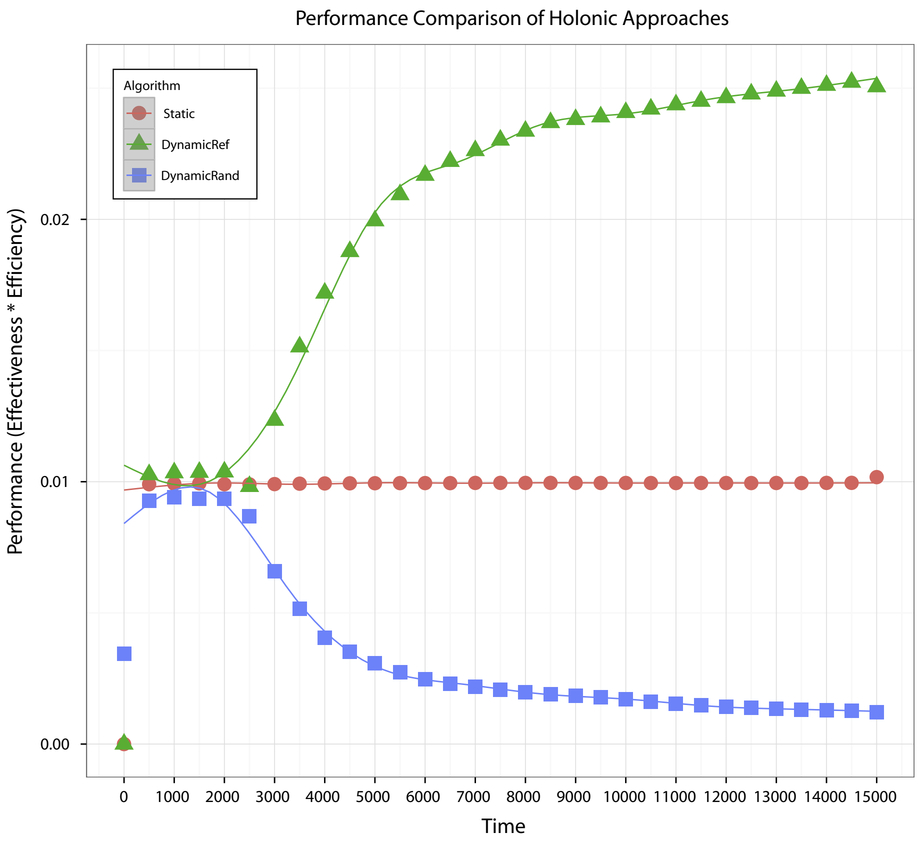


Fig. 10. Performance of holonic approaches based on their effectiveness and efficiency.

