Multi-Agents Collaboration in Open System

Zina Houhamdi[1] and Belkacem Athamena[2]
[1]Software Engineering Department, College of Engineering, Al Ain University, UAE
[2]Business Administration Department, College of Business, Al Ain University, UAE

Abstract: *Share constrained resources, accomplish complex tasks and achieve shared or individual goals are examples requiring collaboration between agents in multi-agent systems. The collaboration necessitates an effective team composed of a set of agents that do not have conflicting goals and express their willingness to cooperate. In such a team, the complex task is split into simple tasks, and each agent performs its assigned task to contribute to the fulfilment of the complex task. Nevertheless, team formation is challenging, especially in an open system that consists of self-interested agents performing tasks to achieve several simultaneous goals, usually clashing, by sharing constrained resources. The clashing goals obstruct the collaboration's success since the self-interested agent prefers its individual goals to the team's shared goal. In open systems, the collaboration team construction process is impacted by the Multi-Agent System (MAS) model, the collaboration's target, and dependencies between agents' goals. This study investigates how to allow agents to build collaborative teams to realize a set of goals concurrently in open systems with constrained resources. This paper proposes a fully distributed approach to model the Collaborative Team Construction Model (CTCM). CTCM modifies the social reasoning model to allow agents to achieve their individual and shared goals concurrently by sharing resources in an open MAS by constructing collaborative teams. Each agent shares partial information (to preserve privacy) and models its goal relationships. The proposed team construction approach supports a distributed decision-making process. In CTCM, the agent adapts its self-interest level and adjusts its willingness to form an effective collaborative team.*

Keywords: *Multi-agents system; open system; collaboration, dependency relationships, decentralized decision making*.

Received February 20, 2021; *accepted March 7, 2021*
https://doi.org/10.34028/iajit/18/3A/2

1. Introduction

A survey of current methods in collaborative team construction, particularly in an open system based on MAS identifies several shortcomings:

- The existing methods design the system as a single neighborhood system, multi-disjoint neighborhoods, or multi-overlapped neighborhoods with transitive dependencies. These methods are expensive due to the interaction cost and constraints of the possible collaborative teams that can be constructed due to the open system properties
- The existing methods model the agents' goals as local or common exclusively during the collaborative teams' formation.
- The existing methods also model the agent as cooperative and self-interested. Nevertheless, in an open system, the self-interested agent should balance between its self-interest level and the collaboration to fulfill several goals concurrently, especially because it accesses to only restricted domain information and shares constrained resources.
- The existing methods also model the decision-making process of the agents is semi-decentralized and force the agents to remain in the Multi-Agent System (MAS) throughout the collaboration process. However, this is limited in an open system in which each agent is allowed to join or leave the system in an unpredictable way without any time constraints.

This paper extends the proposed Collaborative Team Construction Model (CTCM) presented in [13] by describing the detailed algorithm of each phase of the model. It addresses the design purposes of CTCM. The article describes the CTCM design and demonstrates how the proposed model meets the Collaborative Team Construction requirements in open MAS. Note that the main purposes of this study are to model and develop a collaborative team construction model that enables agents: acting in multi-overlapped neighborhoods, concurrently achieving both their local and common goals, adapting their level of self-interest during the formation of the collaborative teams, and finally, making decentralized decisions. The proposed model has the following characteristics, considered as research contributions, needed to satisfy the mentioned requirements:

- *Operation in Multi-Overlapped Neighborhood*: each agent is capable of acting in multi-overlapped neighborhoods, and the communications between agents in each neighborhood are constrained to their explicit or implicit relations.
- *Self-evaluation*: In open MAS, a self-interested agent has partial access to domain information and shared constrained resources with other agents acting in its neighborhood. Such agent adapts its

self-interest level and cooperates with other agents in its neighborhoods to fulfill common and local goals. The agent applies a decentralized decision-making process, without the need of a central supervisor agent, without commitments or negotiation while collaborating.

- *New Social Reasoning Model*: in open MAS, each agent often leaves and joins the system, and thus it needs a technique to collect information about other agents acting in its neighborhood to perceive its goal relations. This issue is essential for constructing collaborative teams of agents that use the same resources to fulfill a set of goals concurrently.

2. Literature Review

This section reviews and summarizes present existing researches that have discussed cooperation community construction. The study described in this section introduces multi-agent open systems in section 2.1. section 2.2 presents the agents' collaboration. The collaboration objective is discussed in section 2.3. section 2.4 surveys the current research studies by focusing on the formation of a collaborative team. Finally, a conclusion discusses current literature and summarizes this study's requirements related to the proposition of the CTCM model, where a set of agents is trying to achieve a collection of different goals in an open system.

2.1. Multi-Agent in Open System

The Multi-Agents in open systems are composed of independent agents that require resources available in the system and collaborate to reach a set of goals. The following are the proprieties of multi-agent in an open system [14]:

- The Agent joins or leaves the system regularly and randomly.
- The Agent has multiple goals and strategies, which can probably be incompatible.
- The Agent is usually self-interested.
- The acquisition of complete information about the agents' existence and resources' availability in the system is always expensive and sometimes unfeasible because the agents do not desire to reveal their complete information.
- The existence of a global scheduler to manage all system agents is as well unfeasible.

2.2. Agent Collaboration

Agent's collaboration dependents extremely on interaction concepts: cooperation, coordination, and negotiation. Cooperation allows a set of agents to operate jointly to achieve a shared goal [1, 23]. Coordination enables a set of agents to work together by adapting their behaviors and tasks in a common context [4], and finally, negotiation allows a set of agents to attain a mutually accepted agreement [10]. As reported by the literary studies, multi-agent collaboration is a process that requires coordination between a set of agents working together in order to address a subject or to support a common goal [3, 20, 25].

On the other hand, particularly in the subject's addressing the case, collaboration is a cooperated and synchronized process resulting from an ongoing effort to develop and preserve a common perception of the subject [18]. Moreover, collaboration is a cooperative process where a set of agents interact together to reach a common purpose [1, 9]. Therefore, we define collaboration as an activity that requires a team of participants to operate jointly towards a particular goal.

