Grid Computing manuscript No.
(will be inserted by the editor)

Cloud-Based Multi-Agent Cooperation for IoT Devices Using Workflow-Nets

Yehia Kotb · Ismaeel Al Ridhawi · Moayad Aloqaily · Thar Baker · Yaser Jararweh · Hissam Tawfik

Received: date / Accepted: date

Abstract Most Internet of Things (IoT)-based service requests require excessive computation which exceeds an IoT device's capabilities. Cloud-based solutions were introduced to outsource most of the computation to the data center. The integration of multi-agent IoT systems with cloud computing technology makes it possible to provide faster, more efficient and real-time solutions. Multi-agent cooperation for distributed systems such as fog-based cloud computing has gained popularity in contemporary research areas such as service composition and IoT robotic systems. Enhanced cloud computing performance gains and fog site load distribution are direct achievements of such cooperation. In this article, we propose a workflow-net based framework for agent cooperation to enable collaboration among fog computing devices and form a cooperative IoT service delivery system. A cooperation operator is used to find the topology and structure of the resulting cooperative set of fog computing agents. The operator shifts the problem defined as a set of workflow-nets into algebraic representations to provide a mechanism

Yehia Kotb
Department of Computer Science, University of Western Ontario, London, Ontario, Canada.
E-mail: YKotb@alumni.uwo.ca

Ismaeel Al Ridhawi
University of Ottawa, School of Electrical Engineering and Computer Science, Ottawa, Ontario, Canada.
E-mail: Ismaeel.AlRidhawi@uottawa.ca

Moayad Aloqaily
Gnowit Inc., 7 Bayview Road, Ottawa, ON, Canada, K1Y2C5.
E-mail: Moayad@gnowit.com

Yaser Jararweh
Jordan University of Science and Technology, Irbid, Jordan.
E-mail: YiJararweh@just.edu.jo

Thar Baker
Liverpool John Moores University, United Kingdom.
E-mail: t.baker@ljmu.ac.uk

Hissam Tawfik
Leeds Beckett University, United Kingdom.
E-mail: h.tawfik@leedsbeckett.ac.uk

for solving the optimization problem mathematically. IoT device resource and collaboration capabilities are properties which are considered in the selection process of the cooperating IoT agents from different fog computing sites. Experimental results in the form of simulation and implementation show that the cooperation process increases the number of achieved tasks and is performed in a timely manner.

Keywords Cloud Computing · Fog Computing · Petri-Net · Workflow-Net · Internet of Things · Agent Cooperation.

1 Introduction

IoT devices, both stationary and mobile, provide simple and complex services, especially, in cloud computing and big data applications. For instance, high levels of performance, accuracy, and robustness are achieved using IoT robotic devices in real-time applications such as manufacturing. However, in more advanced IoT-based scenarios such as 'search-and-rescue' [1], complex service composition [2] and exploring new land and water terrains [3], pre-programmed IoT robotic devices may not be able to achieve the requested service due to their hardware and software limitations. The integration of IoT device communication using traditional and advanced networks (i.e. smart grid networks [4]) with parallel distributed systems has made it possible to achieve new complex tasks such as long-distance medical surgeries and other specialized cases.

Although grid networks integrate communication and node resources, due to the limitations of mobile node resources such as computation and storage, IoT systems have moved towards more advanced and emerging technologies such as cloud computing and big data [5]. Cloud computing is a data driven technology that enables storage, computing, analysis, networking, and visualization of very large data at cloud computing data centers. Such technology allowed for the design of multi-agent distributed systems that can achieve complex tasks with high performance levels [6].

Cloud computing has brought great advances, but cloud-based systems have moved away from the traditional centralized approach towards a more sustainable and robust distributed scheme as well as enhancing the grid with cloud computing [7]. Fog computing has revolutionized traditional cloud computing architectures to assure that cloud computing related tasks are performed at different fog computing sites [8, 9]. Offloading cloud computing related tasks to fog computing devices in a distributed fashion not only balances the load among cloud and fog computing sites, but can also support service task achievement with reduced latency and improved service quality [10]. Fog computing is not an alternative to cloud computing, but rather provides support. It was introduced to solve delay issues for time sensitive applications. Fog computing is an intermediate layer between IoT devices and the cloud datacenter. It extends the traditional cloud computing paradigm towards the underlying networks by providing limited resources to clients. By bringing the network and cloud computing resources closer to the network edge, substantial amounts of requests can be processed near the IoT devices instead of sending data all the way to the cloud datacenter.

Parallel distributed systems incorporate advantages over traditional systems in terms of performance, design simplicity and task achievement that is deemed

impossible using a single process [11–14]. Cooperation is defined as the process of merging and managing resources and capabilities such as sensing, knowledge and computation power to reach an objective. The objective is achieved by having cooperating fog computing entities negotiate for IoT device resources and perform task planning and scheduling.

In this article, we address the problem of multi-fog IoT agent cooperation for enhanced task achievement. For instance, if a particular is requested from a set of IoT devices in one area (i.e. fog computing site), such that a fog's IoT devices' capabilities are insufficient or may consume excessive energy and time to perform, then adopting a collaborative approach which relies on other IoT devices in cooperative fog computing sites may achieve the requested service in accordance with the quality and energy restrictions. An extension to Petri-nets [15], known as Workflow-nets [16, 17] is used to establish a framework for cooperating fog computing IoT agents [18] to achieve tasks by merging and managing the available IoT device capabilities from different fog computing sites.

The cooperating partners are selected from different fog computing sites based on the task coverage they maintain in which properties such as device capabilities and whether device agents can collaboratively achieve the required task on time are considered. The presented work fits well within multi-objective multi-device scenarios such as IoT robotic systems that may require usage of IoT robot device capabilities which are not available in their surrounding environment, but rather may be available at different locations. For instance, assume that a set of collaborative IoT drones are available in a certain location and are capable of communicating with fog devices available at a certain fog computing site as depicted in Figure 1. Assuming that the drones are incapable of completing the task due to a resource constraint such as the lack of computing capabilities or the lack of power availability. The fog determines that another IoT device (or set of devices) at other collaborative fog computing sites are able and willing to cooperate with the drones to achieve the requested service. Therefore, the new cooperating agents will negotiate for resources, then perform task planning and scheduling to complete the requested service in a timely and more efficient manner. This is achieved by constructing, merging, and managing workflow-nets. Workflow-nets have been a favorable choice and are adopted in research areas undergoing intense study such as enhanced cloud service composition [19] [20], agent cooperation for task achievement [21], and vehicular service management [22].

This work relies on a cooperation operator introduced in [23,24] that formulates the topology and structure for a set of cooperating fog computing sites and IoT device agents. The cooperative operator turns the composition problem defined as a set of Workflow-nets into algebraic representations to solve and optimize the problem mathematically. The presented work extends the work introduced in [23] to provide a solution for fog and cloud computing systems targeting areas of IoT agent cooperation to produce enhanced composed services (both simple and complex) in a timely manner. We show through mathematical proofs and experimental results that the proposed cooperative solution is applicable to fog-based systems with limited node resources. To clarify the fog-based cooperative solution incorporated in this paper for the reader, a realistic IoT-based cooperation application scenario is integrated, which focuses on multi-robot (i.e. IoT multi-device) task achievement. Hence, the literature review and experimental evaluation sections will focus on IoT-based robotic issues and solutions.

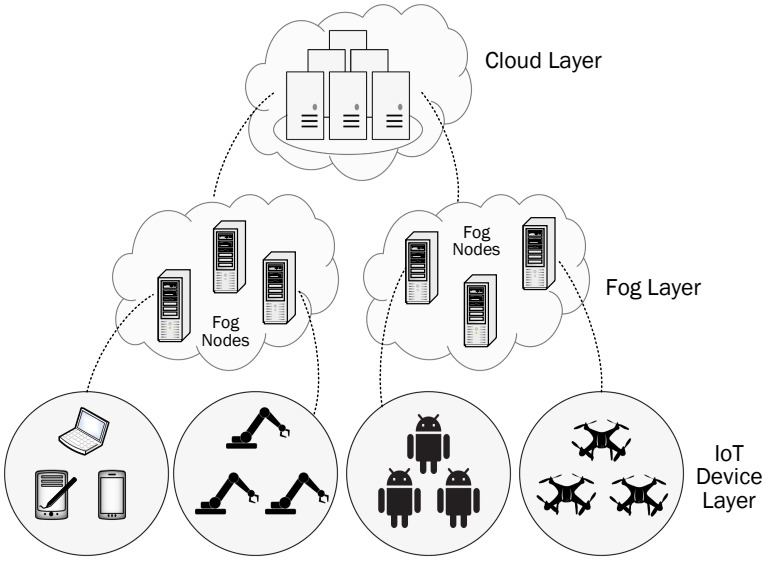

Fig. 1: Merging and managing IoT device capabilities at different fog computing sites to accomplish a task efficiently.

To this end, related work is surveyed in Section 2. Section 3 illustrates the methodology of the proposed framework and the agent cooperation algorithm. Section 4 briefly summarizes the cooperative operator and its properties. Section 5 presents experimental evaluation results in support of the framework through simulation and implementation tests. Section 6 concludes the paper and outlines potential future work.

2 Related Literature

Cooperation is defined as the process of merging and managing resources and capabilities, such as computational power, of different entities to reach an objective. The objective is achieved by having cooperating entities negotiate for resources and perform task planning and scheduling. Cooperation among different entities is widely common in many different engineering disciplines, such as robotics, wireless sensor networks [25, 26], and vehicular communication [27–32]. Advances in both IoT devices and network technologies have made it possible for mobile and stationary devices such as robots to perform collaborative tasks in various environments [33]. Moreover, today's advanced wireless technologies such as 5G have made connectivity to cloud computing services easier, to allow for significantly enhanced IoT system capabilities [34] as well as data management and congestion mitigation in densely crowded environments [35] [36].

Cloud providers collaboration for the purpose of service discovery and management has been discussed in [37]. Solutions such as hybrid cloud and adaptive scheduling for heterogeneous workloads have impacted the performability of work-

flow applications in cloud environments [38] [39]. On the other hand, Fog node collaboration for the purpose of fulfilling requests and services sent from IoT nodes to the fog computing layer and improving fog computing performance is considered in [40] [41]. The proposed solution assumes that if a fog computing node can accept IoT node requests based on its current load, then it will process the requests. Otherwise, it would offload the requests to other fog computing nodes or the cloud datacenter, to achieve what they call "load sharing". Two modes of fog computing interactions are adapted: centralized and distributed. In centralized, within each fog computing domain, a node is selected to act as a central authority to control the interactions among the fog computing nodes in the domain. Fog computing nodes within a single domain report their estimated time to process IoT node requests to the central fog computing node. The central node then announces the estimated processing times to neighboring fog computing nodes. Fog computing nodes compare those timings against pre-calculated thresholds. If a fog computing node is capable of processing this request, then the task is offloaded to the capable fog computing node. Otherwise, the request is forwarded to the cloud datacenter. In the latter mode of fog computing node interaction (i.e. distributed), fog computing nodes can only collaborate with neighboring nodes using a universal protocol. Each fog computing node maintains a reachability table using the estimated task processing times it receives from neighboring fog computing nodes. Similar to the central mode solution, a fog computing node will select the best neighbor that is capable of processing the request with the smallest estimated processing time, plus half of the roundtrip latency.

The authors in [42] introduced a communication framework for Autonomous Underwater Vehicles (AUVs) which expands the computational and data resources that are available to a team of AUVs by including public cloud computing resources. A resource provisioning engine is designed to offload some of the AUVs' tasks to the cloud datacenter. AUV workload is shared between local and cloud computing resources based on communication cost, computation cost, and battery capacity. The aim of the solution is to increase the lifetime of such power-limited IoT devices. Additionally, with the availability of more sophisticated cloud computing resources, high accuracy is also achievable. The focus of their work is on resource-provisioning to allocate tasks between the cloud and local resources based on the objective of minimizing execution time or the price of the execution. The required set of tasks to be performed are represented using a Directed Acyclic Graph (DAG), with tasks denoted by nodes in the graph. Directed edges indicate the information flow from one task to another. Nodes on the same level of the graph are executed in parallel, whereas nodes in consecutive levels are executed sequentially. Simulations using MATLAB were conducted to show the improvement in task accuracy.

Wang et al. [43] proposed a hierarchical auction-based mechanism for autonomous cloud computing robotic system negotiations within an ad hoc network setting. The solution provides fast and efficient access to cloud computing resources under a constrained bandwidth using firm real-time (FRM) communication. The solution assigns a role to each node available in the network to manage the limited communication capability of an ad hoc robot-to-robot (R2R) network. A proxy node is chosen to be responsible for distributing the resources in the cloud according to the auction results. Clients are the robots requiring cloud computing resources. Relay nodes are robots capable of maintaining connectivity for the R2R

network. Relay nodes are responsible for selecting winning clients in the latest auction round. All robots compete for the transmission opportunity through relay nodes. Node priorities are considered when allocating network resources. The objective of the game is to maximize the total transmission rate for the network. Simulation results show that the proposed algorithm outperforms the greedy and the Hungarian algorithms.

In [44], the authors proposed a peer reciprocal learning system for robots to cooperatively accomplish a complex task and help each other learn better. The cooperating robots communicate their individual decisions and collected information to formulate mutual decisions and robotic motions. A mutual learning method is proposed allowing robots to learn from each other by exchanging their neural network learning function weights. The authors claim that simulation results have shown that robots can learn from each other and build general concepts from limited training. Moreover, a real-time experiment was developed to demonstrate the effectiveness of the proposed robot learning system in achieving complex tasks.

In [45], the authors consider the issue of deploying a team of robots to accomplish a task while maintaining an ad-hoc network that supports the data transmission requirements necessary for task fulfillment. A solution is presented that considers estimation of point-to-point channels using pathloss and Gaussian process models to identify reliable point-to-point communication links. Moreover, data routing techniques are considered in determining suitable end-to-end communication routes. Motion planning is also considered to determine robot trajectories for motion control to ensure the survival of the communication network. Experimental evaluations were conducted to show that multi-robot navigation is possible with continuous end-to-end connectivity.

