Computing (2020) 102:1345–1363
https://doi.org/10.1007/s00607-020-00806-9

SPECIAL ISSUE ARTICLE

Enabling distributed intelligence for the Internet of Things with IOTA and mobile agents

Tariq Alsboui[1] · Yongrui Qin[1] · Richard Hill[1] · Hussain Al-Aqrabi[1]

Received: 30 July 2019 / Accepted: 18 March 2020 / Published online: 27 March 2020

Abstract

It is estimated that there will be approximately 125 billion Internet of Things (IoT) devices connected to the Internet by 2030, which are expected to generate large amounts of data. This will challenge data processing capability, infrastructure scalability, and privacy. Several studies have demonstrated the benefits of using distributed intelligence (DI) to overcome these challenges. We propose a Mobile-Agent Distributed Intelligence Tangle-Based approach (MADIT) as a potential solution based on IOTA (Tangle), where Tangle is a distributed ledger platform that enables scalable, transaction-based data exchange in large P2P networks. MADIT enables distributed intelligence at two levels. First, multiple mobile agents are employed to cater for node level communications and collect transactions data at a low level. Second, high level intelligence uses a Tangle based architecture to handle transactions. The Proof-of-Work offloading computation mechanism improves efficiency and speed of processing, while reducing energy consumption. Extensive experiments show that transaction processing speed is improved by using mobile agents, thereby providing better scalability.

Keywords Internet of Things IoT · Distributed intelligence DI · Distributed ledger technology DLT · IOTA Tangle · Mobile agent MA

✉ Tariq Alsboui
tariq.alsboui@hud.ac.uk

Yongrui Qin
y.qin2@hud.ac.uk

Richard Hill
r.hill@hud.ac.uk

Hussain Al-Aqrabi
h.al-aqrabi@hud.ac.uk

[1] School of Computing and Engineering, University of Huddersfield, Huddersfield, UK

1 Introduction

The Internet of Things (IoT) was brought to prominence by the Auto-ID centre, where Electronic Product Codes (EBC) and Radio Frequency Identification (RFID) technology automatically identified physical items in supply chains [1]. IoT is considered a novel paradigm which connects physical objects to the Internet to form ubiquitous networks that enable the sensing and modification of environments in response to dynamic stimuli [1], also referred to as Cyber-Physical Systems (CPS).

Such systems have already demonstrated the potential to enhance the quality of life by turning cities into *smart* cities [2], homes into *smart* homes [3], and campuses into *smart* campuses [4]. Research reports estimate the rapid growth of IoT; in the order of 125 billion devices connected to the Internet in 2030 [5–7]. Consequently, this presents many challenges with regard to data volume, velocity, timely processing, privacy and scalability [8,9].

Distributed Intelligence (DI) has the potential to overcome many of these challenges [10] and is a sub-discipline of artificial intelligence that distributes processing functionality, enabling collaboration between smart objects, and mediating communications to optimally support communications for IoT applications. This definition is the basis for the research described in this article.

The augmentation of capabilities to plan, reason, and solve goal-directed problems, onto CPS [11], facilitates the coordination and subsequent optimisation of complex IoT systems [12]. These systems require computational power that is local to the problem to be solved, and can also become an integral part of a much larger computational entity.

DI relies on efficient communication between interacting entities. Distributed Ledger Technologies (DLT) are emerging as platforms with considerable potential for CPSs such as IoT, by assisting the recording and verification of transactions between participating nodes without requiring a central database or authority. IOTA is an emerging DLT platform that is designed to overcome the problems of scalability, transaction fees, and mining (in the case of the blockchain technology) and is thus applicable to IoT. Central to IOTA is the *Tangle*, a Directed Acyclic Graph (DAG)[13], which provides a potentially scalable solution to enable DI with IoT.

Contribution This paper presents a Mobile Agent Distributed Intelligence Tangle-based approach (MADIT) for IoT that is capable of providing local interactions among IoT devices while offloading computation to rich resource devices to reduce energy consumption.

In summary, our key contributions are as follows:

- We propose a multi-mobile-agent Tangle-based architecture that manages resources and enables the deployment of IoT applications that are scalable and energy efficient.
- We propose a task off-loading mechanism for performing proof-of-work (PoW) on IoT devices, minimising energy consumption on resource constrained devices.
- We propose mobile agents as an efficient architectural approach to facilitate local interaction, collection and aggregation of transaction data with an efficient itinerary plan.

- We conduct a set of experiments that verify the effectiveness and benefits of the proposed approach.

The local interactions among IoT devices will be finally attached to the IOTA Tangle. We propose an integration of IOTA Tangle [13] and Mobile Agents [14] techniques, in order to realise a complete DI approach by providing low-level and high-level intelligence. Functionalities are distributed to both low-level and high-level intelligence layers. MADIT specifically recognises resource-constrained devices, which might not be able to perform the required computation at low-level. High-level computation is performed by more advanced computational devices.

This article is organized as follows: Sect. 2 identifies the motivation and challenges behind the need for distributed intelligence in the IoT era. In Sect. 3, we present the use of mobile agents to assist in enabling DI with a brief overview of the recent developments of interest. Sect. 4 presents our proposed approach, followed by a robust assessment of the performance of the proposed implementation in Sect. 5. In Sect. 5, we evaluate MADIT and compare it with alternative approaches. Sect. 6 discusses related work. Finally, Sect. 7 concludes the paper and presents future directions.

2 Motivations and challenges

IoT systems produce a massive amount of data, which creates large demands upon network resources. IoT networks typically consist of nodes that have limited resources such as constrained energy (battery or solar power), computational capability and memory storage, which makes distributed intelligence a challenging task.

2.1 Scalability

Scalability can be separated into two parts: Horizontal Scaling and Vertical Scaling. Through Horizontal Scaling, the network is expected to increase by adding more nodes. Vertical Scaling is designed to increase existing devices with additional resources such as CPU, RAM, power [15]. IoT needs to react dynamically to broader demands [6,7], and potential solutions should be scalable and can be used to deal with possibly billions of smart objects. *IOTA Tangle* [13] may provide a way to handle the rapid growth of interconnected things and scales well when the number of Tangle nodes grows.