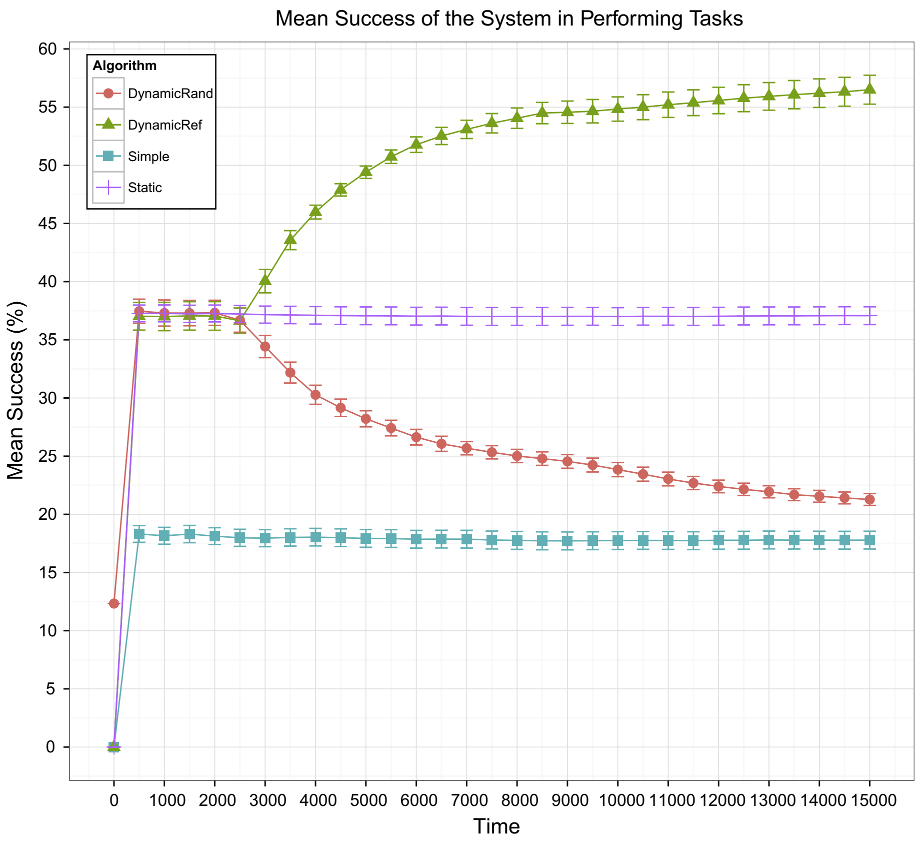


Fig. 11. Average success of the algorithms in performing the tasks during time.

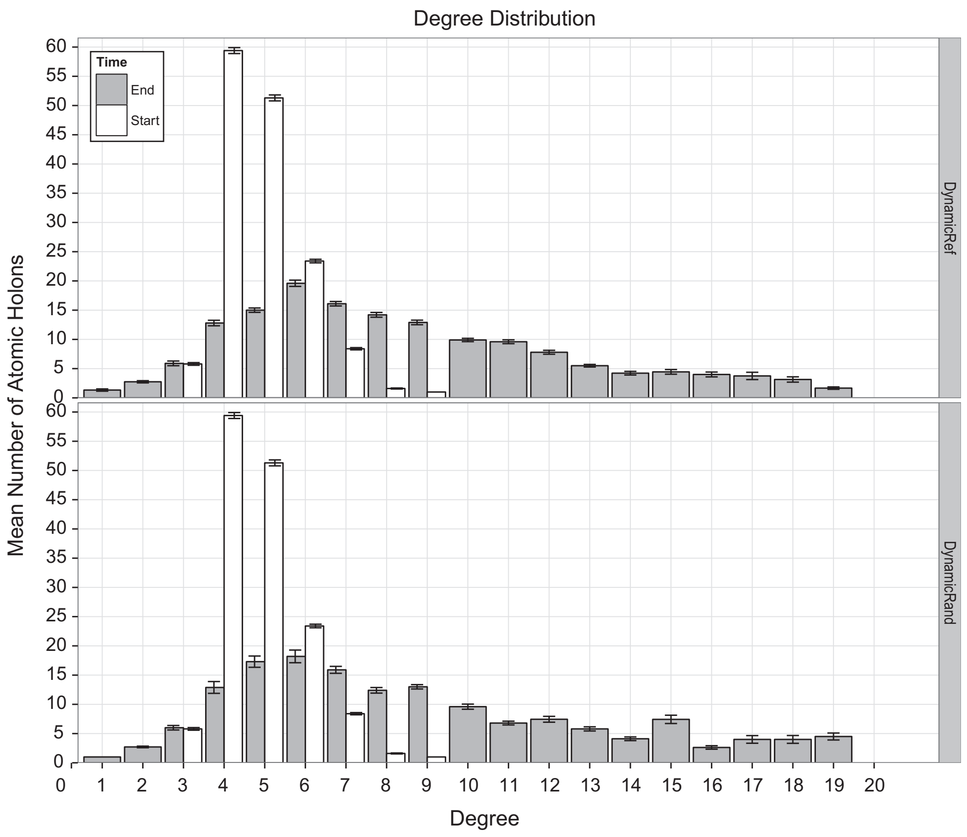


Fig. 12. Average degree distribution of the multi-agent networks at the start and end of reorganization process under two dynamic holonification algorithms.

slightly improved the efficiency for both atomic holons and the holarchy though being still less than the efficiency of the simple non-hierarchical method. In addition to effectiveness and efficiency metrics, we have also demonstrated the behaviors of the holonic algorithms in terms of their performances (as defined in Eqs. (22) and (23)) and the success of all four methods in conducting the introduced tasks. We evaluate the percentage of the success of the algorithms at time t using Eq. (27), where $|Comp\_Tasks_{\leqslant t}|$ denotes the number of completed tasks by the system until time t; and $|Total\_Tasks_{\leqslant t}|$ is the total number of the tasks introduced to the system by time t.

$$Success_t = \left(\frac{|Comp\_Tasks_{\leqslant t}|}{|Total\_Tasks_{\leqslant t}|}\right) \times 100 \tag{27}$$

The performance and success diagrams are presented in Figs. 10 and 11 respectively. Please note that, the results provided in these diagrams are the average results obtained from 30 different runs.