The authors in [46] proposed a neural dynamics approach used for complete area coverage navigation conducted by multi-robots. The neural network is used to model the work environment and then guide a swarm of robots through navigation. Each mobile robot considers other robots as moving obstacles. Robot paths are autonomously generated from the neural activity landscape of the neural network and previous robot positions. Experiments were conducted on cleaning robots to showcase the effectiveness of the robot navigation technique.

The project proposed in [47], is an open-source database available in the cloud, providing services for IoT robotic systems. IoT robotic devices from different locations in the globe can access and update this information. IoT Robotic systems can learn from other robots' experiences, behaviors and operating environments. The project's collected data focuses mainly on target recognition, navigation and intelligent services performed by robots. The architecture is composed of three layers: robot clients, cloud computing engine, and cloud computing database. Tasks involving computation are transferred to the cloud computing engine using a unified data format. The cloud computing engine interacts with the database layer to finalize the results to be sent back to the robot client. Some robot applications may not require access to the cloud computing engine but rather can access the cloud computing database directly for model matching. The requested data is returned back to the robot to complete the specific task.

Table 1 provides a summary of some cooperative IoT systems in the literature highlighting their advantages and disadvantages.

In this article we introduce a workflow-net based framework for agent cooperation of IoT devices. The framework provides an algorithm to verify similarities

TABLE I. COMPARING DIFFERENT COOPERATIVE APPROACHES FOR IOT SYSTEMS.

Application	Objective	Solution Adapted	Factors Considered	Advantages	Disadvantages
Underwater sensors [23]	Decrease energy consumption	New routing protocols are introduced, such that, sensor nodes forward data packets in a multi-hop fashion with a virtual pipeline. Nodes outside the pipeline do not forward data packets to avoid network flooding.	Number of hops, number of neighbors.	Energy-efficient, reduction in packet delay.	Increased numbers of transmitted packets.
Industrial sensors [25]	Accurate sensing and reporting	Two phases: in the first phase, each node shares its information with all other nodes. In the second phase, each node forms a cooperative data packet and sends it to the BS.	TDMA-based MAC protocol, majority rule scheme for decision making.	Information is relayed by neighbouring nodes, helps conserve power for energy-limited nodes.	Network traffic flooding.
Road vehicles [26]	Safe driving and lane changing	A vehicle driving and lane changing model is proposed to plan trajectories for vehicles in the vicinity of traffic signals in advance, by extending the Intelligent Driver Model (IDM).	Signal cycle state, distance to traffic signals, adjacent vehicles.	Reduced vehicle stopping frequency and travel time, improved road traffic throughput.	Increased numbers of transmitted packets.
Smart road vehicles [27]	Deliver continuous cloud services to smart vehicle users	Nodes are clustered according to the services they offer. Nodes that are close to one another and offer similar services are grouped together. Clusters are established using a movement and service description similarity technique. Cluster heads are selected for each cluster according to a node stability identification method in terms of neighboring nodes' link distances. Optimal user service selection is achieved by a trusted-third-party (TTP) selection method.	Service similarity, relative velocity, node link lifetime node link distance, delay, service cost, privacy.	Improved QoS and QoE, guaranteed service delivery, reduced service cost.	Increased packet overhead.
Underwater vehicles [33]	Increase node lifetime	Allocate tasks between cloud and local resources based on the objective of minimizing execution time or the price of execution. Two polynomial-time heuristics are introduced to solve the resource-allocation problem.	Communication cost, computation cost, battery capacity.	Reduced execution delay.	Cost is only reduced if the traditional cloud scheme is used.
Various robotic devices [40]	Maximize total transmission rate	A hierarchical auction-based mechanism is used. The solution assigns a role to each node available in the network. A proxy node is chosen to be responsible of distributing the resources in the cloud according to the auction results. Relay nodes are responsible of selecting winning clients in the latest auction round. All robots compete for the transmission opportunity through relay nodes.	Real-time wireless multi-hop protocol (RT-WMP), node priorities.	Reduced resource usage, reduced latency.	Solution does not consider dynamic network topology changes.
Learning land robots [42]	Enhance robot learning through cooperation	Robot peers formulate a mutual decision and mutual motions, manage their current state, and exchange image information. They keep on communicating during the whole process to ensure that the task is executed correctly. The solution is based on the peer-assisted learning theory and reciprocal learning strategy.	Image information, individual robot decisions.	Increased accuracy of robotic movements.	Solution is limited to simple scenarios.
Mobile robots [43]	End-to-end node connectivity	Solution provides connectivity among a team of robots under uncertain point-to-point wireless communication scenarios. The solution considers estimation of point-to-point channels using pathloss and Gaussian process models to identify reliable point-to-point communication links. Data routing techniques are considered to determine suitable end-to-end communication routes.	Signal strength, node velocity.	Continuous and stable end-to-end node connectivity.	Excessive time for planning.
Cleaning robots [44]	Complete area coverage	A neural network approach is used to model the work environment and then guide a swarm of robots through navigation. Each mobile robot considers other robots as moving obstacles. Robot paths are autonomously generated from the neural activity landscape of the neural network and previous robot positions.	Node position, coverage pattern, task allocation protocol.	Load balancing among nodes.	Increase in overall power consumption.

among agent capabilities discovered from different fog computing sites connected to the cloud in order to determine the possibility of IoT device cooperation with respect to a desired task. The intended task is either not achievable given the available device capabilities in the environment or would require the use of a great portion of their resources if traditional non-cloud-based solutions are adopted. Our work differentiates from the literature through the introduction of a cooperation operator that formulates the topology and structure for the set of cooperating agents. The cooperative operator turns the task composition problem defined as a set of Workflow-nets into algebraic representations to solve and optimize the problem mathematically. We consider robots as an example of collaborative IoT devices, to provide a more realistic scenario that is reader friendly. It is important to note here, that, the considered cooperative solution is not restricted to robots only, but can also be applied to any IoT device that has either sensing, computational, or storage capabilities.

3 The Cooperative Framework

Cooperation between two or more agents is frequently required to achieve a common goal, and it is only considered successful if the desired goal is attained. In the literature, resource sharing and cooperation are often used interchangeably, though there is a clear distinction between them. With resource sharing, agents try to schedule their behavior by postponing certain actions so resources are accessible by all agents at different times. This makes resource sharing simply a schedul-

ing problem, regardless of whether it is achieved through central or distributed scheduling by agents communicating to negotiate resource access. Cooperation, however, refers to agents contributing by not only sharing resources equitably, but also by combining their actions so a common goal is realized. We adopted this definition of cooperation to develop the cooperative fog computing framework in this article.

To meet a pre-defined objective, a cooperative model that incorporates a set of IoT agents and their capabilities must be defined. Workflow-net, which is an extension to the well-known Petri-net [15, 16, 48], is a preferred solution to cooperative scenarios that need to be handled concurrently [49, 50].

3.1 Preliminaries

Petri-net resembles a set of actions taken in sequence or parallel to achieve a task. Hence, a directed graph is used that incorporates both transitions t and places p. Places resemble conditions that must be satisfied for the transitions which resemble actions to be executed. As such, a place either comes before or after a transition, in which usually, the first place is not preceded by a transition, and a transition does not proceed the last place. A token which represents activities performed by transitions reside in places. Thus, a place not withholding tokens disables any transition connected to it. A transition is therefore enabled if an only if there are places (with residing tokens) connected to it as input. A transition that is enabled will execute (or fire) resulting in the removal of one or more tokens from its input places towards its output places. For a more elaborate description both textually and visually, the reader is encouraged to look at [10, 19, 51–54].

According to the definition stated above, a Petri-net is thus a tuple:

$$\mathbb{N} =< \mathbb{P}, \mathbb{T}, \mathbb{F}, \mathbb{W} > \tag{1}$$

where $\mathbb{P}$ is a set of places defining the net. Those places can be seen as preconditions to events or post conditions and results to event occurrences. $\mathbb{T}$ is a set of transitions representing event occurrences. $\mathbb{F}$ is a set that defines the topology of the net. $\mathbb{W}$ is a vector of weights for the arcs defined in $\mathbb{F}$.

$$\mathbb{P} = \{p_1, p_2, \ldots, p_n\} \tag{2}$$

and n is the number of places in $\mathbb{N}$. We define the set of transitions $\mathbb{T}$ to be:

$$\mathbb{T} = \{t_1, t_2, \ldots, t_m\} \tag{3}$$

and m is the number of transitions in $\mathbb{N}$.

Figure 2 provides an example of a petri-net with seven places and four transitions. Four of the places are marked such that T_0 and T_3 are enabled. When T_0 is fired, the token in P_0 is consumed and two tokens are produced in P_1 and P_2, resulting in T_1 and T_2 being enabled and so forth.

$\mathbb{F}$ represents the connectivity between $\mathbb{T}$ and $\mathbb{P}$. Mathematically it is represented as the cross product of the set of transitions times the places union the cross product of the places times the transitions:

$$\mathbb{F} = (\mathbb{T} \times \mathbb{P}) \cup (\mathbb{P} \times \mathbb{T}) \tag{4}$$

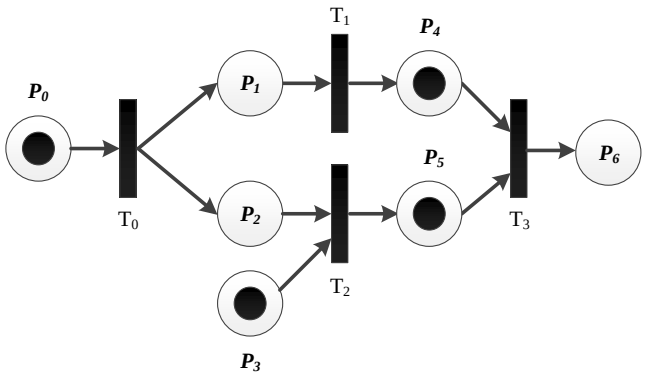

Fig. 2: A graphical example of a petri-net.

Equation 4 describes the topology of the petri-net. It shows that transitions are connected to places and places are connected to transitions. It also shows that transitions cannot be directly connected to transitions and places cannot be directly connected to places, since the cross product is between the transition set and the place set or between the place set and the transition set.

$\mathbb{W}$ is a set of integer numbers representing the weights of every $f \in \mathbb{F}$. This weight defines the number of tokens required for a transition $t \in \mathbb{T}$ to be activated through an input place $p \in \mathbb{P}$. The distribution of tokens in the net places is called marking. At any point of time τ we have token distribution $\mathbb{M}_\tau$ that is represented as follows:

$$\mathbb{M}_\tau = \bigcup_{i=1}^{n} \|p\|_\tau, p \in \mathbb{P}, \text{ at time } \tau \tag{5}$$

Equation 5 shows the distribution of tokens over the petri-net at time τ. n is the number of places and $\|p\|_\tau$ is the number of tokens in place p at time τ.

Another important vector is the firing vector. It is a vector that represents a transition that is enabled and ready to fire based on the current marking of the net. Mathematically, the firing vector $\mathbb{F}^{\rightarrow}$ is represented as follows:

$$\mathbb{F}^{\rightarrow} = \bigcup_{i=1}^{n \times m} \mathbb{M}(p) - \mathbb{W}(p,t), p \in \mathbb{P}, t \in \mathbb{T}, p \times t \notin \emptyset \tag{6}$$

In other words, for every arc between a place p and a transition t (where p is an input to t), p would contribute in enabling t if and only if the marking of p is greater than or equal to the weight of the arc between p and t. Once the transition is enabled it can eventually fire and therefore it is being added to the firing vector.

$\mathbb{F}^{\rightarrow}$ is often represented as a matrix $\hbar$ with dimensions $n \times m$. The rows are the places and the columns are the transitions. $\hbar(i,j) = 0$ if and only if $p_i \times t_j \cup t_j \times p_i = \emptyset$, $\hbar(i,j) = 1$ if and only if $t_j \times p_i \neq \emptyset$ and $\hbar(i,j) = -1$ if and only if $p_i \times t_j \neq \emptyset$. Now if we have a place p and a transition t such that $p \in {\bullet}t$ and $p \in t{\bullet}$ then this matrix has to be separated into two matrices, namely, $\hbar^+$ which holds only the relationship $t_j \times p_i \neq \emptyset$ and $\hbar^-$ which holds only the relationship $p_i \times t_j \neq \emptyset$.