2.2 Privacy

It is essential to build systems to keep information private, e.g., to make sure that if any unauthorised party has accessed the data, they will be unable to make sense of it. Moreover, information leakage is generally the ultimate user concern, especially relating to sensitive data, such as location, and movement trajectory information. *IOTA Masked Authenticated Messaging (MAM) protocol* [16] offers a great option to achieve privacy. For instance, IOTA MAM can be applied in healthcare applications where user data and privacy are concerned, including sensitive information about patients.

2.3 Offline capability

Offline Capability is also known as resiliency and is often defined as the capability of the system, to work in mission-critical or emergency cases, such as when an Internet connection not reachable. Therefore, there should be no need for a network to be connected to the Internet at all times. IOTA Tangle offers the capability to function while offline, but the transactions have to be re-attached to the main tangle if further processing is needed. In such cases, distributed intelligence and processing is desirable and well supported.

3 Mobile agents and distributed intelligence

Mobile agents (MAs) are software abstractions that perform data processing autonomously while physically migrating between nodes in the network to enable the sharing of data amongst participants' nodes [17]. MA facilitates the flexibility and scalability problems of centralised models [18], and are commonly deployed in Wireless Sensor Networks (WSN) for data collection and in-network processing.

Many MA approaches dispatch agents to collect data from the network rather than sending the data back to a gateway. The benefits of using MAs as stated in [19] include: reduced task redundancy, lower network bandwidth, and reduced network load. We refer the interested readers to the recent surveys in chronological order [14,20] and the references therein for a comprehensive review of the mobile agent itinerary planning approaches in WSNs.

The authors in [21] proposed a new itinerary planning strategy, which consists of three phases. First, the network is partitioned into clusters according to the distance between the sensor nodes using the k-means algorithm. Second, the number of MAs is determined for each partition based on the volume of data from each source node and the geographical distance. Third, an optimised itinerary plan is produced for each partition group, identifying the source nodes to be visited according to a greedy randomized adaptive search procedure (GRASP). This approach is scalable, and delay is minimised due to the dispatch of multiple agents for each group. However, this particular algorithm is not sufficiently robust as the data volume increases. Furthermore, the number of partitions has to be manually identified by the user, which can result in sub-optimal partitions of the network.

Similar to the above work is the approach proposed in [22], a spawn multi-mobile agent itinerary planning (SMIP) that uses the x means algorithm for defining the itinerary of the MA. After partitioning the network, the sink node is responsible for assigning a MA to each partition. They also use the concept of agent spawning, which has the ability to create a new agent that has different capacities and capabilities that are contrary to the original agent. The proposed approach achieved better performance and reduction in energy. However, the approach does not support fault tolerance in the case of node failure. This leads to an inability to decide the next hop on the fly.

In [23], a hybrid planning mechanism, mobile agent-based directed diffusion (MADD) is presented. In MADD, if the sources in the target region detect an event of interest, they flood exploratory packets to the sink individually. Based on these exploratory packets, the sink selects sources that will be visited by a mobile agent, which autonomously decides on the source-visiting sequence as it migrates among the nodes in the source-visiting set. As a result, the mobile agent follows a cost-efficient path among target sensors in MADD.

An improvement of the MADD approach is the mechanism introduced in [24]. This works according to three phases, including the controlled gradients setup phase, the exploratory data dissemination phase, and the MA action phase. In the controlled gradients phase, a sink node floods its neighbour with interest messages and sets up an itinerary towards the next hop according to two metrics; minimum hop count and threshold of remaining energy. The operation of the exploratory data dissemination phase is employed for the discovery of the source nodes as well as the setup of the TargetSrcTable (TST, which directs MA's migration routing among source nodes) in each target node. Consequently, the sensory data will be stored in each source node's cache, wait for the MA's operations in the next phase.

In the MA action phase, the MA will be created and dispatched to the identified target region, while the next hop is determined dynamically. The proposed approach is considered a hybrid approach since it uses both static and dynamic techniques. However, due to the use of a single MA, the approach lacks scalability and would result in a delay if the network is large.

More recent advanced techniques for a static itinerary is the algorithm presented in [25], named Iterated Local Search (ILS). The algorithm is centralised, and MA's itinerary is built from the sink node and only considers nodes that are reachable by the transmission range of the sink node. The sink node obtains location information from sensor nodes to estimate the physical distance amongst all node pairs. Based on this information, it finds out which nodes can communicate directly and estimates the power level to enable communication. This information is sufficient to build a network topology graph. Finally, the sink executes the Dijkstra shortest-path algorithm to calculate the communication cost among all possible SN pairs. However, for a network compromising thousands of nodes, the approach would not be scalable to accommodate growth.

Another recent hybrid approach is proposed in [26], which is a multi-agent itinerary planning based energy and fault aware data aggregation (MAEF) approach. It consists of three phases. First, a cluster head selection and cluster construction is built. Second, a cluster head-based itinerary plan that aims to select nodes in range of the sink is used by a minimum spanning tree to plan the itineraries among cluster heads. Third, the sink node dispatches a MA to gather data from the cluster head nodes. The proposed algorithm is energy efficient and scalable as efficient grouping and dispatching of multiple MAs is applied (Table 1).

Table 2 shows the typical mobile agent approaches and presents comparisons in regards to the scalability, the grouping mechanism, type of itinerary, and the delay of each approach.

From the table we can see that scalability is a critical challenge. Our work uses a new grouping mechanism of the DAG and dispatches several agents, which is also considered as a novel mechanism [14].

Table 1 Comparison among mobile agent (MA) approaches

MA approaches	Scalability	Grouping	Type of itinerary	Delay
[21]	Yes	Yes	Static	Yes
[22]	Yes	Yes	Static	No
[24]	No	No	Hybrid	Yes
[25]	No	No	Static	No
[26]	Yes	No	Hybrid	No
[23]	No	No	Hybrid	Yes

Table 2 Performance metrics for experimental work

Performance metrics	
Evaluation metrics	Definition
Transaction per second (TPS)	Refers to the number of transactions published to the Tangle network per second
Throughput	Refers to the efficiency in processing transactions in a given amount of time

4 MADIT: system architecture

The envisioned architecture, Mobile-Agent Distributed Intelligence Tangle-Based approach (MADIT), represents the novel contribution of the work and is depicted in Fig. 1. One of the key contributions of this work is the attempt to establish a baseline for a reference framework for Tangle-based MADIT that can be used to support various IoT applications.