As it can be seen from performance and success diagrams, the proposed referral-based dynamic holonic model(DynamicRef) outperforms the other algorithms in terms of the number of completed tasks, and how effectively and efficiently it executes the tasks. Please note that the performance diagram lacks the simple non-holonic method due to the fact that the metric we have defined for the performance of holonic algorithms are not applicable to non-holonic methods.

Apart from the performance measures, we have been interested in the way that dynamic reorganization affects the network topologies and the changes in sizes of the holons. For this purpose, we have presented the degree distribution of the multi-agent network and holon size distribution of the holarchy, both at the beginning and the end of the reorganization process. These diagrams are depicted respectively in Figs. 12 and 13. As it can bee seen, the dynamic reorganization process has increased the average degree of networks in both of the algorithms though the number of holons with higher degrees is slightly higher in dynamic random method. The degree distribution diagrams show that in order to improve the performance of our holonic multi-agent system model, it is much more important to whom the holons are connected than how many connections they made. Regarding the distribution of the mean sizes of the composite holons, the empirical results show that the reorganization procedure has resulted in the creation of small size holons (of two and three) and reduction in the number of holons with large number of members in some cases. On the other hand, except a small difference in the number of holons with small size, the presented dynamic algorithms differ largely in the distribution of holons of sizes greater than 5. Please note that the dynamic random method studied in this comparison differs the dynamic referral algorithm merely in the rewiring procedure. That is the reason we observe similarities in their degree and holon sizes distributions.

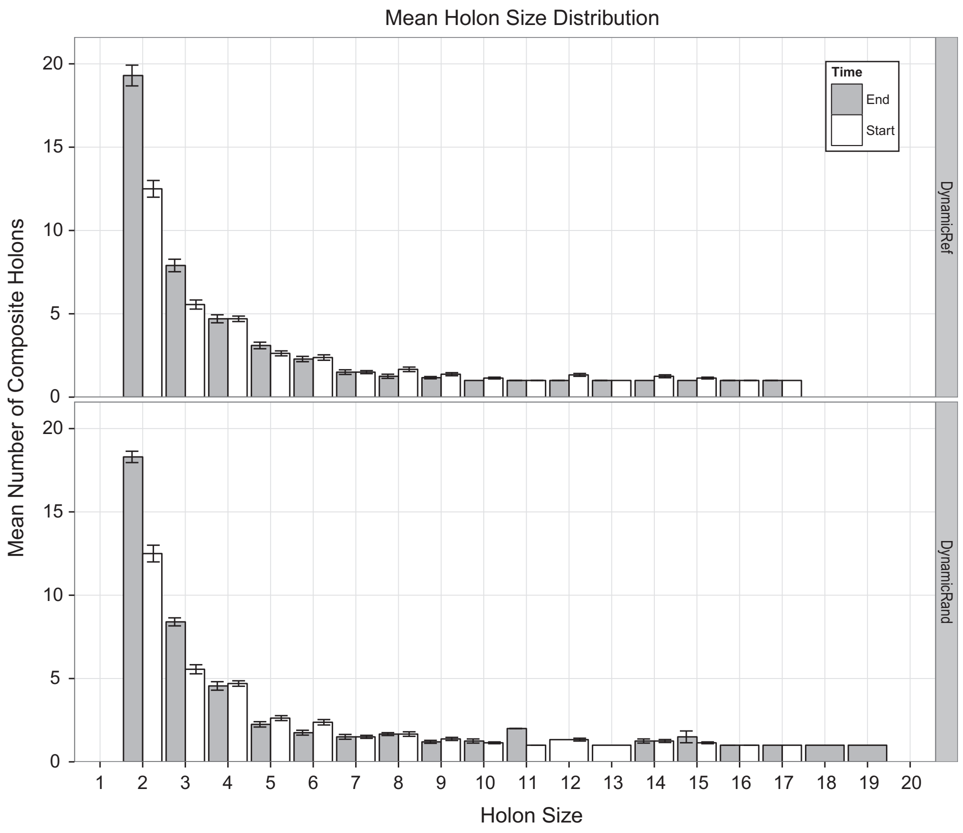


Fig. 13. Average size distribution of the composite holons at the start and end of reorganization process under two dynamic holonification algorithms.