3.2 Agents and Agent Capabilities

An agent multi-cloud service composition problem has been presented in [55]. However, the problem of cooperation involves multi-agents that decide to cooperate together based on their inner capabilities to achieve the required task is more complex task. This task is a set of actions defined as:

$$\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_{\|\Lambda\|}\} \tag{7}$$

The set of agents are represented as follows:

$$\mathbb{A} = \{a_1, a_2, \ldots, a_{\|\mathbb{A}\|}\} \tag{8}$$

where $\|\mathbb{A}\|$ is the length of the set of agents $\mathbb{A}$. We also have a universal set of capabilities Ω. This Ω is domain and application specific and is represented as:

$$\Omega = \{\omega_1, \omega_2, \ldots, \omega_{\|\Omega\|}\} \tag{9}$$

For every agent $a \in \mathbb{A}$, there is a set of capabilities $\Omega_a \in \Omega$. This Ω_a determines what the agent a can or cannot do. For two or more agents $a_i, a_j, \ldots, a_k$ to be able to cooperate, they need to achieve what is defined to be task coverage. That is the ability to perform the required task. We define the task coverage to be:

$$\forall \lambda \in \Lambda, \exists \omega \in \Omega_{a_i} \cup \Omega_{a_j} \cup \ldots \cup \Omega_{a_k} \| \omega \equiv \lambda \tag{10}$$

where the task is considered not covered if and only if:

$$\exists \lambda \in \Lambda, \| \forall \omega \in \Omega_{a_i} \cup \Omega_{a_j} \cup \ldots \cup \Omega_{a_k} \| \omega \not\equiv \lambda \tag{11}$$

3.3 Homogenity and Heterogenity of Agents

The concept of homogenity is applied for two or more agents to replace each other with respect to performing some task. Heterogenity is the opposite of homogeneity. Mathematically, we model homogenity to be:

$$\Lambda_k \cap \Omega_{a_i} \equiv \Lambda_k \cap \Omega_{a_j} \tag{12}$$

If equation 12 applies then a_i and a_j can replace each other to perform a certain task defined by Λ_k. This can be expressed as:

$$\forall \omega \in \Omega_{a_i}, \omega \in \Lambda_k, \omega \in \Omega_{a_j} \tag{13}$$

On the contrary, heterogenity is expressed as follows:

$$\exists \omega \in \Omega_{a_i}, \omega \in \Lambda_k \| \omega \notin \Omega_{a_j} \tag{14}$$

Similarity is a subset of homogenity. Two agents are considered similar if and only if:

$$\exists \omega \in \Omega_{a_i}, \omega \in \Lambda_k \| \omega \in \Omega_{a_j} \tag{15}$$

where a_i and a_j are similar with respect to performing action ω but not necessarily all actions in Λ_k We define agent polymorphism to be the ability for agents to replace one another given certain tasks. This means that agent polymorphism is achieved when two or more agents have similarities among each other.

It is worth noting that the proposed framework is applicable for both homogeneous and heterogeneous agents. For homogeneous agents, the selection is achieved based on availability or lower cost, and for heterogeneous agents, the selection is achieved based on the capability matrix.

3.4 Representing Cooperation Using Workflows

In order to achieve cooperation among IoT device agents that belong to different fog computing sites, the task to be performed must be taken into consideration. The design of the cooperative workflow depends on the diversity of tasks to be performed.

Aalst defined how a workflow can be modeled using petri-nets [16, 48]. We use Aalst's definition of workflows to achieve cooperation among different agents belonging to different fog computing sites after we determine polymorphism and similarities among this set of agents. We do this by proposing a **cooperation algebra** technique that mathematically models the building process of the cooperative workflow from its original basic agent capabilities. As will be seen later in this paper, we propose different mathematical operators that convert the cooperation process into a logical equation. This logical equation is translated into a workflow defined by Aalst. A few theorems are proposed to prove the correctness of the resultant workflow and the mathematical properties of those operators. An overall algorithm is being illustrated for the whole cooperation process.

IoT devices share their capabilities to achieve a cooperative plan. Unlike web service composition approaches [56, 57], we assume that compatibility is assured when agents share their capabilities. In web service composition, the objective is composed and compatible services at design time, while our approach for cooperating agents will determine the best way to share capabilities to execute a cooperative plan at minimal cost. Unlike other work in the literature, our proposed solution is a composition plan that integrates the device capabilities of different fog computing sites, as depicted in Figure 3. Fog computing nodes advertise their connected IoT device capabilities through the cloud, and nodes that are assigned tasks that require composition of cooperative agent capabilities first develop a composition plan in the form of task assignments. A set of workflow-nets are then composed to represent the composition plan, and the optimal workflow solution is adapted. Results are stored in the cloud datacenter to be used for future composition requests.

In the literature, the definition of soundness varies from Aalst [17] to Kindler [58]. Aalst clearly states that every sub-workflow that is building the overall workflow has to be sound in order for the whole workflow to be sound. Kindler, on the other hand, argues that it is important for the overall workflow to have their tokens be able to reach the output overall place for it to be sound even if we have sub-workflows that are not really sound. In this paper, we are adopting the definition of Aalst to the soundness of the workflow.

A Workflow-net is a subclass of petri-nets, such that, it has one input place and one output place. So basically, a workflow has one input gate to the system and one exit gate out of the system. A workflow must be sound by definition and by design and that is why the proof of soundness for any proposed workflow is crucial. Aalst defined a petri-net to be a workflow if and only if the following applies [58]:

1. $\exists i \in \aleph, \bullet i = \emptyset$,
2. $\exists o \in \aleph, o\bullet = \emptyset$ and
3. If $\exists t \in \mathbb{T}_{\aleph}, \bullet t = o, t\bullet = i$ then $\aleph$ becomes strongly connected.

In other words, there is a single input i to workflow $\aleph$ and a single output o to workflow $\aleph$. If the output o is connected to the input i with a feedback loop using

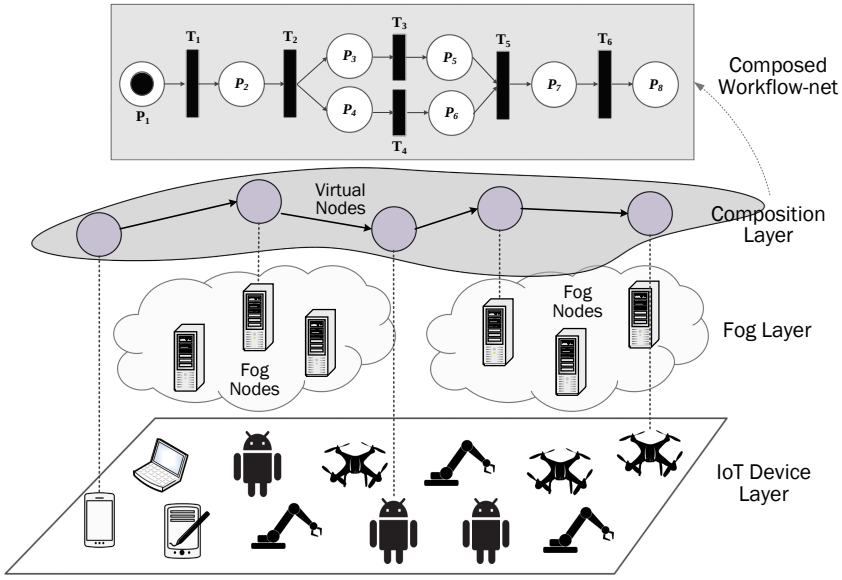

Fig. 3: Integration of IoT device capabilities to complete a task through the composition of workflow-nets.

transition t, any place in the workflow becomes reachable from any other place in the same workflow. Reachability from place a to place b is defined as having a sequence of transitions that will make b reachable from a and is denoted as $b \in [a\rangle$.

We assume that there is a set of basic and primitive actions or capabilities of agents that cannot be divided or fragmented into any smaller actions. If an agent a_i from the set of cooperating agents $\mathbb{A} = \{a_1, a_2, \ldots, a_n\}$ has plan d_j from the set of plans $\mathbb{D} = \{d_1, d_2, \ldots, d_k\}$, it can perform the plan alone only if it meets any time constraints and the following equation holds:

$$\forall \omega \in d_j : \omega \in \Omega(a_i), \tag{16}$$

where ω is an action, and

$$\Omega(a_i) = \{\mathrm{WF}_{\mathrm{net}1}, \mathrm{WF}_{\mathrm{net}2}, \ldots, \mathrm{WF}_{\mathrm{net}l}\} \tag{17}$$

where $\Omega(a_i)$ is the action capability set of agent a_i, $\mathrm{WF}_{\mathrm{net}_i}$ are workflow nets, and $d_j = \{\omega_o(\cup\omega_k)^*\}$, such that $\omega_o \neq \emptyset$ is a starting action and $(\omega_k)^*$ is a set of actions that follow (which could be the empty set $\emptyset$).

Agent a_k is a candidate for cooperation with agent a_i if and only if

$$\forall \omega \in \Delta_j : \Delta_j = d_j - \omega(a_i)\ ,\ \omega \in \Omega(a_k), \tag{18}$$

where Δ_j is the difference between the capabilities required to achieve plan d_j, and the capabilities of agent a_i. A more relaxed condition would be

$$\forall \Omega(a_k|) \cup \Omega(a_i) \equiv \Delta_j \tag{19}$$

We divide workflow into units. Units are the smallest partition of a workflow which is basically one transition, the inputs to this transition and the outputs from this transition. Mathematically a unit u_i is described as follows:

$$u_i = (\mathbb{P} \times t_i, t_i, t_i \times \mathbb{P}) \tag{20}$$

$\mathbb{P} \times t_i$ is the set of input places to t_i and is denoted $\bullet t_i$ and $t_i \times \mathbb{P}$ is the set of output places from t_i and is denoted $t_i \bullet$.

Two units u_i and u_j are considered **identical** if and only if:

$$t_i \equiv t_j \textbf{ and } \mathbb{P} \times t_i \equiv \mathbb{P} \times t_j \text{ and } t_i \times \mathbb{P} \equiv t_j \times \mathbb{P} \tag{21}$$

In other words, the action t_i does the same action that t_j does, the input set $\bullet t_i$ is the same as the input set $\bullet t_j$ and the output set $t_i \bullet$ is the same as the output set $t_j \bullet$. If we relax the condition by taking the output condition out, this would yield similar units. Two units u_i and u_j are similar if and only if:

$$t_i \equiv t_j \textbf{ and } \mathbb{P} \times t_i \equiv \mathbb{P} \times t_j \tag{22}$$

This means that all identical units are similar but not all similar units are identical.

We can proceed by induction to propose the concept of composition. We define a composition to be a set of units connected together. In this case, having two compositions c_i and c_j, such that c_i is a subset of c_j when the following equation holds:

$$c_i \subseteq c_j \leftrightarrow \forall u_i \in c_i, \exists u_j in c_j : |u_i \text{ and } u_j \text{ are identical.} \tag{23}$$

The identity of compositions are identified as follows: c_i and c_j are considered identical if and only if:

$$c_i \subseteq c_j \text{ and } c_j \subseteq c_i \tag{24}$$

and they are similar if and only if

$$c_i \subset c_j \text{ and } c_i \neq \emptyset \tag{25}$$

For two units u_i and u_j (or two compositions c_i and c_j) to cooperate, we need a communication channel ζ. ζ could be a single transition with input $\bullet\zeta = t_i\bullet$ and output $\zeta\bullet = \bullet t_j$. This is the simplest form of ζ, however, ζ could be a whole workflow-net.

Algorithm 1 shows how a cooperative framework is constructed out of separate workflows. The complexity of the algorithm is $O(w) \times O(c) \times O(a)$ where w is the number of cooperating workflows, c is the average number capabilities per workflow, and a is the average number of actions in the plan, which approximates to $O(n^3)$. For some related complexity discussions on the soundness problem of workflow-nets, the reader is encouraged to refer to [59]. The optimization of the proposed algorithm out of the scope of this paper.

It is worth noting that similarity and identity make cooperation much easier since it increases homogeneity in the swarm of agents, and therefore, selection and plan creation is easier since many alternatives are available.

Algorithm 1 Calculate Λ and $\mathbb{D}$

Require: A plan $\mathbb{P}|\mathbb{P} = (P_i \quad \theta \quad P_j)^+$ and $\theta \in \{\wedge, \vee, \rightarrow\}$ and $+$ means one or more times.
Require: $\mathbb{R}\|\mathbb{R} = \{r_1, r_2, \ldots, r_n\}$ and n is the number of agents.
Require: $\Omega\|\Omega = \{\omega_1, \omega_2, \ldots, \omega_n\}$ and ω_i is the set of capabilities of agent r_i.
Ensure: Θ , The cooperative framework.
for all $r_i \in \mathbb{R}$ **do**
  for all $r_j \in \mathbb{R}$ **do**
    if $\omega_i \cap \omega_j \neq \emptyset$ **then**
      $\mathbb{S} = \mathbb{S} \cup (r_i, r_j)$
    end if
    if $\exists P \in \mathbb{P}\|P \notin \mathbb{S}$ **then**
      TERMINATE
    end if
  end for
end for
for all $s_i \in \mathbb{S}$ **do**
  for all $P_j \in \mathbb{P}$ **do**
    if $P_j \in s_i$ **then**
      for all $r_k \in \mathbb{S}$ **do**
        $\mathbb{G}(k) = \mathbb{G}(k) \cup r_k(P)$
      end for
    end if
  end for
end for
for all $g_i \in \mathbb{G}$ **do**
  for all $p_j \in g_i$ **do**
    if $p_j \in g_i \wedge p_j \notin \Theta$ **then**
      $\Theta = \Theta \cup p_j$
    end if
    Connect p_j to p_{j-1}
  end for
end for

4 The Cooperative Operator

In order to represent cooperation mathematically, we propose a cooperative operator $\otimes$. This operator joins two or more workflows and produces a cooperative workflow based on identical and similar units composing those workflows. If $\mathrm{WF}_{\mathrm{net}_k} = \mathrm{WF}_{\mathrm{net}_i} \otimes \mathrm{WF}_{\mathrm{net}_j}$, then the following properties must apply:

1. $\mathrm{WF}_{\mathrm{net}_i}$ and $\mathrm{WF}_{\mathrm{net}_j}$ are sound,
2. $\forall\, \zeta \in \mathrm{WF}_{\mathrm{net}_i} \otimes \mathrm{WF}_{\mathrm{net}_j}$, ζ is sound, and
3. $\bullet i_k = \emptyset$, $o_k \bullet = \emptyset$.