The architecture is divided into four main parts: (1) IoT devices; (2)Tangle to process transactions(txs); (3) PoW enabled server, and; (4) Mobile Agent to carry a list of transactions data. Each IoT device is connected with neighbouring nodes via TCP/IP protocols for communication, and interactions with the Tangle are in the form of transactions. IoT devices are responsible for managing and processing the transactions. A PoW-enabled server is an IoT device that has rich resources, and is responsible for performing costly computations on behalf of IoT devices. Mobile Agents are responsible for transporting a list of transactions when visiting nodes on their routes. This is an impotent task that supports inter-node communications. The Tangle can act as a data management layer for processing and storing data in an efficient way.

4.1 Mobile agent transactions for local interactions

We have employed multiple MAs to avoid delays in reporting transaction data and to support local interactions (i.e., low-level intelligence). We consider that nodes in close proximity of each other will most likely generate similar data; therefore we apply data aggregation techniques to eliminate redundancy using a similar method as described

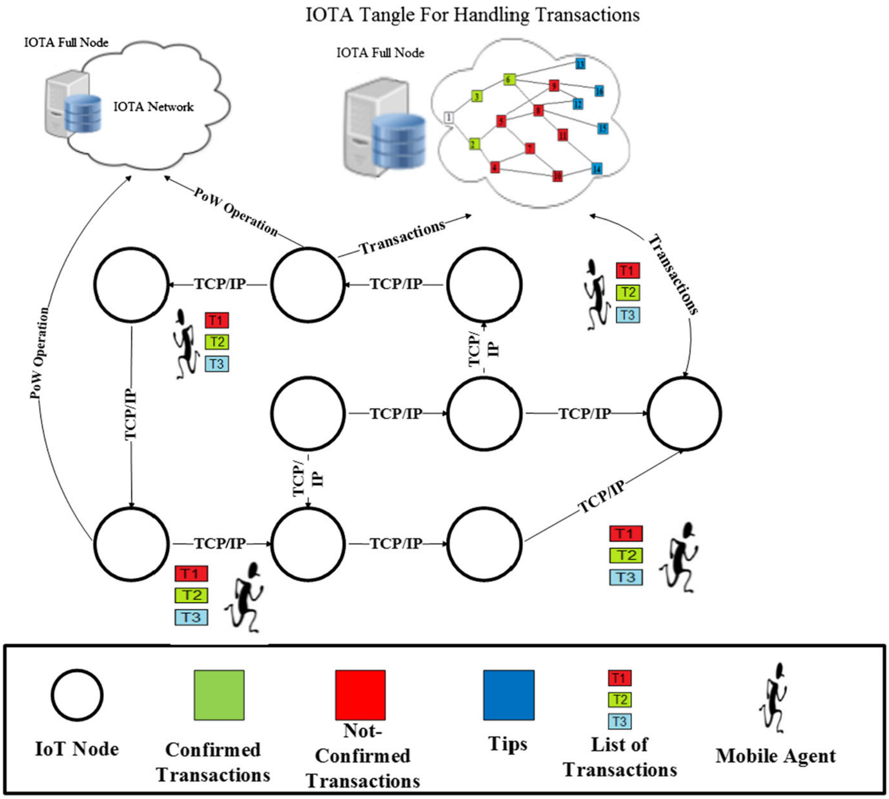

Fig. 1 The mobile agent distributed intelligence Tangle-based approach (MADIT)

in [23,27] to calculate the size of transaction data accumulated by the MA. Transaction data results are fused with an aggregation ratio ($\rho, 0 \leq \rho \leq 1$). Consider L^i_{ma} to be the amount of accumulated transactions data result after the MA finishes from source i, where A_i is the amount of transactions data to be aggregated by p, then:

$$
\begin{aligned}
L^i_{ma} &= A_i \\
L^2_{ma} &= A_i + (1-p) \times A_2 && (1)\\
L^i_{ma} &= L^i_{ma} + (1-p) \times A_2 && (2)\\
&= A_1 + \sum_{g=2}^{1} (1-p) \times A_g && (3)
\end{aligned}
$$

In Eq. (3) there will be no data aggregation in the first node and the value of p depends upon the type of deployed application.

The packet message format of the proposed MADIT is described in Fig. 2. The pair of *Itinerary Planning* and *List of transactions* are the payload of the agents. *Dispatcher ID* is used to identify the root node that creates and dispatches MA. *FirstNode*, denotes

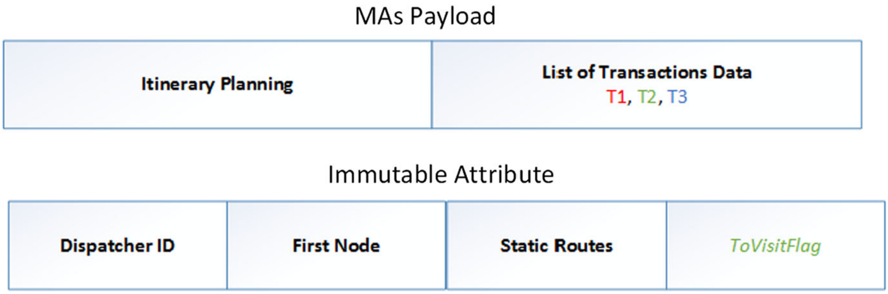

Fig. 2 Message format of the proposed (MADIT) approach

the first node that the MA will visit. *Static Routes*, denotes the computed routes for MAs with all of the assigned nodes to be visited. *ToVisitFlag*, is set to indicate that whether the node has been visited by an agent or not.

The reason for applying mobile agents in our work is not just to support low-level intelligence. It was stated in [28] that one of the most power hungry operations is radio communication; therefore, we dispatch agents to collect transactions data rather than sending it. Furthermore, to simulate a real life scenario, we assume that IoT sensor devices in proximity of each other are most likely to generate the same transactions data. Consequently, agents are also capable of eliminating redundant transactions data by fusing them.