2.3. Collaboration Objective Identification

This section reviews the existing approaches in the literature related to collaboration objectives. Collaboration between agents is required in this situation: Agent must achieve a target, but it cannot reach the target, or in case the agent chooses a collaborative solution [16]. The literature addresses several important objectives related to the collaboration process, such as controlled resources sharing, common goal accomplishment, local profit improvement, local goal accomplishment, and complex action fulfillment. These objectives were addressed by applying distinct methods for team construction. Controlled resource sharing is the main point in dynamic domains. A set of agents coordinate their behavior and collaborate in order to improve the resource usage. This objective was indirectly addressed in methods of coalition formation based on utilities because it is modeled as a common advantage. Also, other multi-agents decentralized methods are studying the allocation of shared resources in dynamic domains [7, 26]. Local profit improvement and local goal accomplishment were addressed in the coalition organization context in which a set of agents enter coalitions and organize their behavior to improve their local payoff [2, 15]. This objective was discussed in the coalition formation context based on utilities. On the other hand, the common goal accomplishment was formalized as a coalition approach [6] and as a teamwork approach [17, 21] based on the agents' type. Finally, to fulfill a complex action, an agent in charge (to which the action is allocated) identifies a set of potential partners called collaborators that can help partially in accomplishing the complex action. These collaborators build a collaborative team. Divide and Conquer is a well-known and preferred technique [5]. [11, 12, 30], where the agent identifying a collaboration need decomposes the complex action into sub-actions, which are more small and simple,

and assigns them to specialized agents. The specialized agent becomes accountable for the assigned sub-action and can iterative apply the divide and conquer technique to decompose the sub-action by making it simpler and smaller and search for additional agents to collaborate [5, 11, 12, 30]. This objective was mainly discussed within the framework of team coordination.

2.4. Collaborative Team Construction

In MAS, an agent is usually modeled as an independent entity. The agent operates autonomously (i.e., without direct human involvement or supervision [26]. Nevertheless, agent autonomy does not mean full independence, and it is not necessary self-sufficiency. Accordingly, the agent has partial information about the domain and limited abilities, and it needs to interact and cooperate with other agents for distinct collaboration objectives [8]. Thus, it is mandatory for autonomous agents to establish teams and coordinate together in order to achieve their goals effectively. Collaborative team construction is an activity where a group of agents create a team and collaborate together for a particular objective. Teams are created only once during the design phase based on a static framework of agents, or during the execution time if a collaboration need arises. In the latter case, the agents build dynamic teams cooperatively. The teams change depending on the environmental modifications. A complete description of the existing contributions in collaborative team construction in a dynamic environment and a comprehensive comparison of current collaborative team construction approaches are presented in [13].

- The existing approaches for collaborative team construction conceive the system as:
- Unique neighborhood: that increases the communication amount.
- Multi-disjointed neighborhoods: that restricts agent to a few alternatives presented in its neighborhoods.
- Multi-overlapped neighborhoods with transitive dependencies: it has an expensive interaction cost (because of the huge number of relationships between all agents).

Accordingly, Neighborhoods' definition aim is the reduction of the complexity of interactions. Thus, modeling a system by allowing agents to act in multi-overlapped neighborhoods is necessary. However, the model should enable agents to determine all possible solutions in all neighborhoods with an acceptable interaction cost.

- Furthermore, existing approaches construct collaborative teams to reach:
- Agent's local goal (that increases local payoff) by modeling agent's resource relations,
- Common goal (by fulfilling a complex task) by modeling agents' action relations.

Moreover, the existing approaches do not model agents' goal relations and consider the agent type as cooperative or self-interested. In fact, in real-world systems with constrained resources, the self-interested agent has a set of goals that can be local or common goals to fulfill concurrently. Thus, the agent needs to perceive its goal relations in order to construct productive collaborative teams. Also, the agent needs a method for adapting its self-interest and cooperative levels, relying on its goal relations, to fulfill its goals concurrently while sharing resources with constraints.

Finally, in current approaches, the process of decision-making is supported by:

- A supervisor agent that assists the team construction process, communication between agents, and making a final decision. But, the existence of a central supervisor is unfeasible based on the properties of an open system.
- A decentralized approach that simplifies hypothesis, for example, the fulfillment of a unique goal and making commitments when constructing a collaborative team.

Nevertheless, the agent needs a decentralized method that enables it to fulfill several goals and facilitates the system leaving and joining in unpredictably while avoiding commitments.

The next section presents our suggested approach to Collaborative Team Construction.

3. Collaborative Agents Team Model

This section proposes a model that allows the construction of a collaborative team of agents having different goals. These agents work in more than one area and share their information with other members of the team. Information sharing, in addition to the social reasoning model, are used by agents to create their goal dependency models, which enable the determination of dependency relationships (for example, conflict, complementary, or collaborative). Based on their dependency models, each agent adjusts its individual-interest level and collaboration in order to achieve a set of goals concurrently while performing in an open system having limited resources. For instance, in case the shared resource is overburdened, agents act in a neighborly manner by deciding to not use the resource, if it is possible, or to cascade the resource claim to other neighborhood, to meet a collaborative goal in a better way by reducing the overload of the shared resource).

The proposed CTCM is a decentralized model where agents are autonomous. Agents communicate together by sharing information in order to collaborate and organize their behavior. In the CTCM, agents are a member of the set of teams, and they are assisted in finding possible options in other neighborhoods by modifying the resource claim to other neighborhoods

if the shared goal can be achieved and the resource in the original neighborhood may be overloaded. Figure 1 illustrates the proposed architecture of the CTCM.

The proposed CTCM architecture consists of six processes. The rectangle represents the process, and the arrow represents the succession of processes. The CTCM begins with Team Update process, and the arrow shows the link between this process and the remaining processes. Moreover, the clouds on this diagram show the models that can be formed at each stage. The cloud shows the model instability as a result of the frequent membership and departure of the agents.

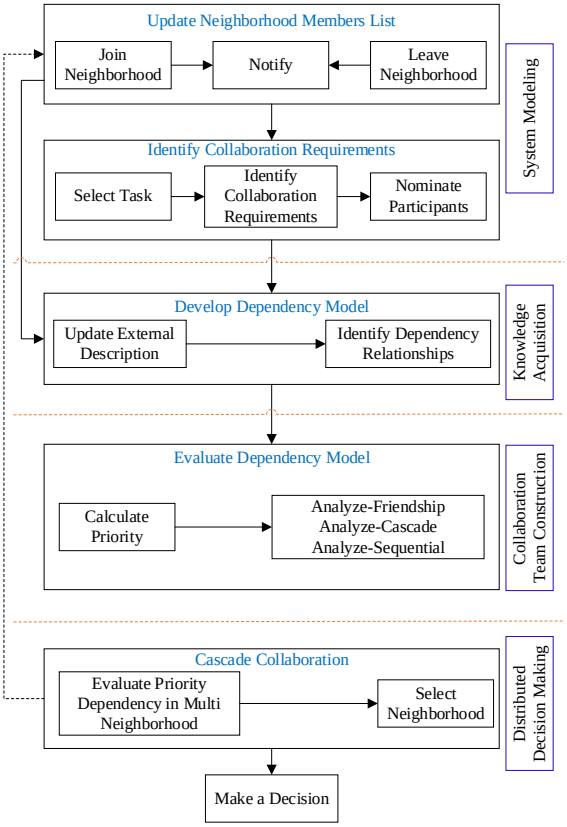

Figure 1. Collaborative team construction model architecture.