The cooperative operator preserves the property of soundness, and is associative, non-commutative and non-distributive. Having workflow x and workflow y, the incident matrix of the cooperation is shaped as follows:

$$\hbar_{\otimes} = \begin{bmatrix} \hbar_x & \zeta \\ \zeta & \hbar_y \end{bmatrix}$$

That is if and only if $\mathbb{P}_x \cap \mathbb{P}_y = \emptyset$ and $\mathbb{T}_x \cap \mathbb{T}_y = \emptyset$. In other words, the cooperation incident matrix is valid only in cases where x and y are **disjoint**. If they are joint, then the common places and transitions are written once. The dimension of $\hbar_{\otimes}$ is calculated as follows:

$$\|\hbar\| = (\|\mathbb{P}_x\| + \|\mathbb{P}_y\| + \|\mathbb{P}_\zeta\|) \times (\|\mathbb{T}_x\| + \|\mathbb{T}_y\| + \|\mathbb{T}_\zeta\|) \tag{26}$$

4.1 The $\wedge$ Operator

The *and* operator $\wedge$ joins incident matrices of its predicates. Two workflows are joined as follows:

$$\aleph_1 \wedge \aleph_2 \rightarrow \exists t_0 \exists p_0 \exists t_e, p_e \| p_0 = \bullet t_0 \text{ and } t_0 = \bullet i_{\aleph_1} \text{ and } t_0 = \bullet i_{\aleph_2} \\ \textbf{and } p_e = t_e \bullet \text{ and } o_{\aleph_1} = \bullet p_e \text{ and } o_{\aleph_2} = \bullet p_e \tag{27}$$

In other words, the two workflows work in parallel with a single input place and a single output place. In this case, ζ is an incident matrix that binds $\aleph_1$ and $\aleph_2$ with a single input and output as an $\wedge$ operation. Figure 4 depicts an example of the behavior of the $\wedge$ operator on two workflows.

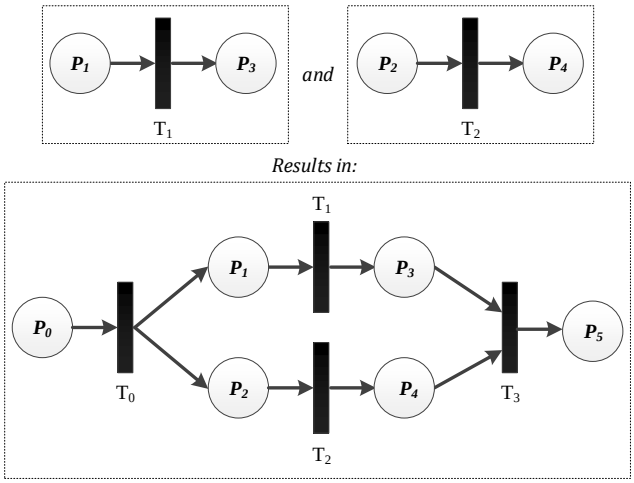

Fig. 4: An example of the behavior of the $\wedge$ operator on two workflows.

We propose the following theorem:

Theorem 1 *Having two workflows $\aleph_x$ and $\aleph_y$, if $\aleph_x$ is sound and $\aleph_y$ are sound, then, $\aleph_x \wedge \aleph_y$ is also sound.*

Proof 1 $\because \aleph_x \wedge \aleph_y =< \mathbb{P}_x \cup \mathbb{P}_y \cup \{p_0, p_e\}, \mathbb{T}_x \cup \mathbb{T}_y \cup \{t_0, t_{e1}, t_{e2}\} >$ *and* $t_0 = p_0\bullet$ *and* $p_e = t_{e1}\bullet$ *and* $p_e = t_{e2}\bullet$ *and* $t_{e1} = o_{aleph_x}\bullet$ *and* $t_{e2} = o_{\aleph_x}\bullet$.
$\therefore \forall \mathbb{M}_0, \mathbb{M}_0 = \mathbb{M}_\tau(i_{\aleph_x})$.
$\because \aleph_x$ *is sound,*
$\therefore \forall \mathbb{M}_\tau(i_{\aleph_x}) = \mathbb{M}_{\tau_2}(o_{\aleph_x})$.
$\therefore \mathbb{M}_{\tau_2}(o_{\aleph_x}) = \mathbb{M}_{\tau_2}(p_e)$
$\therefore$ *for* $\aleph_y$ $\therefore \forall \mathbb{M}_0, \mathbb{M}_0 = \mathbb{M}_\tau(i_{\aleph_y})$.
$\because \aleph_y$ *is sound,*
$\therefore \forall \mathbb{M}_\tau(i_{\aleph_y}) = \mathbb{M}_{\tau_2}(o_{\aleph_y})$.
$\therefore \mathbb{M}_{\tau_2}(o_{\aleph_y}) = \mathbb{M}_{\tau_2}(p_e)$.
$\therefore \aleph_x \wedge \aleph_y$ *is sound.*

4.2 The $\vee$ Operator

The *or* operator $\vee$ also joins the incident matrices of its predicates. Two workflows are joined as follows:

$$\aleph_1 \vee \aleph_2 \rightarrow \exists t_{01} \exists t_{02} \exists p_0 \exists t_e, p_e \| p_0 = \bullet t_{01} \text{ and } t_{01} = \bullet i_{\aleph_1} \text{ and } p_0 = \bullet t_{02} \text{ and } t_{02} = \bullet i_{\aleph_2} \textbf{ and } p_e = t_e \bullet \text{ and } o_{\aleph_1} = \bullet p_e \text{ and } o_{\aleph_2} = \bullet p_e \quad (28)$$

Contrary to the $\wedge$ operator, in the $\vee$ operator, either one of the two workflows will work at a time. The two workflows share the same input and output places. In this case, ζ is an incident matrix that binds $\aleph_1$ and $\aleph_2$ with a single input and a single output as an $\vee$ operation. Figure 5 depicts an example of the behavior of the $\vee$ operator on two workflows.

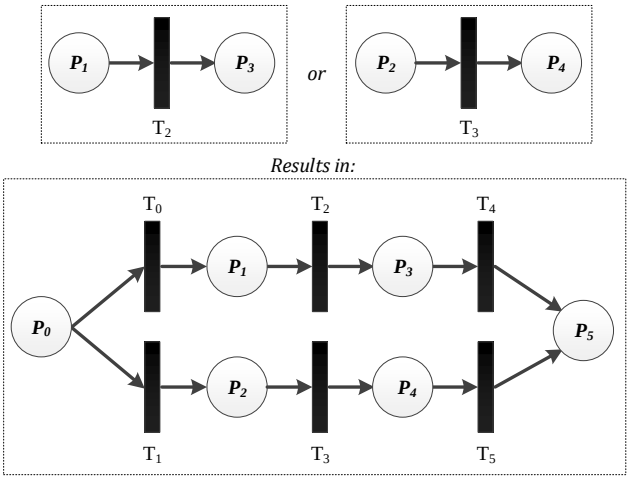

Fig. 5: An example of the behavior of the $\vee$ operator on two workflows.

We propose the following theorem:

Theorem 2 *Having two workflows* $\aleph_x$ *and* $\aleph_y$*, if* $\aleph_x$ *and* $\aleph_y$ *are sound, then* $\aleph_x \vee \aleph_y$ *is also sound.*

Proof 2 *Since* $\aleph_x \vee \aleph_y =< \mathbb{P}_x \cup \mathbb{P}_y \cup \{p_0, p_e\}, \mathbb{T}_x \cup \mathbb{T}_y \cup \{t_0, t_1, t_{e1}, t_{e2}\} >$ *and* $t_0 = p_0\bullet$ *and* $t_1 = p_0\bullet$ *and* $p_e = t_{e1}\bullet$ *and* $p_e = t_{e2}\bullet$ *and* $t_{e1} = o_{aleph_x}\bullet$ *and* $t_{e2} = o_{\aleph_x}\bullet$.
$\therefore \forall \mathbb{M}_0, \mathbb{M}_0 = \mathbb{M}_\tau(i_{\aleph_x})$ *or* $\mathbb{M}_0 = \mathbb{M}_\tau(i_{\aleph_x})$.
$\because \aleph_x$ *and* $\aleph_y$ *are sound,*
$\therefore if \exists \mathbb{M}_\tau(i_{\aleph_x}) \rightarrow \mathbb{M}_\tau(i_{\aleph_x}) = \mathbb{M}_{\tau_2}(o_{\aleph_x})$.
$\therefore if \exists \mathbb{M}_{\tau_2}(o_{\aleph_x}) \rightarrow \mathbb{M}_{\tau_2}(o_{\aleph_x}) = \mathbb{M}_{\tau_2}(o_{\aleph_y})$.
$\therefore if \exists \mathbb{M}_\tau(i_{\aleph_x}) \rightarrow \mathbb{M}_\tau(i_{\aleph_x}) = \mathbb{M}_{\tau_2}(p_e)$.
$\therefore if \exists \mathbb{M}_\tau(i_{\aleph_y}) \rightarrow \mathbb{M}_\tau(i_{\aleph_y}) = \mathbb{M}_{\tau_2}(p_e)$.
$\therefore$, $\aleph_x \vee \aleph_y$ *is sound.*

4.3 The $\rightarrow$ Operator

The *implication* operator $\rightarrow$ joins workflows as follows:

$$(\aleph_1 \rightarrow \aleph_2) \rightarrow \exists t_m \| t_m = o_{\aleph_1} \bullet \text{ and } t_m = \bullet i_{\aleph_2} \tag{29}$$

In other words, using the $\rightarrow$ yields in one of the two workflows being selected at a time. The two workflows share the same input and output places. In this case ζ is an incident matrix that binds $\aleph_1$ and $\aleph_2$ with a single input and output as an $\rightarrow$ operation. Figure 6 depicts an example of the behavior of the $\rightarrow$ operator on two workflows.

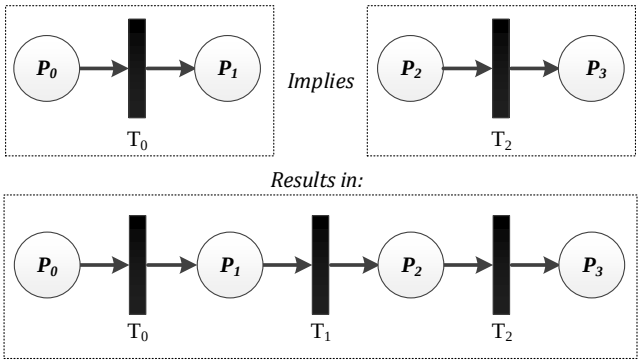

Fig. 6: An example of the behavior of the $\rightarrow$ operator on two workflows.

We propose the following theorem:

Theorem 3 *Having two workflows* $\aleph_x$ *and* $\aleph_y$*, if* $\aleph_x$ *and* $\aleph_y$ *are sound then,* $\aleph_x \rightarrow \aleph_y$ *is also sound.*

Proof 3 $\because \aleph_x$ *and* $\aleph_y$ *are sound,*
$\because \aleph_x \rightarrow \aleph_y =< \mathbb{P}_x \cup \mathbb{P}_y \mathbb{T}_x \cup \mathbb{T}_y \cup \{t_m\} >$ *and* $t_m = o_{\aleph_x}\bullet$ *and* $t_m = \bullet i_{\aleph_y}$.
$\therefore \forall \mathbb{M}_0, \mathbb{M}_0 = \mathbb{M}_\tau(o_{\aleph_x})$.

$\therefore \mathbb{M}_\tau(o_{\aleph_x}), \mathbb{M}_0 = \mathbb{M}_\tau(i_{\aleph_y})$.
$\therefore \mathbb{M}_\tau(i_{\aleph_y}) = \mathbb{M}_\tau o_{\aleph_y})$.
$\therefore \aleph_x \rightarrow \aleph_y$ *is sound.*

Certain properties hold for the cooperation operator, nameley, non-commutativity, associativity and non-distributivity.

4.4 Commutativity

We propose the following theorem:

Theorem 4 *Having two workflows* $\aleph_x$ *and* $\aleph_y$, $\aleph_x \otimes \aleph_y \not\equiv \aleph_y \otimes \aleph_x$.

Proof 4 *For* $\aleph_x$ *to cooperate with* $\aleph_y$, $\Omega_{\aleph_x} \cup \Omega_{\aleph_y} = \Lambda$.
If $\aleph_x \not\equiv \aleph_y$, $\therefore \Omega_{\aleph_x} - \Omega_{\aleph_y} \neq \Omega_{\aleph_y} - \Omega_{\aleph_x}$.
In other words, $\Lambda - \Omega_{\aleph_x} \neq \Lambda - \Omega_{\aleph_y}$. $\therefore \zeta_{xy} \neq \zeta_{yx}$.
$\therefore \aleph_x \otimes \aleph_y \not\equiv \aleph_y \otimes \aleph_x$.

4.5 Associativity

We propose the following theorem:

Theorem 5 *Having three workflows* $\aleph_x$, $\aleph_y$ *and* $\aleph_z$, $(\aleph_x \otimes \aleph_y) \otimes \aleph_z \equiv \aleph_x \otimes (\aleph_y \otimes \aleph_z)$.

Proof 5 *For* $(\aleph_x \otimes \aleph_y) \otimes \aleph_z$, *the resultant workflow is*

$$\aleph_{xy} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \mathbb{P}_{\zeta_{xy}}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \mathbb{T}_{\zeta_{xy}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_y} \cup \mathbb{F}_{\zeta_{xy}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y} \cup \mathbb{W}_{\zeta_{xy}}} > \quad (30)$$

Now, $\aleph_{xy} \otimes \aleph_z$ *gives:*
$\aleph_{xyz} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \mathbb{P}_{\aleph_z} \cup \mathbb{P}_{\zeta_{xyz}}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \mathbb{T}_{\aleph_z} \cup \mathbb{T}_{\zeta_{xyz}},$
$\mathbb{F}_{\aleph_y} \cup \mathbb{F}_{\aleph_z} \cup \mathbb{F}_{\zeta_{xyz}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y} \cup \mathbb{W}_{\zeta_{xyz}} \cup \mathbb{W}_{\zeta_{xyz}} >$
Now, for $\aleph_x \otimes (\aleph_y \otimes \aleph_z)$, *the resultant workflow is*
$\aleph_{yx} =< \mathbb{P}_{\aleph_y} \cup \mathbb{P}_{\aleph_z} \cup \mathbb{P}_{\zeta_{yz}}, \mathbb{T}_{\aleph_y} \cup \mathbb{T}_{\aleph_z} \cup \mathbb{T}_{\zeta_{yz}, \mathbb{F}_{\aleph_y} \cup \mathbb{F}_{\aleph_z} \cup \mathbb{F}_{\zeta_{yz}}, \mathbb{W}_{\aleph_y} \cup \mathbb{W}_{\aleph_z} \cup \mathbb{W}_{\zeta_{yz}}} >$
Now, $\aleph_x \otimes \aleph_{yz}$ *gives:*
$\aleph_{xyz} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \mathbb{P}_{\aleph_z} \cup \mathbb{P}_{\zeta_{xyz}}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \mathbb{T}_{\aleph_z} \cup \mathbb{T}_{\zeta_{xyz}},$
$\mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_y} \cup \mathbb{F}_{\aleph_z} \cup \mathbb{F}_{\zeta_{xyz}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y} \cup \mathbb{W}_{\zeta_{xyz}} \cup \mathbb{W}_{\zeta_{xyz}} >$
$\therefore$, *having three workflows* $\aleph_x$, $\aleph_y$ *and* $\aleph_z$, $(\aleph_x \otimes \aleph_y) \otimes \aleph_z \equiv \aleph_x \otimes (\aleph_y \otimes \aleph_z)$.