Algorithm 1: Generate a random directed acylic graph G

```
Input: $nodeNum$, $edgeNum$
Output: $G$
Initialize $G$ to a directed acylic graph (DAG) with $nodeNum$ nodes but without
any edges, and nodes range from 0 to $nodeNum - 1$
while $edgeNum \geq 0$ do
    $node_a \leftarrow$ randint(0, $nodeNum$)
    $node_b \leftarrow node_a$
    while $node_b == node_a$ do
        $node_b \leftarrow$ randint(0, $nodeNum$)
        Add edge($node_a$, $node_b$) to $G$
        if $G$ is still DAG then
            $edgeNum \leftarrow edgeNum - 1$
        else
            Remove edge($node_a$, $node_b$) from $G$
Return $G$
```

Algorithm 1 presents the pseudocode of establishing a random DAG G. Algorithm 2 presents the pseudocode of computing the routes for all mobile agents. Algorithm 3 presents the pseudocode of dispatching multi-mobile agent to start collecting transactions data.

Initially, we introduce the establishment of a random Directed Acyclic Graph (DAG) of IoT as described in (Algorithm 1), which is designed to build a graph with a random number of Nodes and Edges. The algorithm iterates to add the required number of nodes *nodeNum*. Then, it performs a check to ensure that the graph G is a directed acyclic graph.

```
Algorithm 2: Compute mobile agent routes
  Input: DAG $G$,number of routes $N_r$
  Output: $R$
  Initialize $R$ as an empty set of mobile agent routes
  Generate $N_r$ random routes each of which traverse $G$ and add these routes to $R$
  Return $R$
```

```
Algorithm 3: Dispatch a mobile agent $MA$ to collect transactions
   Input: Visiting route $r$, mobile agent $MA$, and data load $d$
   Output: Transactions $T$ collected by $MA$
 1 Initialize $T$ as an empty set of transactions collected by $MA$
 2 while $MA$ has not completed the allocated tasks do
 3     Move to visit the next node $n$ according to the given route $r$
 4     if $n$ has been visited by any other mobile agent then
 5         Repeat Step 3, until all nodes in $r$ have been visited
 6     if all nodes in $r$ have been visited then
 7         $MA$ completes the allocated tasks
 8     else
 9         Dispatch $MA$ to visit node $n$
10         Collect transactions $T'$ (not exceeding limitation $d$ in total) from node $n$
11         Add transactions in $T'$ to $T$
12         Set $visited$ flag of node $n$ to $true$
13     if $T$ contains $d$ transactions then
14         $MA$ completes the allocated tasks
15 Return $T$
```

The compute mobile agent route Algorithm, as presented in (Algorithm 2), takes G as input from Algorithm 1 and is specifically designed to generate random routes for all mobile agents. Each route is a sequence of nodes in order to traverse G. The routes are considered as static itinerary, i.e., a pre-deterministic plan because paths for agents are planned in advance.

The algorithm that dispatches mobile agents is described in Algorithm 3. It starts by taking the following as input (1) a visiting route $r \in R$, generated by Algorithm 2, (2) a mobile agent MA, and (3) data load d for MA which is the maximum number transactions the MA can carry in one trip. Then, it initializes T as an empty set of transactions collected by MA. It starts dispatching mobile agents with a specific route in R and ensures that no two agents will follow the same route. During the trip, each MA will visit nodes according to the given route r. It will first check whether the current visiting node has been visited by any of the mobile agents or not. If the flag $visited$ of the node is $true$, the MA will move on to visit the next node on the route. Otherwise, if the current node is not visited during the same mission, the MA collects transactions data up to its data load d, and sets the flag $visited$ of the node as $true$. The MA completes the allocated tasks and returns either when all nodes on the given route have been visited, or when the MA has collected d transactions on the trip. The data load d threshold for each agent ensures that the agent buffer is not overloaded with transactions data during one single trip.

4.2 Computation offloading

Offloading can be divided into two categories: data offloading and computation offloading. The former refers to the use of novel network techniques to transmit mobile data

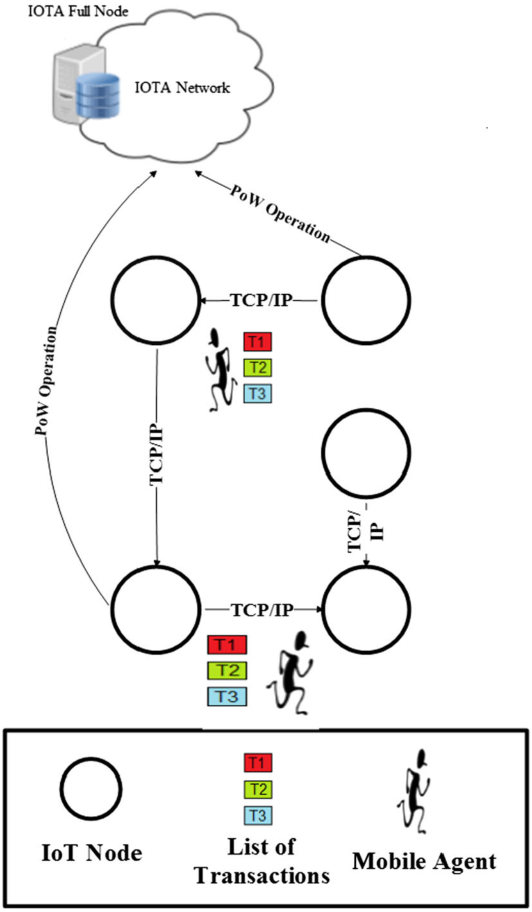

Fig. 3 Computation offloading in MADIT approach

originally planned for transferring via cellular networks. The latter refers to offloading heavy computation tasks to reserve resources [29]. The main goal of offloading is to reduce total energy consumption or overall task execution time, or both of them. A proof of work (PoW) is a piece of data that is calculated by using trial and error to meet certain requirements. The key to PoW is that it is difficult to perform but easy to verify.

Figure 3 illustrates the computation offloading mechanism used in the MADIT approach: the IOTA PoWbox (Proof of Work box). This is a service provided by the IOTA Foundation that allows the offloading of the PoW to nodes with rich resources, thus reducing energy consumption of constrained IoT devices and speeding up the development workflow [30]. Such approach was suggested by the authors in [31] in order to reserve the energy of IoT devices.

In particular, we address the problem of scalability, energy efficiency, and decentralization without loss of efficiency by adapting and integrating the IOTA Tangle and Mobile Agents. We have presented the proposed approach in view of the architecture, a consensus mechanism, and the role of MA and the computation offloading techniques employed.