3.1. Update Neighborhood Members List

As previously mentioned, the agent joins and leaves the neighborhood unpredictably and regularly in open systems. The neighborhood members are informed about all modifications taken place in the neighborhood. Each neighborhood uses the subscriber-publisher design pattern for capturing the exit and the join cases. The subscriber-publisher is known as an effective and extensible pattern in decentralized systems without restriction on the flexibility of the agent leave/join [30]. In our proposed framework, the neighborhood (represents the shared resource) operates as a publisher and the agent subscribes to the resource it wants to make use of and joins the neighborhood related to the resource. All subscribed agents to a specific neighborhood are members of the neighborhood associated with the resource. When the agent leaves the neighborhood, it has to unsubscribe. The publisher registers all modifications in the neighborhood and informs all its members. Formally, let's Res expresses the resources set in a MAS. Res_i defines a unique resource neighborhood ($Area^t_{Res_i}$) and it has two varying attributes: the capacity, noted $Cap^t_{Res_i}$, and demands, noted $Dem^t_{Res_i}$, at timeslot t.

$$\begin{gathered} Res \stackrel{\text{def}}{=} \cup_{i=1}^{n} \{Res_i\} \\ Res_i \stackrel{\text{def}}{=} \{Cap^t_{Res_i}, Dem^t_{Res_i}, Area^t_{Res_i}\} \\ Area^t_{Res_i} \stackrel{\text{def}}{=} \{Agt_{Res_i}, Agt^t\} \end{gathered} \quad (1)$$

$Area^t_{Res_i}$ contains the set of agents, Agt_{Res_i}, subscribed for using Res_i. $Agt_{Res_i} \subset Agts$ where $Agts$ is the set of all agents in the MAS. $Agt^t_\delta \subset Area^t_{Res_i}$ represents the subset of agents that take actions requiring the resourceRes_i at timeslot t.

$$Agt^t_\delta \subset Agt_{Res_i} \subset Agts \quad (2)$$

Figure 2 illustrates the algorithm of Update Neighborhood Members List. When the agent subscribes to use a resource, it gets access to the neighborhood's information (Res_i). The neighborhood should update this information. The Agent accesses this information at any time throughout its operation in the neighborhood by claiming it from the neighborhood, to prevent possible miscommunication between the publisher and the subscriber. The miscommunication occurs in case the publisher is overburdened and cannot inform all acting agents.

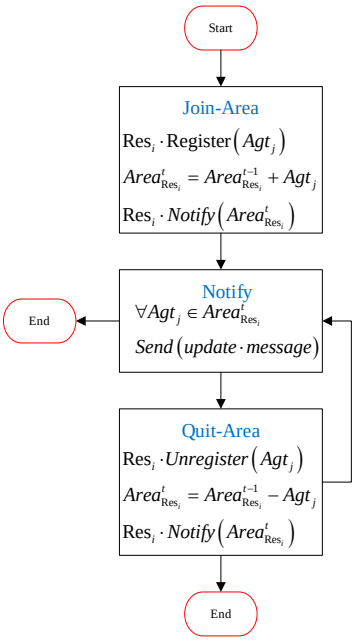

Figure 2. Update neighborhood members list algorithm.

Figure 3 shows an overview of the information concerning the resource that can be accessed by the agent through the neighborhood.

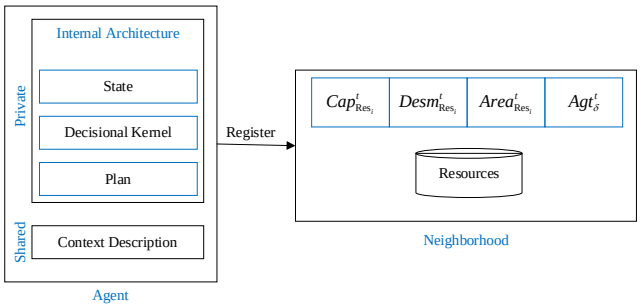

Figure 3. Agent's external description and neighborhood's properties.

3.2. Identify Collaboration Requirements

Since the neighborhood resource is constrained, it becomes overburden at any timeslots. Thus, the agents in the neighborhood have to coordinate to prevent the occurrence of the overload. Because of the absence of a central supervisor or coordinator, the agents have to determine the collaboration requirements. Each agent needs to know the actual $Cap^t_{Res_i}$ and $Dem^t_{Res_i}$ whenever it takes an action. Figure 4 illustrates the algorithm for identifying the collaboration requirement. As illustrated in Figure 2, this process contains three steps: first, the agent selects its task, then increases $Dem^t_{Res_i}$ (the resource's demand) if its task needs to use the resource, it verifies if the resource demand satisfies the available resource capacity $Cap^t_{Res_i}$, and finally, it proposes a set of agents for possible coordination if the $Dem^t_{Res_i} \geq Cap^t_{Res_i}$. In the Nomination of the participants, the agent updates Agt^t_δ by inserting itself and all other agents not added before and their tasks use the shared resource $(\delta^t_{Agt_j})$ at timeslot t.

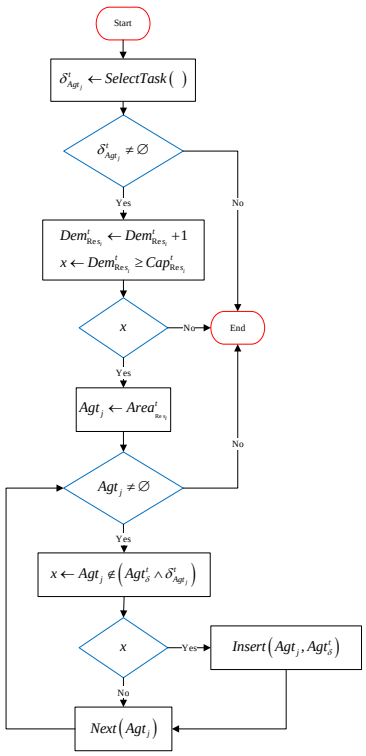

Figure 4. Identify collaboration requirements Algorithm.

3.3. Develop Dependency Model

The planning and simulation techniques are useless for perceiving the agent's behavior and its dependencies because of the agent's unpredictable behavior, and the open system's properties. Therefore, the agent needs a technique to obtaining knowledge about its neighborhood (such as the members of its neighborhood and the actual capacity of the resource) rapidly and in a decentralized manner. On the other hand, the social reasoning techniques proposed and established by Gonfigured and Sivkumar *et al.* [8]. Shah *et al.* [23] have demonstrated encouraging findings in open systems [19, 22, 28], since they allow direct communication between agents, at any time, to exchange information about their tasks, plans, and goals. We choose the social reasoning technique particularly because it allows the information sharing between agents in order to identify their dependency relationships in a distributed way. Regardless of the external description developed by Sichman's model [25] enables the information sharing between agents, it also forces agents to disclose a bunch of their information that is undesired in open systems. Furthermore, Sichman et al.'s social reasoning model is also constraining, since it examines only the agents' task dependency that is beneficial only when the agents fulfill an intricate task. In our model, an updated version of the external description sharing limited information is proposed. The agent builds its Neighboring Dependency Model (DM) by using this updated external description and a social reasoning technique that allows the agents to perceive their goal dependencies during constrained resource sharing.