4.6 Distributivity Over the $\wedge$ Operator

We propose the following theorem:

Theorem 6 *Having three workflows* $\aleph_x$, $\aleph_y$ *and* $\aleph_z$, $(\aleph_x \otimes (\aleph_y \wedge \aleph_z)) \equiv (\aleph_x \otimes \aleph_y) \wedge (\aleph_x \otimes \aleph_z))$.

Proof 6 *For* $(\aleph_x \wedge \aleph_y) \otimes \aleph_z$*, the resultant workflow is*
$\aleph_{x\wedge y} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \{p_0, p_e\}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \{t_0, t_e\}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_y}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y} >$
which makes $\aleph_{x\wedge y} \otimes \aleph_z$ *as follows:*
$\aleph_{(x\wedge y)z} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \mathbb{P}_{\aleph_z} \cup \mathbb{P}_{\zeta_{(x\wedge y)z}} \cup \{p_0, p_e\}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \mathbb{T}_{\aleph_z} \cup \mathbb{T}_{\zeta_{(x\wedge y)z}} \cup \{t_0, t_e\}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_y} \cup \mathbb{F}_{\aleph_z} \cup \mathbb{F}_{\zeta_{(x\wedge y)z}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y}, \mathbb{W}_{\aleph_z} \cup \mathbb{W}_{\zeta_{(x\wedge y)z}} >$
Now, for $(\aleph_x \otimes \aleph_y) \wedge (\aleph_x \otimes \aleph_z))$*, the resultant workflow is :*
$\aleph_{xy} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \mathbb{P}_{\zeta_{xy}}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \mathbb{T}_{\zeta_{xy}}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_y} \cup \mathbb{F}_{\zeta_{xy}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y} \cup \mathbb{W}_{\zeta_{xy}} >$
$\aleph_{xz} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_z} \cup \mathbb{P}_{\zeta_{xz}}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_z} \cup \mathbb{T}_{\zeta_{xz}}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_z} \cup \mathbb{F}_{\zeta_{xz}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_z} \cup \mathbb{W}_{\zeta_{xz}} >$
$\aleph_{xy\wedge xz} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \mathbb{P}_{\aleph_z} \cup \mathbb{P}_{\zeta_{xy}} \cup \mathbb{P}_{\zeta_{xz}} \cup \{p_0, p_e\}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \mathbb{T}_{\aleph_z} \cup \mathbb{T}_{\zeta_{xy}} \cup \mathbb{T}_{\zeta_{xz}} \cup \{t_0, t_e\}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_y} \cup \mathbb{F}_{\aleph_z} \cup \mathbb{F}_{\zeta_{xy}} \cup \mathbb{F}_{\zeta_{xz}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y} \cup \mathbb{W}_{\aleph_z} \cup \mathbb{W}_{\zeta_{xy}} \cup \mathbb{W}_{\zeta_{xy}} >$
From the previous derivation, the difference is in ζ*.*
Since the $\wedge$ *operator only joins the two workflows through joining the two input places and two output places,* $\therefore \zeta_{(x\wedge y)z} \equiv \zeta_{xy} \cup \zeta_{xz}$ *from the separability criteria.*
$\therefore (\aleph_x \otimes (\aleph_y \wedge \aleph_z)) \equiv (\aleph_x \otimes \aleph_y) \wedge (\aleph_x \otimes \aleph_z))$.

4.7 Distributivity Over the $\vee$ Operator

We propose the following theorem:

Theorem 7 *Having three workflows* $\aleph_x$*,* $\aleph_y$ *and* $\aleph_z$*,* $(\aleph_x \otimes (\aleph_y \vee \aleph_z)) \equiv (\aleph_x \otimes \aleph_y) \vee (\aleph_x \otimes \aleph_z))$.

Proof 7 *For* $(\aleph_x \wedge \aleph_y) \otimes \aleph_z$*, The resultant workflow is*
$\aleph_{x\vee y} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \{p_0, p_e\}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \{t_{01}, t_{02}, t_{e1}, t_{e2}\}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_y}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y} >$,
which makes $\aleph_{x\vee y} \otimes \aleph_z$ *as follows:*
$\aleph_{(x\vee y)z} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \mathbb{P}_{\aleph_z} \cup \mathbb{P}_{\zeta_{(x\vee y)z}} \cup \{p_0, p_e\}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \mathbb{T}_{\aleph_z} \cup \mathbb{T}_{\zeta_{(x\vee y)z}} \cup \{t_{01}, t_{02}, t_{e1}, t_{e2}\}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_y} \cup \mathbb{F}_{\aleph_z} \cup \mathbb{F}_{\zeta_{(x\vee y)z}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y}, \mathbb{W}_{\aleph_z} \cup \mathbb{W}_{\zeta_{(x\vee y)z}} >$
Now for $(\aleph_x \otimes \aleph_y) \vee (\aleph_x \otimes \aleph_z))$*, the resultant workflow is:*
$\aleph_{xy} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \mathbb{P}_{\zeta_{xy}}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \mathbb{T}_{\zeta_{xy}}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_y} \cup \mathbb{F}_{\zeta_{xy}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y} \cup \mathbb{W}_{\zeta_{xy}} >$
$\aleph_{xz} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_z} \cup \mathbb{P}_{\zeta_{xz}}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_z} \cup \mathbb{T}_{\zeta_{xz}}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_z} \cup \mathbb{F}_{\zeta_{xz}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_z} \cup \mathbb{W}_{\zeta_{xz}} >$
$\aleph_{xy\vee xz} =< \mathbb{P}_{\aleph_x} \cup \mathbb{P}_{\aleph_y} \cup \mathbb{P}_{\aleph_z} \cup \mathbb{P}_{\zeta_{xy}} \cup \mathbb{P}_{\zeta_{xz}} \cup \{p_0, p_e\}, \mathbb{T}_{\aleph_x} \cup \mathbb{T}_{\aleph_y} \cup \mathbb{T}_{\aleph_z} \cup \mathbb{T}_{\zeta_{xy}} \cup \mathbb{T}_{\zeta_{xz}} \cup \{t_{01}, t_{02}, t_{e1}, t_{e2}\}, \mathbb{F}_{\aleph_x} \cup \mathbb{F}_{\aleph_y} \cup \mathbb{F}_{\aleph_z} \cup \mathbb{F}_{\zeta_{xy}} \cup \mathbb{F}_{\zeta_{xz}}, \mathbb{W}_{\aleph_x} \cup \mathbb{W}_{\aleph_y} \cup \mathbb{W}_{\aleph_z} \cup \mathbb{W}_{\zeta_{xy}} \cup \mathbb{W}_{\zeta_{xy}} >$
From the previous derivation, the difference is in ζ*.*
Since the $\wedge$ *operator only joins the two workflows through joining the two input places and two output places,* $\therefore$*,* $\zeta_{(x\vee y)z} \equiv \zeta_{xy} \cup \zeta_{xz}$ *from the separability criteria.*
$\therefore$*,* $(\aleph_x \otimes (\aleph_y \vee \aleph_z)) \equiv (\aleph_x \otimes \aleph_y) \vee (\aleph_x \otimes \aleph_z))$.

Now that we have discussed the cooperative operator and all its properties, we shift our focus on the scalability of the proposed framework.

4.8 Scalability of the Framework

The scalability of the proposed framework can be proven as shown through the following theorem:

Theorem 8 *Having many workflows* $\aleph_x, \aleph_y, \ldots, \aleph_z$*, if* $\aleph_x, \aleph_y, \ldots, \aleph_z$ *are sound then* $\aleph_x \otimes \aleph_y \otimes \cdots \otimes \aleph_z$ *is also sound.*

We prove this theorem by induction.

Proof 8 *Base case: From Theorems 1,2,11:* $\aleph_x \otimes \aleph_y$ *is sound if* $\aleph_x, \aleph_y$ *are sound.*
Hypothesis : Assume that $\aleph_x \otimes \aleph_y \otimes \cdots \otimes \aleph_z$ *is sound.*
Induction step : we need to prove that $\aleph_x \otimes \aleph_y \otimes \cdots \otimes \aleph_z \otimes \aleph_{z+1}$ *is also sound.*
From the hypothesis, $\aleph_x \otimes \aleph_y \otimes \cdots \otimes_z \aleph_z$ *is a sound workflow.*
If $\otimes_z \equiv \wedge$ *then* $\aleph_x \otimes \aleph_y \otimes \cdots \otimes_z \aleph_z$ *is a sound workflow from Theorem 1.*
If $\otimes_z \equiv \vee$ *then* $\aleph_x \otimes \aleph_y \otimes \cdots \otimes_z \aleph_z$ *is a sound workflow from Theorem 2.*
if $\otimes_z \equiv \rightarrow$ *then* $\aleph_x \otimes \aleph_y \otimes \cdots \otimes_z \aleph_z$ *is a sound workflow from Theorem 11.*
$\therefore$*, the framework is scalable over operator* $\otimes$*.*

4.9 Robustness of the Framework

To show that the proposed cooperative framework is correct and robust, the following criteria must be considered for the workflow: soundness, deadlock-freeness, livelock-freeness, and starvation-freeness [60,61]. Soundness implies that the model is both structurally and behaviorally well-formed. Through Theorems 1-8, we have shown that the workflow preserves the notion of soundness in different cases using the cooperative operators, hence, we can say that the overall framework preserves the notion of soundness.

Deadlock and livelock are a result of choice dependent structures. Similar to soundness, to prove the deadlock-freeness and livelock-freeness of the proposed framework, we need to prove the deadlock-freeness and livelock-freeness for every proposed operator as shown below.

Theorem 9 *Having a cooperative workflow* $\aleph = \aleph_x \wedge \aleph_y$*, if* $\aleph$ *is sound then* $\aleph$ *is also deadlock-free and Livelock-free.*

Proof 9 $\because \aleph$ *is sound,*
$\therefore \aleph_x$ *and* $\aleph_y$ *are both sound.*
$\because \aleph = (p_i, t_i) \cup \aleph_x \cup \aleph_y \cup (t_o, p_o)$ *with* $p_i = \bullet t_i$ *and* $t_i = \bullet \aleph_x$ *and* $t_i = \bullet \aleph_y$ *and* $t_o = \aleph_x \bullet$ *and* $t_o = \aleph_y \bullet$ *and* $t_o = \bullet p_o$
$\therefore \forall M \in p_i, M \in i_{\aleph_x}$ *and* $M \in i_{\aleph_y}$
$\because \aleph_x$ *and* $\aleph_y$ *are both sound,*
$\therefore \forall M \in p_i, M \in o_{\aleph_x}$ *and* $M \in o_{\aleph_y}$
$\therefore \forall M \in p_i, M \in p_o$
$\therefore \aleph$ *is deadlock-free and livelock-free.*

Theorem 10 *Having a cooperative workflow* $\aleph = \aleph_x \vee \aleph_y$*, if* $\aleph$ *is sound then* $\aleph$ *is also deadlock-free.*

Proof 10 $\because \aleph$ *is sound,*
$\therefore \aleph_x$ *and* $\aleph_y$ *are both sound.*
$\because \aleph = (p_i, t_{ix}, t_{iy}) \cup \aleph_x \cup \aleph_y \cup (t_{ox}, , t_{oy} p_o)$ *with* $p_i = \bullet t_{ix}$ *and* $p_i = \bullet t_{iy}$ *and* $t_{ix} = \bullet \aleph_x$ *and* $t_{iy} = \bullet \aleph_y$ *and* $t_{ox} = \aleph_x \bullet$ *and* $t_{oy} = \aleph_y \bullet$ *and* $t_{ox} = \bullet p_o$ *and* $t_{oy} = \bullet p_o$
$\therefore \forall M \in p_i, M \in i_{\aleph_x}$ *or* $M \in i_{\aleph_y}$
$\because \aleph_x$ *and* $\aleph_y$ *are both sound,*
$\therefore \forall M \in p_i, M \in o_{\aleph_x}$ *or* $M \in o_{\aleph_y}$
$\therefore \forall M \in p_i, M \in p_o$
$\therefore \aleph$ *is deadlock-free and livelock-free.*

Theorem 11 *Having a cooperative workflow* $\aleph = \aleph_x \rightarrow \aleph_y$*, if* $\aleph$ *is sound then* $\aleph$ *is also deadlock-free.*

Proof 11 $\because \aleph$ *is sound,*
$\therefore \aleph_x$ *and* $\aleph_y$ *are both sound.*
$\because \aleph = \aleph_x \cup t_m \cup \aleph_y$ *with* $t_m = \bullet\aleph_y$ *and* $t_m = \aleph_x\bullet$
$\therefore \forall M \in i_{\aleph_x}, M \in o_{\aleph_x}$
$\therefore \forall M \in i_{\aleph_x}, M \in i_{\aleph_y}$
$\therefore \forall M \in i_{\aleph_x}, M \in o_{\aleph_y}$
$\therefore \aleph$ *is deadlock-free and livelock-free.*

Starvation is the case when one or more transitions do not get enough resources for a token to fire. The objective of this article is to always guarantee an output of the cooperative workflow, which has already been proven by the proposed theorems. Starvation is guaranteed not to happen since workflows are proven to be sound, deadlock-free, and livelock-free. The fact that the proposed framework is starvation-free is a direct result from all the proposed theorems.