5 Experiments, evaluation and analysis

In this section, we present our experimental results and an evaluation of the proposed solution in terms of scalability, energy efficiency and decentralization. Additionally, we provide analysis and discussion of the results, to establish important insights that illustrate the usefulness of IOTA Tangle integrated with Mobile Agents for the IoT domain.

5.1 Environment setup

We have deployed the latest release of the IOTA Reference Implementation (IRI 1.8.2),[1] which is the official Java build embodying the IOTA network specifications, on the DigitalOcean cloud platform,[2] and another IOTA Reference Implementation (IRI 1.8.2) on a local server dedicated for performing Proof of Work (PoW) operations.

The functionality related to IOTA addresses, transactions, broadcasting, routing, and multi-signatures has been implemented using iota.lib.py [32], the official Python library of the IOTA Distributed Ledger. Different numbers of IOTA participant nodes were used to create the network in order to simulate real life scenarios. In order to measure transaction speed and scalability, we configured each data node to generate transactions based on a time-driven technique as described in [33]. We also used a set of different Minimum Weight Magnitudes (MWM) (9, 11, 14) [34]. The reason for choosing different MWMs is due to the effect they have on the Transaction Per Second (TPS) measure. Consequently, higher a MWM will require more time in attaching transactions and hence the transactions are less likely to be selected as tips by others.

[1] https://github.com/iotaledger/iri/releases/tag/v1.8.2-RELEASE.

[2] https://www.digitalocean.com.

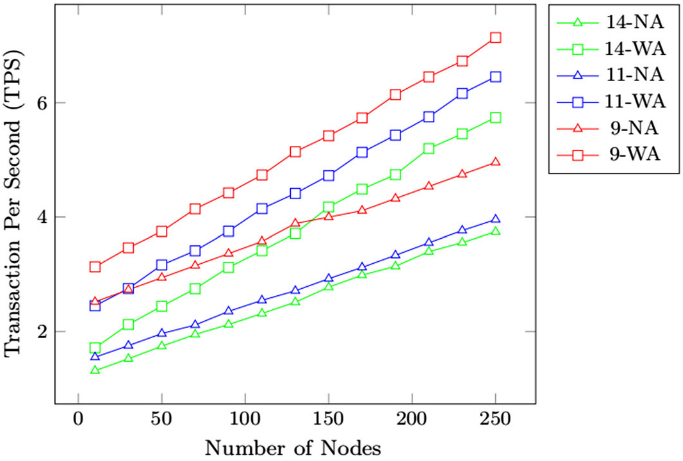

Fig. 4 Scalability in Tangle with/without mobile agents

These transactions are broadcasted and shared amongst all participant nodes. Note that, We have tested TPS for different numbers of nodes (e.g., 50, 100, 150, 250) with different MWM configurations as presented above.

5.2 Results and analysis

The following two performance metrics are used in our experiments: TPS, and Throughput.

Scalability The obtained results can be seen in Fig. 4. As shown in Fig. 4, it is clear that as the number of nodes increases, the TPS transaction speed increases linearly. For example, when the MWM is 9 and 50 nodes are engaged, with one mobile agent dispatched, as shown by the green line, the TPS of MADIT (WA denotes with mobile agents dispatched) reaches 3.749 tx/s (i.e., transactions per second) compared to the baseline (NA denotes no mobile agents dispatched) TPS, which is 2.942 tx/s. Hence, MADIT is 1.27 times faster than the baseline method. Still when the MWM is 9, and the number of nodes is 150, in this case, the average TPS with MA reaches 5.422 tx/s whereas in the baseline, TPS reaches 3.997 tx/s. This time, MADIT is 1.36 times faster than the baseline method. This demonstrates that our proposed MADIT approach is more scalable than the baseline method.

Throughput As shown in Fig. 4, it is clear that our proposed MADIT approach brings an improvement over the baseline approach in terms of efficiency in processing transactions. For example, in the situation in which 150 nodes are engaged, and the MWM is set to 14, the average TPS of baseline reaches 4.176 tx/s (shown by the red line), whereas when employing MAs, the average TPS reaches 2.776 tx/s, as shown by the green line. This is due to two factors: (1) the computation offloading mechanism, and (2) the inclusion of mobile agents in the MADIT approach.

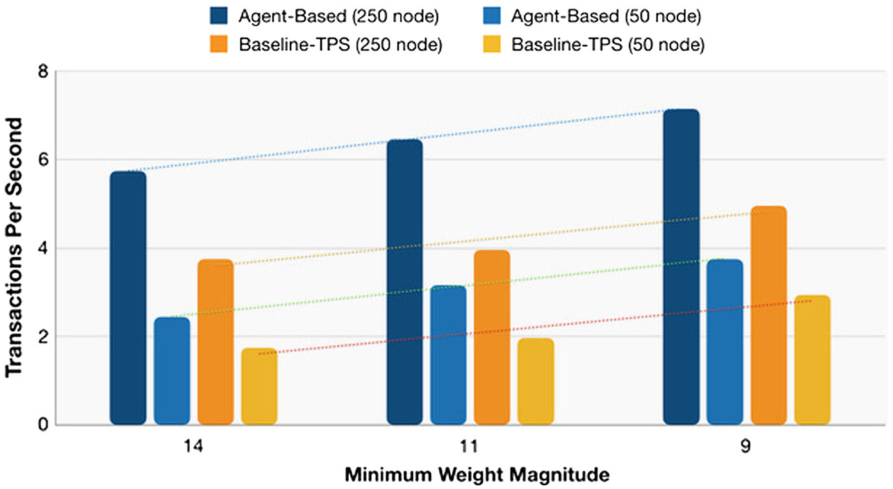

Fig. 5 Performance of baseline-TPS and agent-based under different MWM

Energy-efficiency All nodes involved in performing PoW have an impact upon total energy consumption. Therefore, computation offloading not only conserves energy but also reduces the time to process transactions. MADIT reduces energy because of the use of the offloading mechanism and an associated reduction in the number of transmissions.

Figure 5 demonstrates the effect of MWM on the TPS. In this experiment, we set the MWM to 9,11,14 to measure the effect on the TPS. In Fig. 5, it is clear that the TPS is affected by the use of different MWM configurations as when it is set to 11, it reaches 6.455 tx/s, and when it is set to 14, it reaches 7.141 tx/s.