After the nomination of the participants, the nominated agent for collaboration with other agents acting in the same neighborhood, should determine its neighbors and obtain additional information to comprehend its dependency relationships. Each agent shares part of its information with its neighbors by means of its external description, and stores the information it has obtained from its neighbors. The external description enables the agent to examine its dependency relationships and form its DM. Section 3.3.1 introduces the external description and section 3.3.2 describes the social reasoning technique.

3.2.1. External Description

The agent external description is a data structure accessible by all agents in a neighborhood. This external description contains the shared and acquired information. It produces an abstraction level for diverse agents and it is stored separately from the agent internal architecture (see Figure 3). Explicitly, the external description, adopted from [8], is expressed as:

$$Ext_{Agt_i} \stackrel{\text{def}}{=} \bigcup_{j=1}^{n} Ext_{Agt_i}(Agt_j) \quad (3)$$

Where Ext_{Agt_i} defines the external description of Agt_i (our model uses only this definition of external dependency, the rest of proposed DM is new) and the $Ext_{Agt_i}(Agt_j)$ expresses the entry used to store Agt_j's information. Therefore, the agent discloses limited information and it is not necessary to share its plan, or decision-making process. $Ext_{Agt_i}(Agt_j)$ is defined as:

$$Ext_{Agt_i}(Agt_j) \stackrel{\text{def}}{=} \{G_{Agt_j}, \delta^t_{Agt_j}, S_{Agt_j}, N_{Agt_j}, AP_{Agt_j}, TP_{Agt_j}\} \quad (4)$$

- Goals G_{Agt_j} defines the set of goals that Agt_j desires to fulfill. Each Agent can realize different goals concurrently.

$$G_{Agt_j} = \{G_1, G_2, .., G_n\} \quad (5)$$

- Task $\delta^t_{Agt_j}$ is the task Agt_j performs at timeslot t, representing the following timeslot. In particular, CTCM is focused only on the tasks involving the usage of the shared resource. The Agent's tasks are defined using $\Delta_{Agt_j}(G_k)$, including all tasks that Agt_j has to perform to fulfill the goal G_k.

$$\Delta_{Agt_j}(G_k) = \{T\Delta_{Agt_j}(G_k), \delta^t_{Agt_j}, P\Delta_{Agt_j}(G_k)\} \quad (6)$$

$T\Delta_{Agt_j}(G_k)$ defines the set of already performed tasks for goal G_k. $P\Delta_{Agt_j}(G_k)$ defines the set of pending tasks required to perform later, and $\delta^t_{Agt_j}$ defines the task to be performed in the next timeslot t. In our model, the tasks are represented as a Boolean. The false value indicates that the task does not need the use of a shared resource and true value otherwise.

- Strategies S_{Agtj} defines a set of strategies Agt_j establishes to fulfill its goals. Agent's strategies determine its self-interest and collaboration level for each goal. The collaboration level is determined by two variables, $Min_{G_k}^{Agt_j}$ and $Max_{G_k}^{Agt_j}$.

$$S_{Agt_j} \stackrel{\text{def}}{=} \cup_{k=0}^{n} S_{Agt_j}(G_k) \quad (7)$$

$$S_{Agt_j}(G_k) \stackrel{\text{def}}{=} \{ Min_{G_k}^{Agt_j}, Max_{G_k}^{Agt_j}\} \quad (8)$$

$Min_{G_k}^{Agt_j}$ defines a criterion set by Agt_j to specify the minimum number of tasks, T_{g_k} , that must be performed for G_k before timeslot t'. $Min_{G_k}^{Agt_j}$ depends on the Agt_j internal state and preferences and it defines the timeslot from which Agt_j will be able to coordinate with other agents in the same neighborhood when performing task to fulfil G_k Thus, if Agt_j achieves T_{gk} number before/at timeslot t', it is highly possible to be cooperative. Particularly, a smart phone with sufficient battery charge shows a cooperative behavior compared to when it gets insufficient battery charge. Moreover, setting $Min_{G_k}^{Agt_j}$ allows to agent to find out its dependency relationships (described in section 3.2).

$$Min_{G_k}^{Agt_j} = \{T_{g_k}, t'_k\} \quad (9)$$

$Max_{G_k}^{Agt_j}$ specifies the maximum number of tasks, requiring access to a shared resource Res_i, the agent desires to perform during its access to Res_j before timeslot t''(representing the last timeslot that Agt_j is able to use Res_i for G_k).

$$Max_{G_k}^{Agt_j} \stackrel{\text{def}}{=} \{(\Lambda_{G_k}, t''_k) : \Lambda_{G_k} \leq \left|\Delta_{Agt_j}(G_k)\right|\} \quad (10)$$

To illustrate the concept of S_{Agt_j} and its $Min_{G_k}^{Agt_j}$ and $Max_{G_k}^{Agt_j}$, suppose that a smartphone needs electrical power to charge its battery. $Min_{G_k}^{Agt_j}$ value is set to the minimum power amount necessary to operate normally and $Max_{G_k}^{Agt_j}$value is set to the maximum power amount needed to completely charge the battery. These two values allow the smartphone to better perceive its requirements and to comport more rationally.

- Neighborhood N_{Agtj} defines the number of neighborhoods of which Agt_j is a member.
- *Priority*: Each agent computes two priorities AP_{Agtj} (for resource access). And TP_{Agtj} (transferred priority to Agt_j from other agents when the collaboration). These priorities are deeply described in section 4.

3.3.2. Identify Dependency Relationships

In order to identify a candidate team for possible collaboration, each agent has to understand the resource dependency nature. Also, the resource dependency nature helps the agents to discover their own dependency relationships which play an important role in the collaborative team construction process.

In CTCM, the autonomous agent shares a resource in their neighborhood. All neighbors require the shared resource usage to fulfil their goals but none of them cannot monitor the resource. Thus, the neighbors are resource-dependent.

To identify the type of dependencies the agent has with its neighbors, each agent compares its local information to the information of the neighbors' external description. The existing approaches model the behavior of an agent statically as cooperative or competitive for resource sharing and ignore their goal dependencies [28]. Nevertheless, additional kinds of dependencies exist between agents' goals, moreover, these dependencies are not static and can change depending on the shared resource availability and the progress of their goal fulfillment. By way of illustration, the passengers in public transportation comport always competitively, but in an emergency circumstance, they behave cooperatively. Consequently, each agent must understand its goal dependencies during its operation in the MAS. Figure

5 shows different types of dependency relationships and the criteria used to select the adequate type of dependencies.