5. Conclusions

This paper presented a novel distributed and dynamic model for holonic multi-agent systems. The proposed model assumes an interaction networks among the agents of a multi-agent system and constructs the holonic structure in a bottom-up approach, which has adopted many concepts from human social sciences, such as diversity, norms, social choices, social exchange theory, and organizational performance measures. In this approach, the holons do not need any extra system level information to build and maintain the holonic structures. This decentralized and local-info-based nature of the suggested model makes it an appropriate approach for large-scale open multi-agent systems. The proposed holonification algorithm is expected to be general enough to be employed in broad ranges of applications. In this article, we have used task environment to show not only how the proposed model works, but also how it performs and improves the performance of the system. The task environment deployed in this paper can be configured in many ways to be as challenging as needed. In addition to the dynamic holonification algorithm, this paper also presents contributions in the way a holonic multi-agent system can be employed in a task environment.

Both holonification and reorganization phases of the proposed model were assessed in a highly loaded task assignment application. The evaluations were carried out in terms of performance, effectiveness, and efficiency of the model, and the obtained results were fairly compared with the ones achieved from three other algorithms. According to the empirical results, the holonification algorithm significantly improved the success rate of the system in dealing with the tasks introduced. The only drawback of our proposed method was a decrease in efficiency, which caused by extra inter and intra holonic communication costs. However, such a decrease in efficiency was accompanied by a considerable increase in effectiveness and success of the entire system. Furthermore, the results obtained about the interaction networks and the distribution of the sized of the holons demonstrated that the proper reconnection of the holons during the reorganization phase plays the most important role in the success of our proposed dynamic holonification algorithm.

The dynamic distributed holonic model presented in this paper can be further enhanced. Having the ambitious goal of making artificial societies as close as possible to human societies in mind, some of these improvements include: dealing with dynamic changes in the skills of the holons, managing fatal events occurred in the system, defining a proper ontology for the holonic interactions, and employing emerging social norms. We are currently working on some parts of these ideas, and suggest them together with applying the proposed model to other large scaled multi-agent applications as future works.

Acknowledgments

The authors would like to appreciate the very helpful comments of the anonymous reviewers on improving the quality of this paper.

References

Abdoos, M., Esmaeili, A., & Mozayani, N. (2012). Holonification of a network of agents based on graph theory. In G. Jezic, M. Kusek, N.-T. Nguyen, R. Howlett, & L. Jain (Eds.), *Agent and multi-agent systems: Technologies and applications SE – 42* (pp. 379–388). Berlin, Heidelberg: Springer.

Abdoos, M., Mozayani, N., & Bazzan, A. L. C. (2013). Holonic multi-agent system for traffic signals control. *Engineering Applications of Artificial Intelligence, 26*(56), 1575–1587.

Adam, E., & Mandiau, R. (2003). Bringing multi-agent systems into human organizations: Application to a multi-agent information system. In M. Jeusfeld & Ó. Pastor (Eds.), *Conceptual modeling for novel application domains SE – 17. Lecture notes in computer science* (Vol. 2814, pp. 168–179). Berlin, Heidelberg: Springer.

Barbosa, J. (2015). Self-organized and evolvable holonic architecture for manufacturing control. Ph.D. thesis, Université de Valenciennes et du Hainaut Cambrésis.

Barbosa, J., Leitão, P., Adam, E., & Trentesaux, D. (2013). Structural self-organized holonic multi-agent manufacturing systems. In V. Maík, J. Lastra, & P. Skobelev (Eds.), *Industrial applications of holonic and multi-agent systems SE – 6. Lecture notes in computer science* (Vol. 8062, pp. 59–70). Berlin, Heidelberg: Springer.

Bartuševičienė, I., & Šakalytė, E. (2013). *Organizational assessment: Effectiveness vs. efficiency*. Social Transformations in Contemporary Society, 1.