Decentralization Our proposed MADIT approach is fully decentralized as the use of the consensus mechanism is adopted.

6 Existing distributed intelligence approaches in IoT

For the last several years, distributed intelligence has begun to attract the attention of a number of researchers from the field of IoT [10,12,35,36]. Many of these research projects address issues related to data management and processing, scalability and privacy. In earlier studies, distributed intelligence is achieved by integrating the wireless sensor network architecture with IoT to enable distributed intelligence across different layers [37,38]. These approaches aim to present a flexible architecture for connecting wireless sensor networks to the Internet and distribute intelligence and decision-making processes across different layers [39]. Such approaches are energy efficient due to the distribution of data processing, flexible, and application-agnostic. Nonetheless, there is a lack of scalability, security and offline processing capabilities that are perceived to be crucial obstacles for the IoT domain.

In order to overcome many of the inherent problems in earlier studies, the authors in [12] have introduced the concept of Sensor Function Virtualization (SFV) as a possible future technique to assist distributed intelligence in IoT. This enables distributed processing of certain functionalities by offloading them from constrained devices to unconstrained infrastructure such as a virtualized gateways, clouds and other in-network infrastructure. SFV focuses on scalability, IoT heterogeneity, and transparency. To achieve scalability, the approach relies on cloud infrastructure by allowing part of SFV functionalities to run on the cloud benefiting from the elasticity, and a tiered design. This handles the increased load when devices are joining the network. The second point refers to the heterogeneity of IoT in terms of constraining resources, and the user should be taken from the low-level information of the devices. The final point addresses simplicity in which any virtual functions that are applied to the devices must be built on top of current communication interfaces, and modifications in protocols operating on edge applications must be limited and ideally non-existent. Nevertheless, the issues of security and privacy are only narrowly considered in their approach.

The research work by [10] incorporates fog computing architecture as a method for the delivery of distributed intelligence in IoT. The suggested solution defined fog nodes in terms of both hardware architecture and software architecture. From a hardware perspective, fog nodes can be used as ancillary functions on standard network components such as gateways, edge devices and routers, or as stand-alone fog boxes. From a software perspective, fog nodes are highly virtualised machines with several VMs operating under a highly capable hypervisor. Nevertheless, fog computing still has security and privacy concerns [9,40,41].

Most recently, a current computing paradigm called Edge Mesh aims at allowing distributed intelligence in IoT and is being introduced in [35]. This paradigm distributes decision-making tasks between edge devices on a network instead of sending all data to a centralised server for further processing and analysis. Through Edge Mesh, all these massive computation tasks and data are generally exchanged using a mesh network of edge devices and routers. The Edge Mesh architecture consists of four major device types. First, end devices are primarily used for sensing and actuating. Second, edge devices can be used for pre-processing and connecting to end devices. Second, routers were used to transfer data among edge devices. Finally, the cloud is used to conduct big data analytics on historical data. Further advantages of the edge mesh approach include distributed processing, low latency, fault tolerance, as well as improved performance and scalability, better security, and privacy protection. Nevertheless, they have components to ensure security and privacy, but no consideration is given to how privacy can be achieved. Therefore, implementation and evaluation are not given.

In contrast, the work presented in [36] proposes an AI-based distributed intelligence assisted approach named as the Future Internet of Things Controller (FITC). The proposed approach uses both edge and clouds to distribute intelligence. In particular, edge controllers are used to provide low-level intelligence, and cloud based controllers to provide high-level intelligence, which they refer to as distributed intelligence. The benefits of their work are to reduce response time and loosen the requirements for rules. However, the approach lacks mechanisms that enable privacy and offline capability.

Taking their work a step further, the authors in [42] investigated the role of Mobile Edge Computing (MEC) to support distributed intelligence. The proposed approach is scalable and avoids delays. However, the system lacks the ability to work in emergency cases i.g, offline capability, and privacy is not considered in their design.

An approach named as PROTeCt–Privacy aRchitecture for the integration of the Internet of Things and Cloud computing to enable distributed intelligence is presented in [43]. The proposed approach consists of IoT devices and cloud platforms. IoT devices are responsible for sensing and implementing a cryptographic mechanism i.e., asymmetric algorithm to ensure privacy before transmitting the data to the cloud. Similarly, in [44], the authors present an approach based on Mobile Cloud Computing to support distributed intelligence. The main idea is to merge sensing and processing at different levels of the network by sharing the application's workload between the server side and the smart things, and clouds are employed when needed. However, these approaches are neither scalable nor suitable for time-critical applications. Furthermore, the resiliency of the system i.e., an offline capability is outlined as future work.

From the above, we can see that most of the existing approaches to enabling distributed intelligence in IoT suffer from inherent problems. Firstly, they rely on centralized architectures for processing data [41], which introduces a high cost and delay that is not acceptable for distributed applications. In addition, such architectures introduce inherent security vulnerabilities as data has to be transported to shared cloud resources. Such examples include health monitoring, emergency response, autonomous driving, and so on. In addition to that, they consume much network bandwidth [2], as redundant data must be moved prior to processing using remote cloud resources. It is suggested in previous research that future IoT systems need to move away from central points of control [45]. Bottlenecks and delays are to be expected from centralized systems[44]. Besides, solutions based on fog computing still have issues regarding security and privacy [9]. Moreover, there is a need for a standardized way for describing the data generated by IoT, such as the one promised by IOTA Identity of Things (IDoT) [16], which will also help secure the network. Another problem is the lack of a mechanism to describe in what form the data should be, and who can be trusted to obtain access to it (multiparty authentication scenarios), all of which are related to privacy [46,47]. Finally, only a few of the approaches facilitate the implementation and evaluation of their proposed solution.

7 Conclusion and future work

This paper advocates IOTA ***Tangle*** and Mobile Agents for supporting distributed intelligence in IoT. It presents an IOTA ***Tangle*** and Mobile Agent based approach as a solution to the problem of the limitations of traditional distributed intelligence systems. Mobile Agents deliver an efficient way of collecting transactions. The advantages of MADIT include: scalability; energy-efficiency, decentralization, elimination of redundant transaction data, and the facilitation of node level communications (low level intelligence).