5 Evaluation Results

Both simulations and implementations were conducted on the proposed cooperation framework. Simulations are used to empirically demonstrate the correctness of the framework. On the contrary, implementations are used to test the validity of the framework.

5.1 Simulation Set-Up

A simulator was developed using C++ to implement the cooperation algorithm described in Section 3. Details regarding the simulation setup were introduced in [23, 24] and are summarized below:

- Simulator input is a linear logic expression that represents a cooperative plan using cooperative operator.
- Each input parameter represents a set of IoT agents and their capabilities. Such that, agents represent IoT devices and their resource capabilities and their rational decision-making process.
- Capabilities correspond to actions that are performed, with a cost associated to each action.
- Actions that are not part of an agent's set of capabilities have a cost set to infinity.
- Actions and their costs are assigned randomly using a uniformly distributed function.

The set of actions needed to achieve a task are used to determine the set of agents that need to cooperate. Such cooperation is used to construct a cooperative plan, that represents each agent capability and the dependency of the action execution. To simplify the problem at hand, we randomly selected a plan for execution, expressed as follows:

$$((((\lambda_A \vee \lambda_E) \rightarrow (\lambda_B \vee \lambda_C)) \rightarrow \lambda_D) \wedge (\lambda_G \rightarrow \lambda_F)) \tag{31}$$

The presented plan corresponds to seven different capabilities, such that each capability corresponds to a device, and all are needed to execute the task. A set of 50 agents (i.e. IoT device) were used in 100 simulation runs. The probability for an agent to withhold a capability was controlled manually. For instance, in one of the simulation tests, we set the probability for an agent to withhold a certain capability to 0.3. This indicates that each one of the 50 agents would have a 30% chance for possessing each of the 7 capabilities shown in Equation (31). The execution time is calculated as the total execution cost of the workflow. Time units are depicted in terms of transition costs in a workflow.

5.2 Simulation Results

Experiments were conducted to compare the execution time needed to complete up to 1,000 tasks for different plans and workflows, and Figure 7 illustrates the result of adapting four different plans. Plan[0] is a fully parallel non-cooperative plan used to execute the requested tasks, Plan[1] is a fully cooperative plan in which some form of parallelism is applied, Plan[2] is a partially cooperative plan where some tasks are performed in sequence and Plan[3] is a fully sequential non-cooperative non-parallel solution used to complete the requested tasks. The results show that the fully parallel solution completes the requested tasks faster than the other solutions. However, though there is a significant increase in terms of time, this solution has higher costs with respect to agent capability replication, since each agent must acquire the full set of capabilities needed to complete all tasks. This is considered either unrealistic, or it will cause an unacceptably high cost for each agent. In contrast, the fully cooperative solution has a similar delay for task completion, and also outperforms the other solutions in terms of the least cost for agent capabilities when the set of capabilities are divided among the set of agents. The figure also shows the delay incurred for both a partially cooperative plan and a non-cooperative sequential plan; the latter requires more time to complete the requested tasks.

Figure 8 illustrates two different cooperative workflows used to execute a plan and compares them to a fully parallel execution. As shown in the algorithm presented in Section 3, a back-tracking mechanism is adopted to choose the optimal solution among the set of workflows. In this particular example, two workflows (WF1 and WF2) are created to achieve the required tasks, and the one with the least delay (WF1) is chosen to execute the plan. Comparing WF1 to the fully parallel solution shows that their execution times are similar. As stated previously, the cooperative solution outperforms the fully parallel solution in terms of the cost required for agent capability requirements.

5.3 Implementation Experiment Set-up and Details

As stated earlier in this article, we adopted robots as an example to showcase the effectiveness of collaborative IoT devices in fog and cloud computing environments. It is important to note here, that, the considered experimental setup can be replicated to any IoT device that has either sensing, computational, or storage capabilities. We assumed there are two sets of robot clusters in two separate rooms,

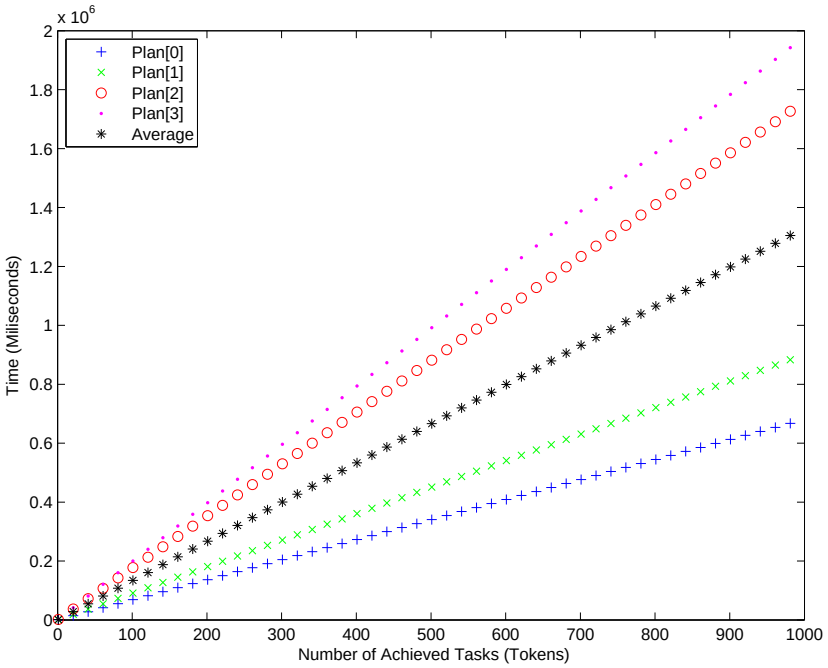

Fig. 7: Number of achieved tasks versus time required to complete tasks for four sets of plans (full parallelism (Plan[0]), full cooperation (plan[1]), partial cooperation (plan[2]), sequential (plan[3])).

as depicted in Figure 9. A map traversal problem was incorporated to compel a set of robots move from one location to another while avoiding certain obstacles (i.e. walls and furniture). The robots can communicate with a fog computing node, which publishes and requests information to/from a cloud computing storage site. The robots direction of travel, represented by a dashed arrow, indicates the determined traversal path to the destination, which is represented by a solid circle. The robots are placed in two different rooms with the same furniture and layout.

All robots are constructed from the GoPiGo robot kit [62] and connected to a Raspberry Pi 3 mini-computer [63]. The robots communicate wirelessly with each other and the fog device. We assumed that the fog device is a computer equipped with an Intel Core i7-3520M CPU, 8GB RAM and 1TB storage. Robots R1 and R4 are equipped with a Raspberry Pi camera used for office similarity detection (see Figure 10(a)). Robot R2 is equipped with an ultrasonic sensor to measure the distance between objects (see Figure 10(b)). Robots R3 and R5 do not have sensing capabilities, and can only store and process information retrieved from other robots and the fog node (see Figure 10(c)).

Since robots R3, R4 and R5 cannot move through an office in a timely manner without colliding into obstacles due to the lack of ultrasonic sensors, R4 sends images of the surrounding environment to the fog computing device. Using an image recognition algorithm [64], the fog computing device determines that the layout of the office is similar to an office associated with robots R1 and R2; thus robots R1 and R2 determine a traversal path using their sensing capabilities (camera

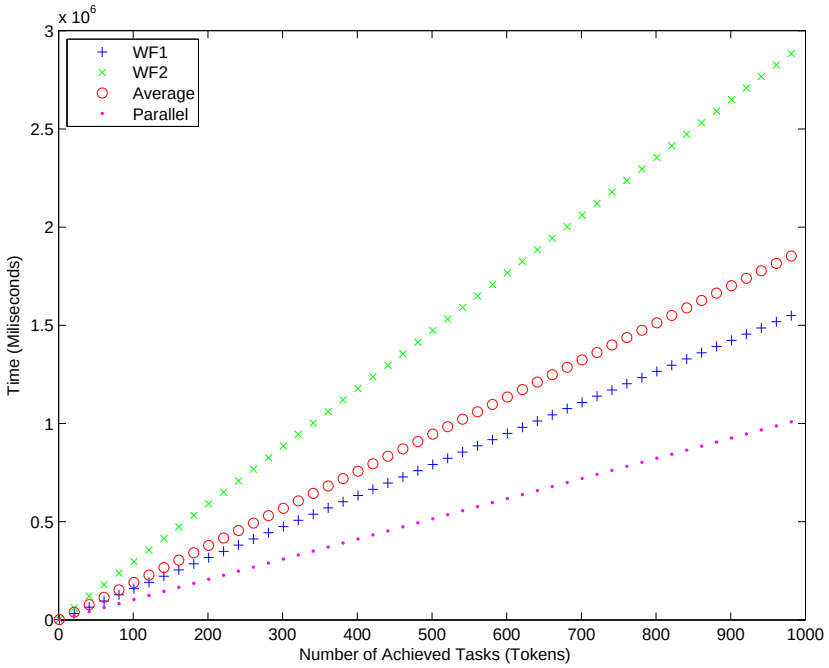

Fig. 8: Number of achieved tasks versus time required to complete tasks for a cooperative solution (WF1 and WF2) and a fully parallel solution.

and ultrasonic), and build a traversal map. Traversal maps are represented as a 2-dimensional grid of cells; a cell being a region with a predefined size. The accuracy of the grid depends on its resolution, which is the number of cells that occupy a square meter. Each cell is defined as a tuple T $|\mathbb{T} = (x, y, v)$, where (x, y) are the $x-$ and $y-$ coordinates, and v is the status of the cell, such that:

$$v = \begin{cases} \text{-1} & \text{non-explored path} \\ 0 & \text{obstacle in the path} \\ +1 & \text{safe path} \end{cases}$$

Figure 11 depicts the final merged global map of the complete office area constructed by robots R1 and R2 in our example. Maps are built by cooperating robots in a distributed approach, and a mathematical model is used to merge the learned sub-maps into one global map. We define an operator $\oplus$ as the merge operator of the sub-maps. With a robot set $\mathbb{R} = \{R_1, R_2, \ldots, R_n\}$ and a set of sub-maps $\mathbb{M} = \{m_1, m_2, \ldots, m_n\}$, the learning process outcome is $\mathbb{R} \times \mathbb{M}$. Since the outcome of the $\times$ operator is a set of multiple sub-maps, we need to merge the sub-maps such that $\mathbb{M} = m_1 \cup m_2 \cup \cdots \cup m_n$. This is performed using the $\oplus$ operator, such that $m_i \oplus m_j = \forall(x, y)|X_{min} \leq X \leq X_{max}$ and $Y_{min} \leq Y \leq Y_{max}$, where $X_{min} = MIN(m_i(x) \cup m_j(x))$, and $X_{max} = MAX(m_i(x) \cup m_j(x))$, and $Y_{min} = MIN(m_i(y) \cup m_j(y))$, and $Y_{max} = MAX(m_i(y) \cup m_j(y))$. Figure 12 is an example of two sub-maps generated initially by R1 and R2. The other

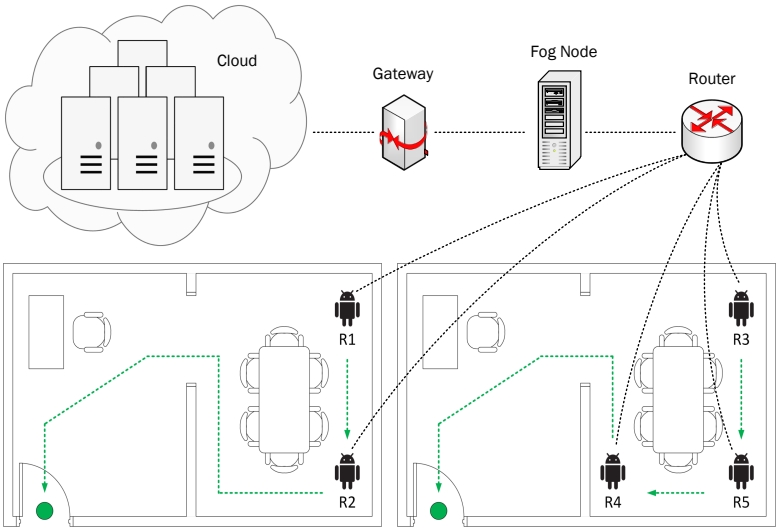

Fig. 9: Tested real-time scenario using a set of five robots belonging to two different fog computing sites. Robots R1 and R2 belong to the first cluster, and R3, R4 and R5 belong to the second cluster. The two clusters can share information and cooperate to accomplish a task.

sub-maps are generated, and a merge process is initiated to create the final global map.

An issue that could arise is having two sub-maps partially overlapping each other, with part of an area discovered by one robot also discovered by another robot, as depicted in Figure 13. An update mechanism is used for the cell status in overlapping areas, and when comparing the cell status from the two maps the new cell status v^* is determined as follows:

$$v^* = \begin{array}{c} \\ -1 \\ 0 \\ +1 \end{array} \begin{array}{c} \begin{array}{ccc} -1 & 0 & +1 \end{array} \\ \begin{bmatrix} -1 & 0 & +1 \\ 0 & 0 & 0 \\ -1 & 0 & +1 \end{bmatrix} \end{array}$$

If two robots have the same cell status, the new cell status is not modified. However, if one of the robots has a cell status of -1 (unexplored) and the other has a cell status of 0 (obstacle), the new cell status would also be 0. Similarly, if one robot has a cell status of -1 and the other has a cell status of +1 (safe path), the new cell status would also be +1. In addition, if one of the robots has a cell status of 0 and the other has a cell status of +1, the new cell status is set to 0 to avoid collisions.