Bowling, M., & Veloso, M. (2002). Multiagent learning using a variable learning rate. *Artificial Intelligence, 136*(2), 215–250.

Carley, K. M., & Gasser, L. (1999). Computational organization theory. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, 299–330.

Ciufudean, C., & Filote, C. (2011). Artificial social models for holonic systems. In *Holonic and multi-agent systems for manufacturing* (pp. 133–142). Springer.

Claus, C., & Boutilier, C. (1998). The dynamics of reinforcement learning in cooperative multiagent systems. In *AAAI/IAAI* (pp. 746–752).

Clegg, B. T. (2007). Building a holarchy using business process-oriented holonic (proh) modeling. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 37*(1), 23–40.

DeLoach, S. A., & Matson, E. (2004). An organizational model for designing adaptive multiagent systems. In *The AAAI-04 workshop on agent organizations: Theory and practice (AOTP 2004)* (pp. 66–73).

DeLoach, S. A., Wood, M. F., & Sparkman, C. H. (2001). Multiagent systems engineering. *International Journal of Software Engineering and Knowledge Engineering, 11*(03), 231–258.

Dignum, V., Meyer, J. -J., Weigand, H., & Dignum, F. (2002). An organizational-oriented model for agent societies. In *Proc. int. workshop on regulated agent-based social systems: Theories and applications (RASTA'02)*, at AAMAS, Bologna, Italy.

Dos Santos, D. S., & Bazzan, A. L. (2012). Distributed clustering for group formation and task allocation in multiagent systems: A swarm intelligence approach. *Applied Soft Computing, 12*(8), 2123–2131.

Easley, D., & Kleinberg, J. (2010). *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press.

Emerson, R. M. (1976). Social exchange theory. *Annual Review of Sociology*, 335–362.

Esmaeili, A., Mozayani, N., & Jahed Motlagh, M. R. (2014). Multi-level holonification of multi-agent networks. In *12th Iranian conference on intelligent systems (ICIS 2014)* (pp. 1269–1273). Bam: IEEE.

Esmaeili, A., Mozayani, N., Motlagh, M. R. J., & Matson, E. T. (2016). The impact of diversity on performance of holonic multi-agent systems. *Engineering Applications of Artificial Intelligence, 55*, 186–201.

Esteva, M., Rodriguez-Aguilar, J.-A., Sierra, C., Garcia, P., & Arcos, J. L. (2001). On the formal specification of electronic institutions. In *Agent mediated electronic commerce* (pp. 126–147). Springer.

Ferber, J., Gutknecht, O., & Michel, F. (2003). From agents to organizations: An organizational view of multi-agent systems. In *International workshop on agent-oriented software engineering* (pp. 214–230). Springer.

Ferber, J., Gutknecht, O., & Michel, F. (2004). From agents to organizations: An organizational view of multi-agent systems. In *Agent-oriented software engineering IV* (pp. 214–230). Springer.

Forsyth, D. (2009). *Group dynamics*. Cengage Learning.

Gaston, M., & DesJardins, M. (2005). Agent-organized networks for dynamic team formation.

Gerber, C., Siekmann, J., & Vierke, G. (1999). Holonic multi-agent systems. Tech. rep. 681, German Research Center for Artificial Intelligence (DFKI), Kaiserslautern.

Hannoun, M., Boissier, O., Sichman, J. S., & Sayettat, C. (2000). Moise: An organizational model for multi-agent systems. In *Advances in artificial intelligence* (pp. 156–165). Springer.

Hayden, S. C., Carrick, C., & Yang, Q. (1999). A catalog of agent coordination patterns. In *Proceedings of the third annual conference on autonomous agents* (pp. 412–413). ACM.

Hilaire, V., Koukam, A., Gruer, P., & Müller, J.-P. (2000). Formal specification and prototyping of multi-agent systems. In *Engineering societies in the agents world* (pp. 114–127). Springer.

Hilaire, V., Koukam, A., & Rodriguez, S. (2008). An adaptative agent architecture for holonic multi-agent systems. *ACM Transactions on Autonomous and Adaptive Systems (TAAS), 3*(1), 2.

Horling, B., & Lesser, V. (2004). A survey of multi-agent organizational paradigms. *The Knowledge Engineering Review, 19*(4), 281–316.