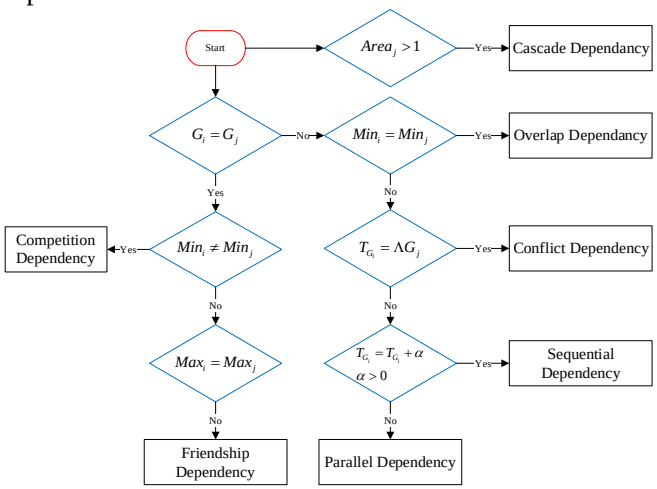

Figure 5. Identify dependency relationships algorithm.

There are seven different types of possible dependencies between agents' goals contrary to the existing resource allocation techniques in MAS that define only two (competitive and cooperative behaviors). The additional proposed goal dependency relationships describe the agents' behavior during constraint resource sharing. Sometimes, the agent has distinct goals that impact its behavior (for instance, self-interested, collaborative, or adjusting its self-interest level).

- *Cascade Dependency*: the agent, which is a member of multiple neighborhoods, cascades the request of constrained resources to another neighborhood having the identical resource.
- *Overlap Dependency*: this type of dependency exist between the agents that are members of the same neighborhood and perform tasks with the same Min for different goals. Possessing the same Min means that these agents need to use an equal amount of the constrained resource before timeslot t'.
- *Conflict Dependency*: this type of dependency exists between the agents that need the same constrained resource but to achieve different goals, however, their strategies forbid cooperation and coordination.
- *Sequential Dependency*: there is a dependence between the agents' goals. This dependence necessitates a sequence in goal fulfillment.
- *Parallel Dependency*: two agents perform the same tasks to fulfill distinct goals and possess distinct strategies without sequential or conflict relationships. Thus, these agents can, but not must, use the resource simultaneously.
- *Competition Dependency*: two agents possess the same goals and perform the same tasks but possess different strategies. Thus, each agent wants to make full use of the constrained resource in the minimum time amount.
- *Friendship Dependency*: the agents possess identical goals, perform identical tasks, possess identical strategies, and coordinate together to fulfill each other's goals.

3.4. Evaluate Dependency Model

Nominated agents from the 'Identify Collaboration Requirements' phase (see Figure 4), reassess their selected tasks before the collaboration starts. In this phase, each agent tries to reduce its demand. This process includes two sub-process as shown in Figure 1, which are: Calculate Priority and Evaluate External Description.

3.4.1. Calculate Priority

Here the agent calculates its priority to use the constrained resource, in comparison to the priorities of the other agents. In CTCM, the agent has two different values for the priority values: AP AP_{Agti} (Access-Priority) and TP_{Agti} (Transferred Priority).

Access-Priority: AP_{Agti} is calculated by summing the priorities of all Agt_i's goals depending on the task $\delta_{Agt_i}^t$ performed in timeslot t. A unique priority is computed depending on the number of pending tasks $|P\Delta_{Agt_i}(G_k)|$ and the time period spent by the agent in the neighborhood, $Duration_{Agti}$ A lower priority indicates that the agent possesses less pending tasks to perform in the remaining time. In case the agents cooperate and consider their dependencies, the agent with low priority gets a slight chance to use the constrained resource in comparison to others having higher priorities.

$$AP_{Agt_i} = \sum_{k=1}^{n} \frac{|P\Delta_{Agt_i}(G_k)|}{Duration_{Agt_i}} \quad (11)$$

Transferred Priority: Agt_j's Transferred-Priority, TP_{Agtj} records the Access-Priorities of other agents that are transferred to Agt_i. An agent transfers its priority in two cases:

1. It has Sequential Dependency, thus the agent failure to fulfill its goals implies all dependents agents will also fail to fulfil their goals.
2. During the collaboration, the agent with the highest priority receives the transferred priorities from other agents in the team to be able to use the shared resource. At this step, $TP_{Agtj=}$ 0 (see Figure 6).

The Transferred Priority belongs to the interval [0, Max_{TP}] where Max_{TP} depends on the application and defines the maximum value of Transferred-Priority. Max_{TP} contributes in distributing the transferred priorities among agents possessing high priorities and avoiding the aggregation of transferred priorities into a unique agent.

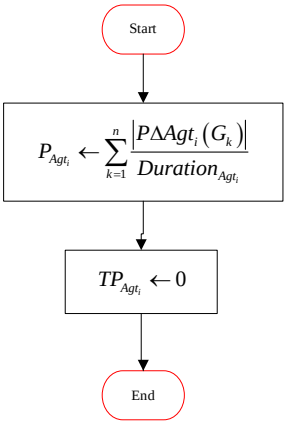

Figure 6. Calculate priority algorithm.

3.4.2. Evaluate External Description

After the priorities calculation, each selected agent Agt_i identifies its Ext_{Agt_i} for *Friendship*, *Cascade*, and *Sequential Dependencies* to discover a potential demands reduction before initiating the collaboration (see Figure 7).

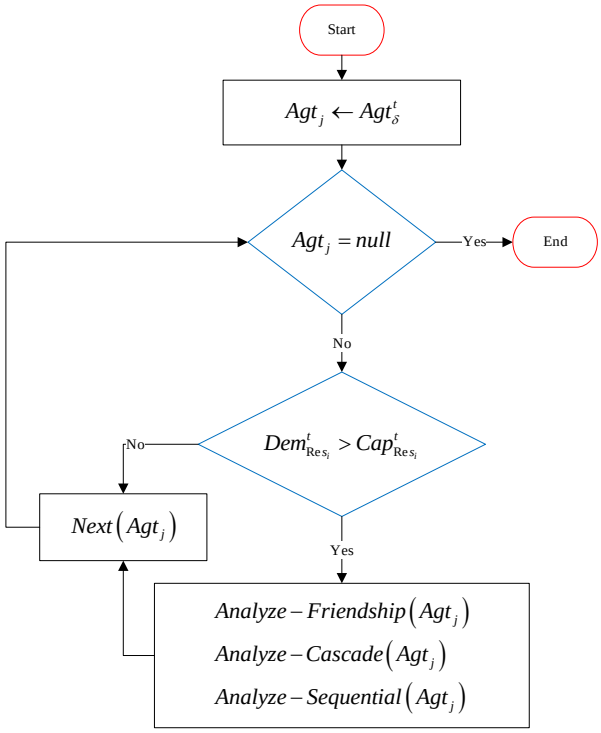

Figure 7. Evaluate external description algorithm.