We use induction to prove the scalability of the $\oplus$ operator as follows:

Base: Given two maps m_1 and m_2, $m = m_1 \oplus m_2$ from the definition given above.
Hypothesis: Let $m = m_1 \oplus m_2 \oplus \cdots \oplus m_k$.
Inductive step: Proof that $m = m_1 \oplus m_2 \oplus \cdots \oplus m_k \oplus m_{k+1}$.

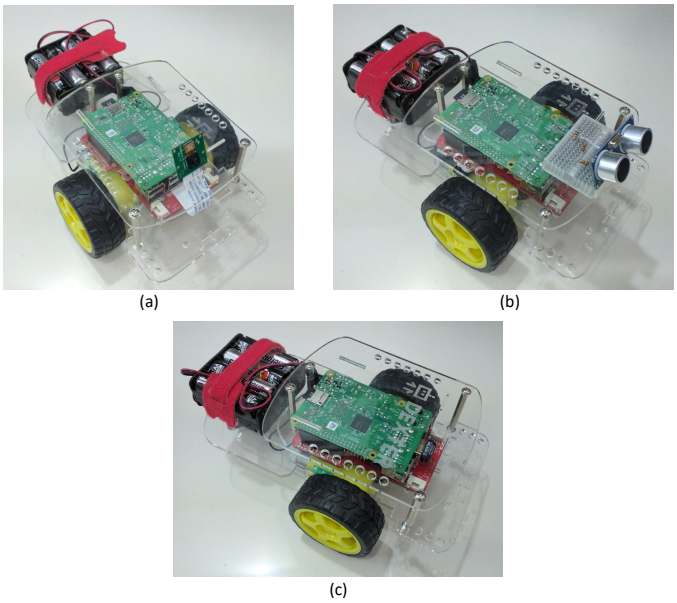

Fig. 10: Of the five robots used in the experiment, R1 and R4 are equipped with a Raspberry Pi camera, R2 is equipped with an ultrasonic sensor, and R3 and R5 do not have sensing capabilities, and are only capable of storing and processing information.

From the hypothesis, let $m_t = m_1 \oplus m_2 \oplus \cdots \oplus m_k$, therefore $m = m_1 \oplus m_2 \oplus \cdots \oplus m_k \oplus m_{k+1} = m_t \oplus m_{k+1}$. Therefore, from the base step, $m = m_t \oplus m_{k+1}$.

Once a global map is generated, it is shared with robots R3, R4 and R5 via the fog computing node. The robots cooperate, and robot R4 with a camera sensor informs robots R3 and R5 of their locations relative to the shared global map. Then a follow-the-leader traversal solution is adopted, and R3 and R5 constantly inform R4 of their cell locations. Essentially, robot R4 guides the others through the safe path.

5.4 Implementation Experiment Results

Various test sets were performed on the robots to assess the effectiveness of the cooperation method. Three techniques were adopted: i) a cooperative solution in which robots from different locations connected through the fog node cooperate to complete a task; ii) a semi-cooperative solution in which robots in the same location can only cooperate, without connection to the fog node; and iii) a non-cooperative solution in which robots must complete the task individually. The results in Figure 14 show that using the semi- and non-cooperative techniques prevents robots R3 and R5 from accomplishing the task. This is due to their inability to traverse a safe path without prior knowledge of the relative distances between them and surrounding objects, given their lack of sensing capabilities. On the contrary, robots R1 and R2, are capable of moving in the room using the semi- and non-cooperative approaches. Although sensing capabilities such as the

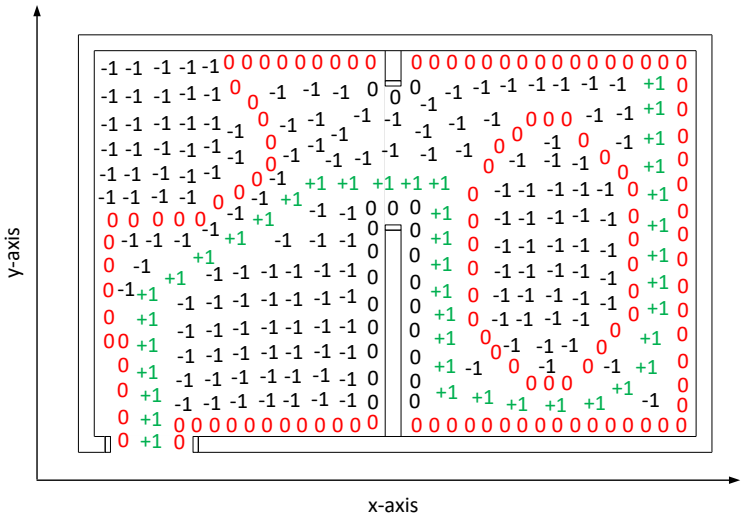

Fig. 11: A merged global map showing the explored office area, where -1 indicates a non-explored path, 0 an obstacle in the path and +1 a safe path.

camera and ultrasonic sensors allow them to find a traversal path, it takes them significantly longer. Thus, the cooperative approach is the optimal solution, with robots R3, R4 and R5 determining a safe path in the shortest time.

In addition to the task completion time test, another experiment tested the effectiveness of the proposed technique in terms of collision avoidance. The results in Figure 15 show that robot R4 had no collisions when using the cooperative technique, and the number of collisions for the robots without sensing capabilities are significantly reduced to two or three.

Path accuracy was also experimentally evaluated to test the effectiveness of the proposed solution. A robot is said to be accurate if it follows the determined path; that is, the selected set of cells from source to destination. The results in Figure 16 show that the cooperative approach achieves almost perfect accuracy for all the robots. With the semi-cooperative approach, robots R3 and R5 had very low accuracy due to their inability to precisely determine their location with respect to surrounding objects. Similarly, with the non-cooperative approach, robots R3 and R5 had an accuracy rate of only 2-3%, due to having neither on-board sensors nor communication capabilities with the other robots. The experiments showed that the cooperative approach not only accomplishes the required task, but does so in a timely, accurate and efficient manner.

6 Conclusion and Future Work

The management process of distributed IoT devices within different fog computing sites has proven to be a challenging task. The design and implementation of fog-based cooperative frameworks adapted for service delivery to IoT devices should

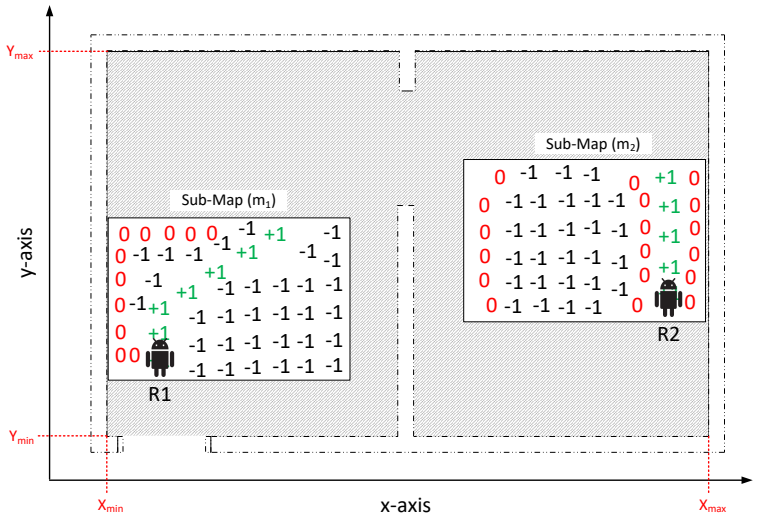

Fig. 12: Two sub-maps generated by robots R1 and R2 located within the maximum and minimum allowable coordinates of the x- and y-axis for the global map.

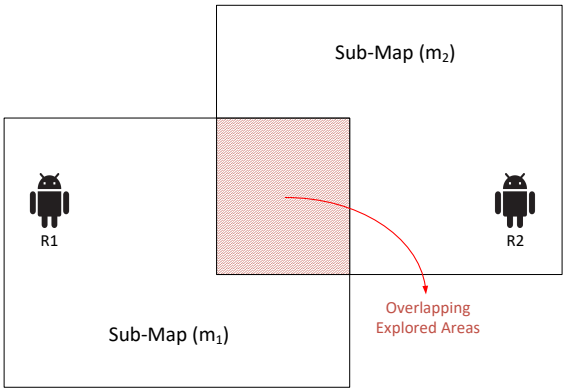

Fig. 13: Overlapping in two sub-maps.

consider the plethora of capabilities available in the cloud and fog computing environment. This article proposed a Petri-net based cooperative framework for fog-based IoT systems. The framework provided an algorithm to compose workflows based on similarities among IoT device agent capabilities that belong to different fog computing sites. It gives agents the opportunity to cooperate in order to complete the selected tasks. The dynamic behavior of the framework was verified by examining its reachability, soundness and scalability criteria. Examples of using IoT robotic agents to define and provide solutions for cooperation issues were provided throughout the article. Results gathered from both simulations and

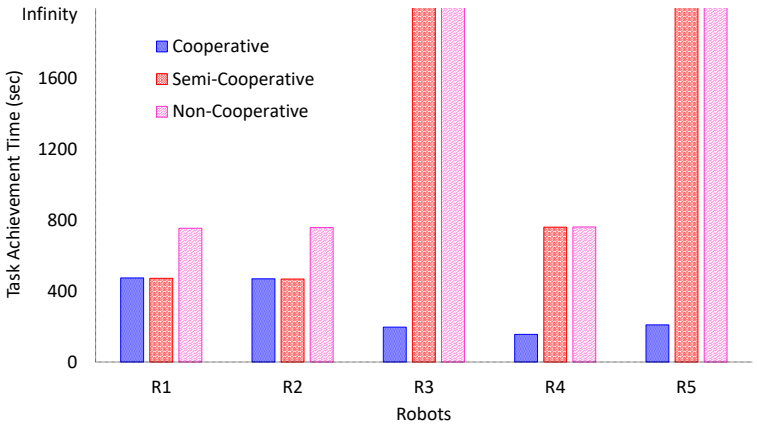

Fig. 14: Task completion time using Cooperative, Semi-Cooperative and Non-Cooperative techniques.

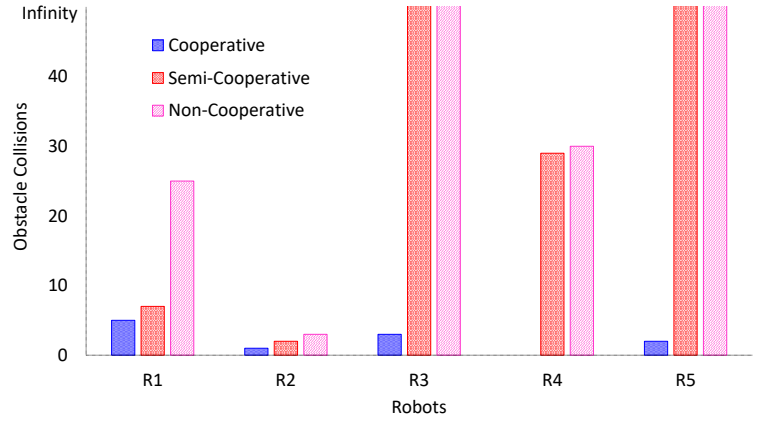

Fig. 15: Number of robot collisions with obstacles using Cooperative, Semi-Cooperative and Non-Cooperative techniques.

real-time experiments indicated that the proposed framework provides acceptable cooperation among IoT devices, in a timely and efficient manner when compared against existing work. According to the experiment results, task achievement in a cooperative approach can be achieved seven times faster with more accuracy when compared against a non-cooperative approach.

Many open research areas still remain to be investigated. One main issue that arises from cooperation and resource sharing is the willingness of different fog computing sites to cooperate. Since fog computing sites may belong to different network providers, fog computing nodes and IoT devices may not cooperate unless some form of incentives and profit sharing is guaranteed. Another issue that may arise from cooperation is security and privacy [65]. Some approaches that may be used to tackle such issue is to use artificial intelligence to analyze and classify

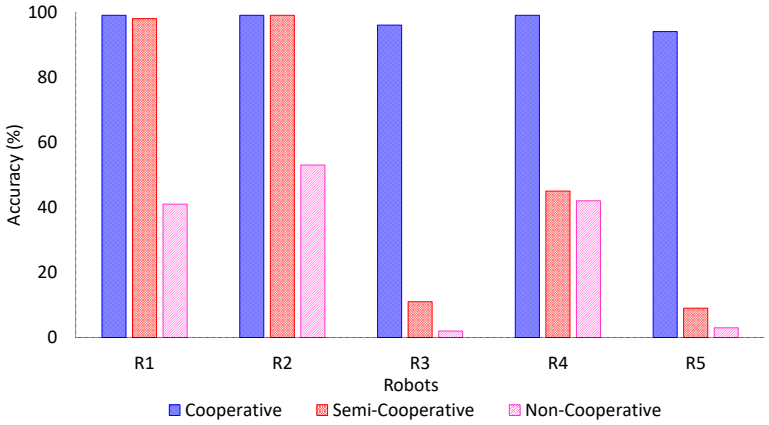

Fig. 16: Accuracy of each robot with respect to the intended traversal path using Cooperative, Semi-Cooperative and Non-Cooperative techniques.

data traffic. Although solid solutions still do not exist, we believe that in the near future, promising solutions are yet to be discovered.