There are a number of limitations in the work so far that need to be addressed in the future, for example, the cost incurred by maintaining and deploying dedicated servers for performing the PoW, location privacy and constructing a static itinerary plan for agents. As this is an emerging research field, there are a number of interesting directions for future work that researchers in relevant fields may follow.

First, how to derive a dynamic or a hybrid itinerary plan for MAs is a critical task, which allows each MA to decide the visiting sequence on-the-fly. This is particularly useful for providing fault-tolerance and can be achieved by adopting an efficient clustering method in which nodes will be grouped according to specific criteria, and MAs will be directed to a particular group as described in [14].

Second, the IOTA ***Tangle*** can be used to solve the problem of offline capability. This task is not simply a network entities configuration problem; the major issue is related to clustering the network. However, it can be achieved by creating offline ***Tangles*** where a certain number of nodes can effectively go offline and issue transactions among themselves. This means that an active internet connection is not needed while the Tangle is offline. Upon completion, it is possible to simply attach the transactions of the offline ***Tangle*** back to the online one.

Third, it would be interesting to explore Masked Authentication Messaging with a mixture of modes to enable multiparty authentication scenarios [8], and access policy. Also, location privacy [47], which are fundamental issues for the maintenance of effective IoT privacy.

Fourth, since device security is also one of the crucial fundamental challenges that determine the successful implementation of IoT applications, cyber-security [48] would be an important added improvement to the proposed MADIT approach. Ensuring the robustness of the MADIT system against hacking is critical.

Furthermore, the benefits offered by IOTA ***Tangle*** can be explored in other areas, such as Wireless Sensor Networks (WSN). It will not necessarily be pertinent to the scalability and energy-efficiency issues and undoubtedly these issues will be taken into consideration. Furthermore, how to customize IOTA ***Tangle*** to drive an efficient routing protocol for IoT, taking into consideration various factors, such as Quality of Service, would be promising. In addition to that, it would be interesting to investigate the possibility of adapting it to suit Information Extraction (IE) techniques in WSNs such as event-driven (Threshold-based), time-driven (periodic), and query-based (request-response) [33]. Therefore, not limiting the benefits of IOTA ***Tangle*** to a specific problem or problem domain.

Finally, how to design and develop a *new* programming abstraction model [49] that will suit all of the IE techniques. Consequently, it will be used as a building block in establishing an infrastructure for a *new* integrated hybrid IE framework. It will be made up of a specific, customised components and techniques along with the development of distributed algorithms from several technologies such as Network Function Virtualization (NFV) [50], Coordination Models and Languages [51], Distributed Ledger technology [52], and Micro- services [53], wrapped up with an Application Programming Interface (API).

References

1. Atzori L, Iera A, Morabito G (2010) The internet of things: a survey. Computer Networks 54(15):2787–2805
2. Perera C, Qin Y, Estrella JC, Reiff-Marganiec S, Vasilakos AV (2017) Fog computing for sustainable smart cities: a survey. ACM Comput Surv 50(3):32:1–32:43
3. Doan TT, Safavi-Naini R, Li S, Avizheh S, Muni Venkateswarlu K, Fong PWL (2018) Towards a resilient smart home. In: Proceedings of the 2018 workshop on IoT security and privacy, IoT S&P@SIGCOMM 2018, Budapest, Hungary, Aug 20 2018, pp 15–21
4. De Angelis E, Ciribini ALC, Tagliabue LC, Paneroni M (2015) The brescia smart campus demonstrator. renovation toward a zero energy classroom building. Proc Eng 118:735–743
5. Cisco. Internet of things at a glance. December 2016
6. Gartner. Gartner says the internet of things installed base will grow to 26 billion units by 2020. December 2013
7. API Research. More than 30 billion devices will wirelessly connect to the internet of everything in 2020. May 2013
8. Al-Aqrabi H, Pulikkakudi JA, Hill R, Lane P, Liu L (2019) A multi-layer security model for 5g-enabled industrial internet of things. In: 7th International Conference on Smart City and Informatization (iSCI 2019), Guangzhou, China, Nov 12–15 2019, Lecture Notes in Computer Science, Switzerland, 8. Springer International Publishing AG
9. Esposito C, Castiglione A, Pop F, Choo KR (2017) Challenges of connecting edge and cloud computing: a security and forensic perspective. IEEE Cloud Comput 4(2):13–17
10. Byers CC, Wetterwald P (2015) Fog computing distributing data and intelligence for resiliency and scale necessary for IoT: the internet of things (ubiquity symposium). Ubiquity 2015(November):41–412
11. Lynne Parker (2007) Distributed intelligence: Overview of the field and its application in multi-robot systems. In: The AAAI fall symposium series, AAAI digital library
12. Van den Abeele F, Hoebeke J, Teklemariam GK, Moerman I, Demeester P (2015) Sensor function virtualization to support distributed intelligence in the internet of things. Wirel Pers Commun 81(4):1415–1436
13. Popov Serguei. The tangle. (1), October 2017
14. Alsboui T, Alrifaee M, Etaywi R, Jawad MA (2017) Mobile agent itinerary planning approaches in wireless sensor networks- state of the art and current challenges. In: Maglaras LA, Janicke H, Jones K (eds) Ind Netw Intell Syst. Springer, Cham, pp 143–153
15. Bondi Andre B (2000) Characteristics of scalability and their impact on performance. In: Workshop on software and performance, pp 195–203
16. Valle SD (2018) Identity of thing based on iota tangle (visited on 10 Jan 2020)
17. Min C, Taekyoung K, Yuan Y, Leung V (2006) Mobile agent based wireless sensor networks. J Comput 1:04
18. Massaguer D, Fok C-L, Venkatasubramanian N, Roman G-C, Lu C (2006) Exploring sensor networks using mobile agents. In: Proceedings of the 5th international joint conference on autonomous agents and multiagent systems, AAMAS ’06. ACM, New York, pp 323–325
19. Lange DB, Oshima M (1999) Seven good reasons for mobile agents. Commun ACM 42(3):88–89
20. Venetis IE, Gavalas D, Pantziou GE, Konstantopoulos C (2018) Mobile agents-based data aggregation in wsns: benchmarking itinerary planning approaches. Wirel Netw 24(6):2111–2132
21. Aloui I, Kazar O, Kahloul L, Aissaoui A, Sylvie S (2016) A new “data size” based algorithm for itinerary planning among mobile agents in wireless sensor networks. In: Proceedings of the international conference on big data and advanced wireless technologies, BDAW ’16. ACM, New York, pp 36:1–36:9
22. Qadori H, Zukarnain Z, Zurina MH, Subramaniam S (2017) A spawn mobile agent itinerary planning approach for energy-efficient data gathering in wireless sensor networks. Sensors 17:1280, 06
23. Chen M, Kwon T, Yuan Y, Choi Y, Leung VCM (2006) Mobile agent-based directed diffusion in wireless sensor networks. EURASIP J Adv Signal Process 2007(1):036871
24. Jiang F, Shi H, Xu Z, Dong X (2009) Improved directed diffusion-based mobile agent mechanism for wireless sensor networks. In: 4th International conference on communications and networking in China, pp 1–5
25. Damianos G, Ioannis EV, Charalampos K, Grammati EP (2017) Mobile agent itinerary planning for WSN data fusion: considering multiple sinks and heterogeneous networks. Int J Commun Syst 30:1