As previously explained, in presence of the *Friendship* dependency between two agents Agt_i and Agt_j, Agt_i transfers its priority to Agt_j in three cases (Figure 8):

- No agent in Agt_δ^t having *Friendship* dependency with Agt_i
- Agt_i priority is less than Agt_j priority
- $TP_{Agt_j} < MaxTP$.

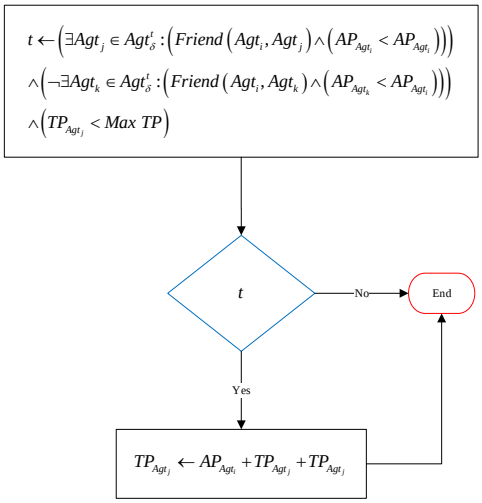

Figure 8. Analyze-*friendship* algorithm.

In presence of the *Cascade* dependency between two agents Agt_j and Agt_j, Agt_j demands Agt_j to cascade the resource request to another neighborhood. Agt_j assesses its neighborhoods, and accordingly it chooses whether or not to modify its task in the actual neighborhood and perform the task in the new neighborhood (Figure 9).

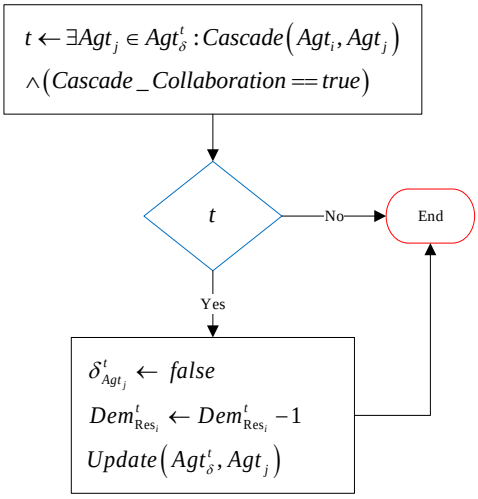

Figure 9. Analyze-*cascade* algorithm.

In presence of the *Sequential* dependency between two agents Agt_j and Agt_j, Agt_j wants Agt_j to achieve its goal faster. Agt_i transfers its AP_{Agt_i} and TP_{Agt_i} to Agt_j to enhance Agt_j's chance to use the shared resource. Then, Agt_i modifies its task and updates Dem^t_{Resj} In case the agent has more than one *Sequential* dependencies with a set of agents, it selects one among them randomly. If there is a *Sequential* dependencies chain between agents, this algorithm is executed recursively (see Figure 10). In case the agent chooses to rest in the actual neighborhood, it calls Algorithm 5 and starts cooperation in the new neighborhood. In a successful collaboration, the agent modifies its task in the original neighborhood and decreases Dem^t_{Resj}.

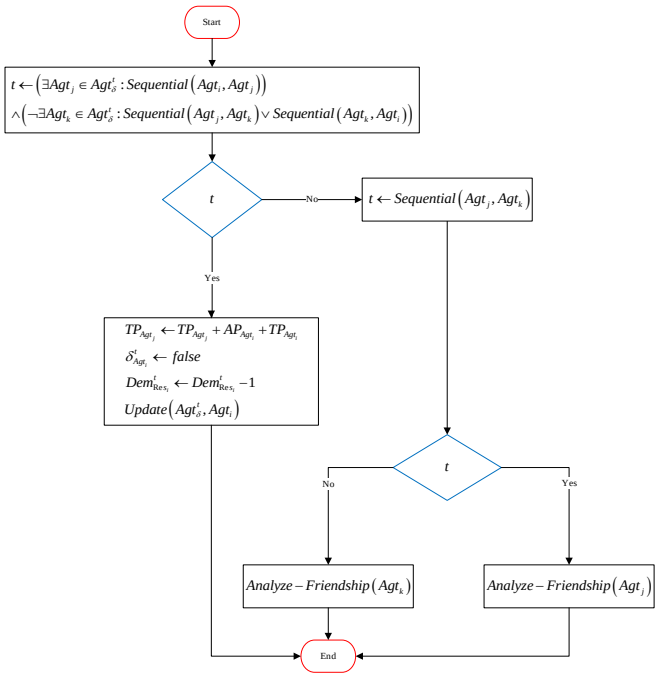

Figure 10. Analyze-*sequential* algorithm.

The remaining dependencies type such as *Overlap, Conflict, Parallel, and Competition* Dependencies are necessary for the course of the decision-making process.

3.5. Cascade Collaboration

This stage represents the 5th step in CTCM where each agent belonging to several neighborhoods evaluates its priorities and dependencies and decides to change its tasks and cascades the resource claim to a different neighborhood (Figure 11).

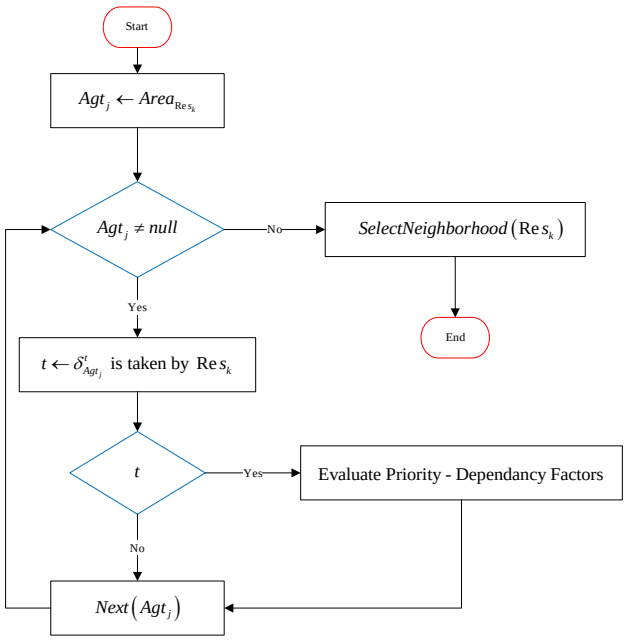

Figure 11. Cascade collaboration algorithm.

As previously stated, an agent belongs to multiple neighborhoods resulting in overlapped neighborhoods. Depending on the selected task, the agent chooses the neighborhood. The agents acting in each neighborhood of multiple neighborhoods context are classified into four categories (Table 1). These categories are useful concepts if an agent wants to assess its different neighborhoods and compare them to its actual neighborhood.

Table 1. Multiple neighborhood categories and properties.