References

1. M. Gharibi, R. Boutaba, and S. L. Waslander, "Internet of drones," *IEEE Access*, vol. 4, pp. 1148–1162, 2016.
2. O. Semeniuta and P. Falkman, "Flexible image acquisition service for distributed robotic systems," in *2018 Second IEEE International Conference on Robotic Computing (IRC)*, pp. 106–112, Jan 2018.
3. J. Prado, F. Yandun, M. T. Torriti, and F. A. Cheein, "Overcoming the loss of performance in unmanned ground vehichles due to the terrain variabilitiy," *IEEE Access*, pp. 1–1, 2018.
4. A. Alnasser and H. Sun, "A fuzzy logic trust model for secure routing in smart grid networks," *IEEE Access*, vol. 5, pp. 17896–17903, 2017.
5. N. Zhang, P. Yang, J. Ren, D. Chen, L. Yu, and X. Shen, "Synergy of big data and 5g wireless networks: Opportunities, approches, and challenges," *IEEE Wireless Communications*, vol. 25, pp. 12–18, February 2018.
6. G. Mohanarajah, D. Hunziker, R. D'Andrea, and M. Waibel, "Rapyuta: A cloud robotics platform," *IEEE Transactions on Automation Science and Engineering*, vol. 12, pp. 481–493, April 2015.
7. B. Krašovec and A. Filipčič, "Enhancing the grid with cloud computing," *Journal of Grid Computing*, vol. 17, pp. 119–135, Mar 2019.
8. X. Masip-Bruin, E. Marin-Todera, G. Tashakor, A. Jukan, and G. J. Ren, "Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems," *IEEE Wireless Communications*, vol. 23, pp. 120–128, October 2016.
9. X. Sun and N. Ansari, "Edgeiot: Mobile edge computing for the internet of things," *IEEE Communications Magazine*, vol. 54, pp. 22–29, December 2016.
10. I. A. Ridhawi, Y. Kotb, and Y. A. Ridhawi, "Workflow-net based service composition using mobile edge nodes," *IEEE Access*, vol. 5, pp. 23719–23735, 2017.
11. I. Shames, B. Fidan, B. D. O. Anderson, and H. Hmam, "Cooperative self-localization of mobile agents," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, pp. 1926–1947, July 2011.
12. J. Cao, X. Wang, and S. K. Das, "A framework of using cooperating mobile agents to achieve load sharing in distributed web server groups," *Future Generation Computer Systems*, vol. 20, no. 4, pp. 591 – 603, 2004. Advanced services for Clusters and Internet computing.

13. Y. Lei and Z. Junxing, "Service composition based on multi-agent in the cooperative game," *Future Generation Computer Systems*, vol. 68, pp. 128 – 135, 2017.
14. Y. Yu, R. Q. Hu, C. S. Bontu, and Z. Cai, "Mobile association and load balancing in a cooperative relay cellular network," *IEEE Communications Magazine*, vol. 49, pp. 83–89, May 2011.
15. W. M. P. van der Aalst, "The application of petri nets to workflow management," *Journal of Circuits, Systems, and Computers*, vol. 8, no. 1, pp. 21–66, 1998.
16. W. van der Aalst, "Verification of workflow nets," in *Proceedings of the 18th International Conference on Application and Theory of Petri Nets*, pp. 407–426, 1997.
17. W. M. P. van der Aalst, "Interorganizational workflows: An approach based on message sequence charts and petri nets," *Systems Science*, vol. 34, no. 3, pp. 335–367, 1999.
18. C. Savaglio, G. Fortino, M. Ganzha, M. Paprzycki, C. Bădică, and M. Ivanović, *Agent-Based Computing in the Internet of Things: A Survey*, pp. 307–320. Cham: Springer International Publishing, 2018.
19. I. A. Ridhawi, Y. Kotb, M. Aloqaily, and B. Kantarci, "A probabilistic process learning approach for service composition in cloud networks," in *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 1–6, April 2017.
20. T. Baker, O. F. Rana, R. Calinescu, R. Tolosana-Calasanz, and J. Á. Bañares, "Towards autonomic cloud services engineering via intention workflow model," in *Economics of Grids, Clouds, Systems, and Services* (J. Altmann, K. Vanmechelen, and O. F. Rana, eds.), (Cham), pp. 212–227, Springer International Publishing, 2013.
21. C. Luo, S. X. Yang, X. Li, and M. Q. H. Meng, "Neural-dynamics-driven complete area coverage navigation through cooperation of multiple mobile robots," *IEEE Transactions on Industrial Electronics*, vol. 64, pp. 750–760, Jan 2017.
22. M. Aloqaily, B. Kantarci, and H. T. Mouftah, "Fairness-aware game theoretic approach for service management in vehicular clouds," in *2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*, pp. 1–5, IEEE, 2017.
23. Y. Kotb, *Workflow-Net Based Cooperative Multi-Agent Systems*. PhD thesis, The University of Western Ontario, London, ON, 2011.
24. Y. Kotb, "Work-flow nets for multi-agent cooperation," tech. rep., London, ON, 2011.
25. N. Javaid, T. Hafeez, Z. Wadud, N. Alrajeh, M. S. Alabed, and N. Guizani, "Establishing a cooperation-based and void node avoiding energy-efficient underwater wsn for a cloud," *IEEE Access*, vol. 5, pp. 11582–11593, 2017.
26. Z. Iqbal, K. Kim, and H. Lee, "A cooperative wireless sensor network for indoor industrial monitoring," *IEEE Transactions on Industrial Informatics*, vol. 13, pp. 482–491, April 2017.
27. Y. He, D. Sun, M. Zhao, and S. Cheng, "Cooperative driving and lane changing modeling for connected vehicles in the vicinity of traffic signals: A cyber-physical perspective," *IEEE Access*, vol. 6, pp. 13891–13897, 2018.
28. I. A. Ridhawi, M. Aloqaily, B. Kantarci, Y. Jararweh, and H. T. Mouftah, "A continuous diversified vehicular cloud service availability framework for smart cities," *Computer Networks*, vol. 145, pp. 207 – 218, 2018.
29. M. Aloqaily, S. Otoum, I. Al Ridhawi, and Y. Jararweh, "An intrusion detection system for connected vehicles in smart cities," *Ad Hoc Networks*, 2019.
30. S. Otoum, B. Kantarci, and H. T. Mouftah, "On the feasibility of deep learning in sensor network intrusion detection," *IEEE Networking Letters*, 2019.
31. S. Otoum, B. Kantarci, and H. Mouftah, "Adaptively supervised and intrusion-aware data aggregation for wireless sensor clusters in critical infrastructures," in *2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2018.
32. A. A. Alkheir, M. Aloqaily, and H. T. Mouftah, "Connected and autonomous electric vehicles (caevs)," *IT Professional*, no. 6, pp. 54–61, 2018.
33. G. Fortino, C. Savaglio, and M. Zhou, "Toward opportunistic services for the industrial internet of things," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, pp. 825–830, Aug 2017.
34. V. Balasubramanian, M. Aloqaily, F. Zaman, and Y. Jararweh, "Exploring computing at the edge: A multi-interface system architecture enabled mobile device cloud," in *2018 IEEE 7th International Conference on Cloud Networking (CloudNet)*, pp. 1–4, IEEE, 2018.
35. M. Aloqaily, V. Balasubramanian, F. Zaman, I. Al Ridhawi, and Y. Jararweh, "Congestion mitigation in densely crowded environments for augmenting qos in vehicular clouds," in *Proceedings of the 8th ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications*, pp. 49–56, ACM, 2018.

36. M. Aloqaily, I. A. Ridhawi, H. B. Salameh, and Y. Jararweh, "Data and service management in densely crowded environments: Challenges, opportunities, and recent developments," *IEEE Communications Magazine*, vol. 57, pp. 81–87, April 2019.
37. S. Memon, J. Jens, E. Willem, H. Neukirchen, M. Book, and M. Riedel, "Towards federated service discovery and identity management in collaborative data and compute cloud infrastructures," *Journal of Grid Computing*, vol. 16, pp. 663–681, Dec 2018.
38. L. Chunlin, T. Jianhang, and L. Youlong, "Hybrid cloud adaptive scheduling strategy for heterogeneous workloads," *Journal of Grid Computing*, Mar 2019.
39. D. Oliveira, A. Brinkmann, N. Rosa, and P. Maciel, "Performability evaluation and optimization of workflow applications in cloud environments," *Journal of Grid Computing*, Jan 2019.
40. A. Yousefpour, G. Ishigaki, R. Gour, and J. P. Jue, "On reducing iot service delay via fog offloading," *IEEE Internet of Things Journal*, vol. 5, pp. 998–1010, April 2018.
41. M. Al-khafajiy, T. Baker, H. Al-Libawy, Z. Maamar, M. Aloqaily, and Y. Jararweh, "Improving fog computing performance via fog-2-fog collaboration," *Future Generation Computer Systems*, vol. 100, pp. 266 – 280, 2019.
42. P. Pandey, D. Pompili, and J. Yi, "Dynamic collaboration between networked robots and clouds in resource-constrained environments," *IEEE Transactions on Automation Science and Engineering*, vol. 12, pp. 471–480, April 2015.
43. L. Wang, M. Liu, and M. Q. H. Meng, "A hierarchical auction-based mechanism for real-time resource allocation in cloud robotic systems," *IEEE Transactions on Cybernetics*, vol. 47, pp. 473–484, Feb 2017.
44. T. H. S. Li, C. Y. Liu, P. H. Kuo, Y. H. Chen, C. H. Hou, H. Y. Wu, C. L. Lee, Y. B. Lin, W. H. Yen, and C. Y. Hsieh, "Reciprocal learning for robot peers," *IEEE Access*, vol. 5, pp. 6198–6211, 2017.
45. J. Fink, A. Ribeiro, and V. Kumar, "Robust control of mobility and communications in autonomous robot teams," *IEEE Access*, vol. 1, pp. 290–309, 2013.
46. C. Luo, S. X. Yang, X. Li, and M. Q. H. Meng, "Neural-dynamics-driven complete area coverage navigation through cooperation of multiple mobile robots," *IEEE Transactions on Industrial Electronics*, vol. 64, pp. 750–760, Jan 2017.
47. M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Glvez-Lpez, K. Hussermann, R. Janssen, J. M. M. Montiel, A. Perzylo, B. Schiele, M. Tenorth, O. Zweigle, and R. V. De Molengraft, "Roboearth," *IEEE Robotics Automation Magazine*, vol. 18, pp. 69–82, June 2011.
48. W. van der Aalst, V. Hee, and G. Houben, "Modelling and analysing workflow using a petri-net based approach," in *In proceeding of the second workflow on computer support cooperative work, petri nets and related formalisms*, pp. 31–50, June 1994.
49. I. Chebbi, S. Tata, and S. Dustdar, "The view-based approach to dynamic inter-organizational work-flow cooperation," Tech. Rep. TUV-1841-2004-23, Vienna University of Technology, Salzburg, Austria, 2004.
50. D. Grigori, F. Charoy, and C. Godart, "Coo-flow: A process technology to support cooperative processes," *International Journal of Software Engineering and Knowledge Engineering*, vol. 14, no. 3, 2003.
51. I. Al Ridhawi, M. Aloqaily, Y. Kotb, Y. Al Ridhawi, and Y. Jararweh, "A collaborative mobile edge computing and user solution for service composition in 5g systems," *Transactions on Emerging Telecommunications Technologies*, vol. 0, no. 0, p. e3446.
52. I. Al Ridhawi and Y. Kotb, "A secure workflow-net model for service-specific overlay networks," in *Mobile and Wireless Technologies 2017* (K. J. Kim and N. Joukov, eds.), (Singapore), pp. 389–399, Springer Singapore, 2018.
53. I. A. Ridhawi, N. Mostafa, Y. Kotb, M. Aloqaily, and I. Abualhaol, "Data caching and selection in 5g networks using f2f communication," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1–6, Oct 2017.
54. I. A. Ridhawi and Y. Kotb, "A secure service-specific overlay composition model for cloud networks," *Software Networking*, vol. 2017, no. 1, pp. 221–240, 2017.
55. P. Kendrick, T. Baker, Z. Maamar, A. Hussain, R. Buyya, and D. Al-Jumeily, "An efficient multi-cloud service composition using a distributed multiagent-based, memory-driven approach," *IEEE Transactions on Sustainable Computing*, pp. 1–1, 2019.
56. T. Wei, F. Yushun, Z. MengChu, and Z. MengChu, "A petri net-based method for compatibility analysis and composition of web services in business process execution language," *Automation Science and Engineering, IEEE Transactions on*, vol. 6, pp. 94–106, January 2009.

57. Y. Du, X. Li, and P. Xiong, "A petri net approach to mediation-aided composition of web services," *IEEE Transactions on Automation Science and Engineering*, vol. 9, pp. 429–435, April 2012.
58. E. Kindler, A. Martens, and W. Reisig, "Inter-operability of workflow applications: Local criteria for global soundness," *Lecture Notes in Computer Science, Business process management*, vol. 1806, pp. 235–253, 2000.
59. G. Liu, "Some complexity results for the soundness problem of workflow nets," *IEEE Transactions on Services Computing*, vol. 7, pp. 322–328, April 2014.
60. G. Liu, M. Zhou, and C. Jiang, "Petri net models and collaborativeness for parallel processes with resource sharing and message passing," *ACM Trans. Embed. Comput. Syst.*, vol. 16, pp. 113:1–113:20, May 2017.
61. M. Wang, G. Liu, P. Zhao, C. Yan, and C. Jiang, "Behavior consistency computation for workflow nets with unknown correspondence," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, pp. 281–291, Jan 2018.
62. "Gopigo." `https://www.dexterindustries.com/gopigo3/`. Accessed: 2018-04-17.
63. "Raspberry pi." `https://www.raspberrypi.org/`. Accessed: 2018-04-17.
64. Lemaire, T. Berger, C. Jung, I.-K. Lacroix, and Simon, "Vision-based slam: Stereo and monocular approaches," *International Journal of Computer Vision*, vol. 74, pp. 343–364, Sep 2007.
65. P. Zhang, M. Zhou, and G. Fortino, "Security and trust issues in fog computing: A survey," *Future Generation Computer Systems*, vol. 88, pp. 16 – 27, 2018.