26. El Fissaoui M, Beni-hssane A, Saadi M (2018) Multi-mobile agent itinerary planning-based energy and fault aware data aggregation in wireless sensor networks. EURASIP J Wirel Commun Netw 2018(1):92
27. Tseng Y-C, Kuo S-P, Lee H-W, Huang C-F (2003) Location tracking in a wireless sensor network by mobile agents and its data fusion strategies. In: Zhao F, Guibas L (eds) Inf Process Sens Netw. Springer, Berlin, pp 625–641
28. Pottie GJ, Kaiser WJ (2000) Wireless integrated network sensors. Commun ACM 43(5):51–58
29. Peng K, Leung V, Xiaolong X, Zheng L, Wang J, Huang Q (2018) A survey on mobile edge computing: focusing on service adoption and provision. Wirel Commun Mob Comput 2018:10
30. IOTA Foundation (2017) Minimum weight magnitude (visited on 10 Jan 2020)
31. Elsts A, Mitskas E, Oikonomou G (2018) Distributed ledger technology and the internet of things: a feasibility study. pp 7–12
32. IOTA Foundation (2018) Pyota: the iota python API library (visited on 8 Aug 2019)
33. Alsboui T, Abuarqoub A, Hammoudeh M, Bandar Z, Nisbet A (2012) Information extraction from wireless sensor networks: system and approaches. Sens Transduc 14(2):1
34. IOTA Foundation (2017) IRI configuration options. 3:15–34
35. Sahni Y, Cao J, Zhang S, Yang L (2017) Edge mesh: a new paradigm to enable distributed intelligence in internet of things. IEEE Access 5:16441–16458
36. Rahman H, Rahmani R (2018) Enabling distributed intelligence assisted future internet of things controller (FITC). Appl Comput Inf 14(1):73–87
37. Vazquez JI, Almeida A, Doamo I, Laiseca X, Orduña P (2009) Flexeo: An architecture for integrating wireless sensor networks into the internet of things. In: Corchado JM, Tapia DI, Bravo J (eds) 3rd Symposium of ubiquitous computing and ambient intelligence 2008. Springer, Berlin, pp 219–228
38. Uckelmann D, Harrison M, Michahelles F (2011) An architectural approach towards the future Internet of Things. Springer, Berlin, pp 1–24
39. Al-Aqrabi H, Hill R (2018) A secure connectivity model for internet of things analytics service delivery. In: 2018 IEEE SmartWorld, ubiquitous intelligence and computing, advanced and trusted computing, scalable computing and communications, cloud and big data computing, internet of people and smart city innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI). IEEE, pp 9–16
40. Yi S, Li C, Li Q (2015) A survey of fog computing: concepts, applications and issues. In: Proceedings of the 2015 workshop on mobile big data, Mobidata 15. ACM, New York, pp 37–42
41. Gillam L, Katsaros K, Dianati M, Mouzakitis A (2018) Exploring edges for connected and autonomous driving. In: IEEE INFOCOM 2018—IEEE conference on computer communications workshops (INFOCOM WKSHPS), pp 148–153
42. Rahman H, Rahmani R, Kanter T (2019) The role of mobile edge computing towards assisting IoT with distributed intelligence: a smartliving perspective. Springer International Publishing, Cham, pp 33–45
43. Pacheco LAB, Pelinson EA, Barreto M, Solís PA (2018) Device-based security to improve user privacy in the internet of things. In: Sensors
44. Mora H, Pont MT, Gil D, Johnsson M (2018) Collaborative working architecture for IoT-based applications. Sensors 18:1676
45. Perera C, Zaslavsky A, Christen P, Georgakopoulos D (2014) Context aware computing for the internet of things: a survey. IEEE Commun Surv Tutor 16(1):414–454
46. Al-Aqrabi H, Hill R (2019) Dynamic multiparty authentication of data analytics services within cloud environments. In: Proceedings of the 20th international conference on high performance computing and communications, 16th international conference on smart city and 4th international conference on data science and systems, HPCC/SmartCity/DSS 2018. IEEE Computer Society, pp 742–749
47. Sun G, Chang V, Ramachandran M, Sun Z, Li G, Hongfang Y, Liao D (2017) Efficient location privacy algorithm for internet of things (iot) services and applications. J Netw Comput Appl 89:3–13
48. Sohal AS, Sandhu R, Sood SK, Chang V (2018) A cybersecurity framework to identify malicious edge device in fog computing and cloud-of-things environments. Comput Secur 74:340–354
49. Mottola L, Picco GP (2011) Programming wireless sensor networks: fundamental concepts and state of the art. ACM Comput Surv 43(3):19:1–19:51
50. Zhao D, Ren J, Lin R, Xu S, Chang V (2019) On orchestrating service function chains in 5g mobile network. IEEE Access 7:39402–39416
51. Papadopoulos GA, Arbab F (1998) Coordination models and languages. Volume 46 of advances in computers. Elsevier, pp 329 – 400

52. Klimos P (2018) The distributed ledger technology: a potential revamp for financial markets? Cap Mark Law J 13(2):194–222
53. Shadija D, Rezai M, Hill R (2017) Microservices: granularity vs. performance. In: UCC 2017 Companion—companion proceedings of the 10th international conference on utility and cloud computing. Association for Computing Machinery, Inc., pp 215–220