Category	Category 1	Category 2	Category 3	Category 4
Properties	$\delta_{Agt_i}^t$=1 & N_{Agt_i}= 1	$\delta_{Agt_i}^t$=0 & N_{Agt_i}= 1	$\delta_{Agt_i}^t$=1 & N_{Agt_i} >1	$\delta_{Agt_i}^t$= 0 & N_{Agt_i} >1

- *Category* 1 defines the set of agents performing tasks in the following timeslot *t* and belong to only one neighborhood.
- *Category* 2 defines the set of agents which do not perform any task in the following timeslot *t* and belong to only one neighborhood.
- *Category* 3 defines the set of agents performing tasks in the following timeslot *t* and belong to multiple neighborhoods.
- *Category* 4 defines the set of agents which do not perform any task in the following timeslot *t* and belong to multiple neighborhoods. These agents can or not perform tasks in other neighborhoods. In case an agent is performing task, it can modify the task and decides to share the resource in this neighborhood and forwards the collaboration request.

Whenever the agent needs to choose to cascade the collaboration request, it examines four factors in each neighborhood of which it belongs:

- The Cap^t_{Resj} and Dem^t_{Resj} for the next timeslot *t*.
- The agents' population in each category.
- Its Access-Priority.
- Its Dependency Relationships (in particular, *Friendship and Competition* Dependencies) .

The agent estimates (based on these factors) its chance to use the shared resource in every neighborhood. Depending on these factors values, there are three possible cases for an agent once it decides to cascade the resource request:

- *Case* 1: The agent has access to the shared resource in its current neighborhood, in spite of all its neighbors are operating in other neighborhoods and decide to act in this neighborhood.
- *Case* 2: The agent has access to the shared resource if the actual constraints of the neighborhood are stable.
- *Case* 3: The agent cannot access the shared resource because of the actual constraints of the neighborhood. In this case, comparable to case 2, the agent with higher priority or dependency relationships changes to another neighborhood.

3.6. Make Decentralized Decision

The collaboration decision is mainly based on two factors: the dependency relationships and priorities. Decision-making is an iterative process. In each iteration, the nominated agent with the smallest priority executes the decision-making algorithm and completes its task. The algorithm continues until $Dem^t_{Res_i} \leq Cap^t_{Res_i}$ or all agents in Agt^t_δ executed the process and completed their tasks. In this process, the agents possessing the smallest priorities have to modify their tasks to reduce the demand. Nevertheless, these agents should verify their dependency relationships before making any decision (see Figure 12).

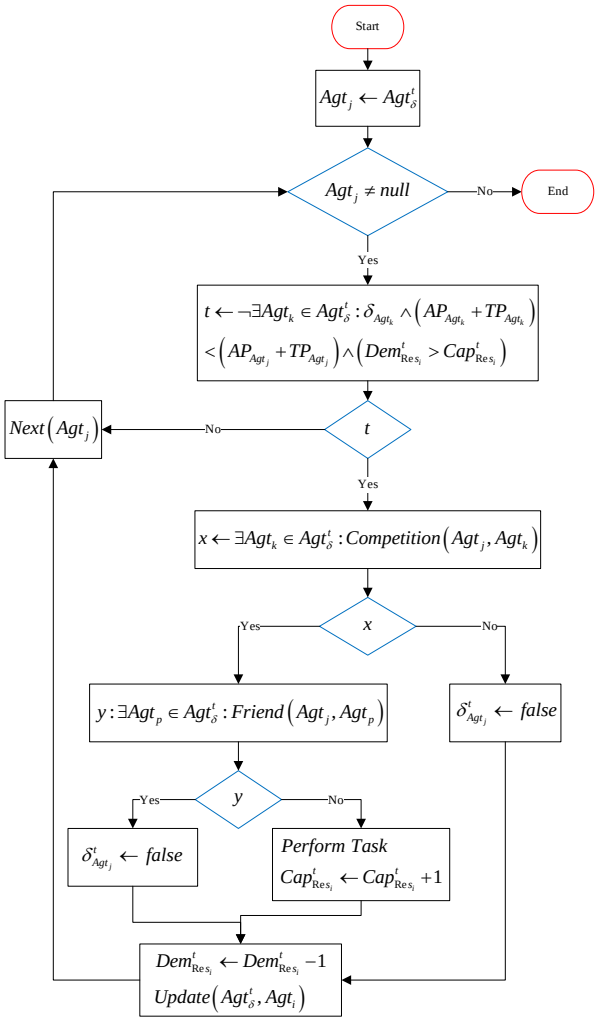

Figure 12. Decentralized collaboration decision algorithm.

4. Conclusions

This paper defined the characteristics and requirements of CTCM that allow a set of agents to create cooperative teams in an open system by sharing constrained resources to perform a set of tasks concurrently in order to achieve several goals. The proposed model CTCM is decentralized and allows a set of agents to act in different overlapping neighborhoods, to create collaborative teams by applying a social reasoning technique, and to cooperate in a decentralized way. CTCM represents an open-system as a multi-neighborhood system where agents act in several neighborhoods. This feature allows agents to exploit the interaction with all their neighbors, without restraining agents from having any particular dependency relations. Moreover, CTCM defines a new technique for obtaining information, and a modified version of Sichman *et al*. [24] social reasoning model that allows agents to proceed with their decision factors from selecting exclusively between a common goal or local goal. Instead of that CTCM considers both goals (common and local) while accessing constrained resources. In addition, CTCM presents a novel collaborative Team construction method that enables each agent to adjust its self-interest level and collaboration, based on its state and dependency relationships while trying to achieve several goals concurrently. CTCM presents a decentralized decision-making process that relaxes the agent from making engagements and staying in the neighborhood throughout the coordination process.

This paper presented the theoretical design of the CTCM. As future work, we have to develop a porotype to implement the different main components of CTCM by describing the major modules at Neighborhood and Agent layers. The prototype will be used for CTCM evaluation at a different levels of neighborhoods' density and agents' mobility.

References

[1] Bansal G., Nushi B., Kamar E., Lasecki W., Weld D., and Horvitz E., "Beyond Accuracy: The Role of Mental Models in Human-AI Team Performance," *in Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, Skamania Lodge, pp. 2-11, 2019.

[2] Bistaffa F., Farinelli A., and Ramchurn S., "Sharing Rides with Friends: A Coalition Formation Algorithm for Ridesharing," *in Proceedings of the AAAI Conference on Artificial Intelligence*, Phoenix, pp. 4746-4752, 2015.

[3] Burton M., Brna P., and Pilkington R., "Clarissa: A Laboratory for The Modelling of Collaboration," *International Journal of Artificial Intelligence in Education*, vol. 11, no. 2, pp. 79-105, 2000.

[4] Chen G., Yang Z., He H., and Goh K., "Coordinating Multiple Agents Via Reinforcement Learning," *Autonomous Agents and Multi-Agent Systems*, vol. 10, no. 3, pp. 273-328, 2005.

[5] Dunin-Kęplicz B. and Verbrugge R., *Teamwork in Multi-Agent Systems: A Formal Approach*, John Wiley and Sons, 2010.