Hübner, J. F., Sichman, J. S., & Boissier, O. (2002). Moise+: Towards a structural, functional, and deontic model for mas organization. In *Proceedings of the first international joint conference on autonomous agents and multiagent systems: Part 1* (pp. 501–502). ACM.

Irandoust, H., & Benaskeur, A. R. (2008). Multi-organizational structures. In *Proc. of association for the advancement of artificial intelligence, Chicago* (pp. 25–33).

Jie, L., Wei-Ming, Z., Bao-Xin, X., & Zhong, L. (2011). An organization model in MAS based on holon. In *2011 IEEE ninth international conference on dependable autonomic and secure computing* (pp. 951–957).

Koestler, A. (1968). *The ghost in the machine*. Macmillan.

Kota, R., Gibbins, N., & Jennings, N. R. (2012). Decentralized approaches for self-adaptation in agent organizations. *ACM Transactions on Autonomous and Adaptive Systems (TAAS), 7*(1), 1.

Leitão, P., & Restivo, F. (2006). Adacor: A holonic architecture for agile and adaptive manufacturing control. *Computers in Industry, 57*(2), 121–130.

Leitao, P., & Restivo, F. J. (2008). Implementation of a holonic control system in a flexible manufacturing system. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 38* (5), 699–709.

Marcellino, F. J. M., & Sichman, J. S. (2010). *A holonic multi-agent model for oil industry supply chain management*. Berlin, Heidelberg: Springer, pp. 244–253.

Mullen, B. (1987). Self-attention theory: The effects of group composition on the individual. In *Theories of group behavior* (pp. 125–146). Springer.

Rodriguez, S., Gaud, N., Hilaire, V., Galland, S., & Koukam, A. (2007a). An analysis and design concept for self-organization in holonic multi-agent systems. In *Engineering self-organising systems* (pp. 15–27). Springer.

Rodriguez, S., Hilaire, V., & Koukam, A. (2005). Formal specification of holonic multi-agent systems framework. In *Computational science–ICCS 2005* (pp. 719–726). Springer.

Rodriguez, S., Hilaire, V., & Koukam, A. (2007b). Towards a holonic multiple aspect analysis and modeling approach for complex systems: Application to the simulation of industrial plants. *Simulation Modelling Practice and Theory, 15*(5), 521–543.

Scerri, P., Farinelli, A., Okamoto, S., & Tambe, M. (2005). Allocating tasks in extreme teams. In *Proceedings of the fourth international joint conference on autonomous agents and multiagent systems* (pp. 727–734). ACM.

Sen, S., & Weiss, G. (1999). *Multiagent systems*. Cambridge, MA, USA: MIT Press, Chapter: Learning in multiagent systems (pp. 259–298).

Shannon, C. (1949). *The mathematical theory of communication*. Urbana: University of Illinois Press.

Sims, M., Corkill, D., & Lesser, V. (2008). Automated organization design for multi-agent systems. *Autonomous Agents and Multi-Agent Systems, 16*(2), 151–185.

Stefanoiu, D., Ulieru, M., & Norrie, D. (2000). Fuzzy modeling of multi-agent systems behavior: Vagueness minimization. In *Proceedings of world multiconference on systemics, cybernetics and informatics (SCI'2000)*. Citeseer (Vol. 3, pp. 118–123).

Ulieru, M. (2002). Emergence of holonic enterprises from multi-agent systems: A fuzzy evolutionary approach. *Soft computing agents: A new perspective on dynamic information systems*, 187–215.

Ulieru, M., & Geras, A. (2002). Emergent holarchies for e-health applications: A case in glaucoma diagnosis. *IEEE 2002 28th annual conference of the industrial electronics society [IECON 02]* (Vol. 4, pp. 2957–2961). IEEE.

Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., & Peeters, P. (1998). Reference architecture for holonic manufacturing systems: Prosa. *Computers in Industry, 37*(3), 255–274.

Weiss, G. (Ed.). (2013). *Multiagent systems* (2nd ed.). MIT Press.

Wooldridge, M. (2009). *An introduction to multiagent systems*. John Wiley & Sons.

Zhang, X., & Norrie, D. H. (1999). Holonic control at the production and controller levels. In *Proceedings of the 2nd international workshop on intelligent manufacturing systems*. Citeseer (pp. 215–224).