[6] Eddy Y., Gooi H., and Chen S., "Multi-Agent

System for Distributed Management of Microgrids," *IEEE Transactions on Power Systems*, vol. 30, no. 1, pp. 24-34, 2014.

[7] Faruk M. and Sivakumar D., "Towards Self Configured Multi-Agent Resource Allocation Framework for Cloud Computing Environments," *International Journal of Engineering Technology*, vol. 6, no. 2, pp. 920-928, 2014.

[8] Golpayegani F., Dusparic I., and Clarke S., "Using Social Dependence to Enable Neighbourly Behaviour in Open Multi-Agent Systems," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 3, pp. 1-31, 2019.

[9] Guarnieri P., "Interactive Intentionality and Norm Formation," *Journal of Institutional Economics*, vol. 15, no. 4, pp. 579-593, 2019.

[10] Harrison D., "Better Together: Integrating Artificial Intelligence into Team Cognition," School of Advanced Military Studies US Army Command and General Staff College Fort Leavenworth, 2019.

[11] Hausknecht M., Mupparaju P., Subramanian S., Kalyanakrishnan S., and Stone P., "Half Field Offense: An Environment for Multiagent Learning and Ad Hoc Teamwork," *in Proceedings of AAMAS Adaptive Learning Agents Workshop*, Singapore, pp. 2-7, 2016.

[12] Hayano M., Hamada D., and Sugawara T., "Role And Member Selection in Team Formation Using Resource Estimation for Large-Scale Multi-Agent Systems," *Neurocomputing*, vol. 146, pp. 164-172, 2014.

[13] Houhamdi Z. and Athamena B., "Collaborative Team Construction in Open Multi-Agents System," *in Proceedings of 21st International Arab Conference on Information Technology*, 6th of October city, pp. 1-7, 2020.

[14] Huynh T., Jennings N., and Shadbolt N., "An Integrated Trust and Reputation Model for Open Multi-Agent Systems," *Autonomous Agents and Multi-Agent Systems*, vol. 13, no. 2, pp. 119-154, 2006.

[15] Janovsky P. and Deloach S., "Increasing Use of Renewable Energy by Coalition Formation of Renewable Generators and Energy Stores," *in Proceedings of Multi-Agent Systems and Agreement Technologies*, Cham, pp. 140-147, 2017.

[16] Khan S., "Rational Agents: Prioritized Goals, Goal Dynamics, and Agent Programming Languages with Declarative Goals," *in Proceedings of 9th International Conference on Autonomous Agents and Multiagent Systems*, Toronto, pp. 1653-1654, 2010.

[17] Liemhetcharat S. and Veloso M., "Weighted Synergy Graphs for Effective Team Formation with Heterogeneous Ad Hoc Agents," *Artificial Intelligence*, vol. 208, no. 1, pp. 41-65, 2014.

[18] Meter P., List A., Lombardi D., and Kendeou P., *Handbook of Learning from Multiple Representations and Perspectives*, Routledge, 2020.

[19] Petukhova V., Sharifullaeva F., and Klakow D., "Modelling Shared Decision Making in Medical Negotiations: Interactive Training with Cognitive Agents," *in Proceedings of International Conference on Principles and Practice of Multi-Agent Systems*, Cham, pp. 251-270, 2019.

[20] Russell N., Barros A., and Hofstede A., "Towards a Coordinative Theory for Flexible Work Collaboration," *in Proceedings of the 38th International Conference on Information Systems*, Seoul, pp. 1-21, 2017.

[21] Sacheli L., Aglioti S., and Candidi M., "Social Cues to Joint Actions: The Role of Shared Goals," *Frontiers in Psychology*, vol. 6, pp. 1034, 2015.

[22] Santarra T., "Communicating Plans in Ad Hoc Multiagent Teams," University of Californiasanta Cruz, 2019.

[23] Shah S., Ahmad J., and Rehman N., "Design and Implementation of Inter-Operable and Secure Agent Migration Protocol," *The International Arab Journal of Information Technology*, vol. 17, no. 4, pp. 461-470, 2020.

[24] Sichman S., Conte R., Castelfranchi C., and Demazeau Y., "A Social Reasoning Mechanism Based on Dependence Networks," *in Proceedings of 11th European Conference on Artificial Intelligence*, New York, pp. 188-192, 1994.

[25] Sichman S. and Conte R., "Multi-Agent Dependence by Dependence Graphs," *in Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems: part 1*, Bologna, pp. 483-490, 2002.

[26] Sidner C., Bickmore T., Nooraie B., Rich C., Ring L., Shayganfar M., and Vardoulakis L., "Creating New Technologies for Companionable Agents to Support Isolated Older Adults," *ACM Transactions on Interactive Intelligent Systems*, vol. 8, no. 3, pp. 1-27, 2018.

[27] Skobelev P., "Multi-Agent Systems for Real Time Resource Allocation, Scheduling, Optimization and Controlling: Industrial Applications," *in Proceedings of the International Conference on Industrial Applications of Holonic and Multi-Agent Systems*, Berlin, pp. 1-14, 2011.

[28] Wang H. and Chu X., "Distance-Constrained Resource-Sharing Criteria for Device-to-Device Communications Underlaying Cellular Networks," *Electronics letters*, vol. 48, no. 9, pp.

528-530, 2012.
[29] Zhang I., Sharma N., Szekeres A., Krishnamurthy A., and Ports D., "Building Consistent Transactions with Inconsistent Replication," *ACM Transactions on Computer Systems*, vol. 35, no. 4, pp. 1-37, 2018.
[30] Zhang Y., Volz T., Loerger R., and Yen J., "A Decision-Theoretic Approach for Designing Proactive Communication in Multi-Agent Teamwork," *in Proceedings of the ACM Symposium on Applied Computing*, New York, pp. 64-71, 2004.
[31] Zimmerling M., Mottola L., and Santini S., "Synchronous Transmissions in Low-Power Wireless: A Survey of Communication Protocols and Network Services," *arXiv*, vol. 53, no. 6, pp. 1-39, 2020.

Zina Houhamdi received her Ph.D. in Software Engineering in 2004. She is a Professor at the Department of Cybersecurity, College of Engineering, Al Ain University, United Arab Emirates. Her research work has been published in several academic journals and has been presented at scientific conferences. Her main research interest is on Internet of Thing, Artificial Intelligence, particularly in Multi-Agent Systems Modelling, Testing and Applications. She is published several papers in journals and international peer-reviewed conferences.

Belkacem Athamena holds a Ph.D. in System Analysis and Applications. He is an Associate Professor at the Department of Business Administration, College of Business, Al Ain University, United Arab Emirates. His main research interest is in system modeling and analysis, multi-agent, fuzzy logic, software testing, Petri nets, formal methods, data quality, and fault diagnosis. He has published many refereed journal articles, contributed chapters and presented papers at conferences.