Improving Computational Efficiency of the TONS algorithm in Selecting Neighbor Agents in Blockchain Trust-based IoT Environments

Giancarlo Fortino[a], Fabrizio Messina[b], Domenico Rosaci[c] and Giuseppe M. L. Sarné[d]

[a]DIMES, Univ. of Calabria, Via P. Bucci, 87036 Rende (CS), Italy
[b]Department of Mathematics and Computer Science, University of Catania, Italy
[c]DIIES, university Mediterranea of Reggio Calabria, Via Graziella snc, 89122 Reggio Calabria, Italy
[d]Department of Psychology, University of Milan Bicocca, Piazza dell'Ateneo Nuovo, 1, 20126 Milan, Italy

Abstract

Blockchain (BC) is increasingly applied to the Internet of Things (IoT) domains to realize decentralized IoT environments where reliable and anonymous activities can be carried out. BCs exploit a distributed ledger, which requires to be continuously synchronized and to maintain a high level of consistency. To allow BCs applied to the IoT of maintaining high levels of reliability in the network and resilience against malicious or fraudulent nodes, in the recent past has been proposed a **T**rust-based **O**ptimum **N**eighbor **S**election (TONS) algorithm able to find the Minimum Spanning Tree of the agent network with the purpose of selecting those agents which optimize communication and trustworthiness. In this paper, a new version of TONS, named TONS2, has been conceived for agent-based IoT environments to improve the overall efficiency of the TONS algorithm. The results ot the test we performed have confirmed that in terms of consumed resources TONS2 is appreciably more efficient than TONS.

Keywords

Agents, Blockchain, Internet of Things, Optimum Neighbor Selection, Trust,

1. Introduction

The great interest generated by the Internet of Things (IoT) comes from its ability to interconnect heterogeneous devices providing attractive functionality across a wide horizon of domains. [1, 2, 3]. But the large data traffic that connected IoT devices can generate may compromise the required the necessary levels of Quality of Service (QoS), due to bandwidth, storage capacity and computation constraints [4, 5].

Recently, several IoT applications have exploited the blockchain [6] (BC) to realize decentralized environments in which performing trusted and anonymous activities by providing the

WOA 2024: 25th Workshop "From Objects to Agents", July 8-10, 2024, Forte di Bard (AO), Italy
giancarlo.fortino@unical.it (G. Fortino); messina@dmi.unict.it (F. Messina); domenico.rosaci@unirc.it (D. Rosaci); giuseppe.sarne@unimib.it (G. M. L. Sarné)
https://www2.unical.it/portale/strutture/dipartimenti_240/dimes/paginedoc/gfortino/ (G. Fortino); https://web.dmi.unict.it/docenti/fabrizio.messina (F. Messina); https://www.unirc.it/scheda_persona.php?id=696 (D. Rosaci); https://www.unimib.it/giuseppe-maria-luigi-sarne (G. M. L. Sarné)
0000-0002-4039-891X (G. Fortino); 0000-0002-3685-3879 (F. Messina); 0000-0002-9256-9995 (D. Rosaci); 0000-0003-3753-6020 (G. M. L. Sarné)

required robustness against malicious attacks. In the convergence between IoT and BC [7, 8] the Distributed BC Ledgers (DLs) records transactions contemporary on more nodes without using any central data repository or global administration nodes, since each DL node independently processes, verifies and records each data item, creating a consensus on its validity. However, several events can cause different readings and compromising the consistency of a DL [9], but BCs adopt a mechanism able to warranty that data stored on the DL will remain synchronized making it consistent.

In various types of blockchains the concept of "neighbors" is relevant, particularly in decentralized and distributed ones. Nodes connect with each other as neighbors to share information, validate transactions, and maintain the security and integrity of the network. The specific terminology and role of nodes and neighbors (i.e., miners, endorsers, validators, voters, notaries, etc.) may vary depending on the type of technology and network structure.

For example, the Bitcoin mining network is designed as a peer-to-peer overlay, in which nodes, called miners, are randomly connected. Blocks are transmitted over this network using a multi-hop transmission scheme. In other words, a block creator broadcasts its newly mined block to all its neighbors [10]. Ethereum nodes randomly selecting some neighbor nodes to forward messages to ensure that transactions and status updates are propagated through the network [11]. In permissioned blockchains, where network access is restricted to pre-approved participants, the concepts of neighbors can be even more specific. In Hyperledger Fabric, nodes (peers) connect to each other within a communication channel. Validator nodes (endorsers) and commit nodes consider themselves neighbors within their channel, sharing transactions and status updates with each other [12]. Corda uses a network model where nodes interact directly with each other to execute transactions, and nodes can consider themselves neighbors if they are participating in the same transaction or contract [13]. Transactions are shared directly among the relevant participants.

In such a scenario, it is known from the literature that a high level of consistency of a BC DL will require a small number of neighbors per nodes and a low rate of delivery time between neighbors [14, 15] and, depending from the specific BC, also from the absence of unreliable malicious actors in the system. In this context, the **O**ptimum **N**eighbor **S**election (**ONS**) for a connected network can be exploited to search its Minimum Spanning Tree (MST) but, to the best of our knowledge, only [16] considered the presence of unreliable actors in an IoT environment. To this purpose, in [16] a specific algorithm, called **T**rust-based **ONS** (**TONS**), enables nodes to communicate with a globally optimized selection of the most reliable neighboring as possible, recognizes the presence of misbehaving nodes and optimizes the neighbor selection of the network based on their delivery time rates and reputation scores. TONS has been proposed and validated on networks generated on the basis of both the models Barabási–Albert (BA) and Erdős–Rényi (ER) [17, 18] and compared with the classical approach Random Neighbor Selection (RNS) [19], often used to compute ONS in blockchain networks, outperforming both in efficiency (time and number of exchanged messages to build ONS) and in effectiveness (percentage of misbehaving nodes included in the ONS).

Given this premise, the contribution of this paper consists of proposing TONS2, an improved version of the algorithm TONS described in [16], designed for agent-based IoT environments. In detail, TONS consumes significant, precious computational, storage and power resources to calculate the best set of neighbors in BC scenarios. To improve the overall efficiency of this

algorithm, TONS2 has been modified in the sense that it is not always necessary to select a new set of neighbor agents for each new BC block to validate. We tested TONS2 through a session of simulations, and the results revealed that TONS2 is more efficient than TONS, saving resources without significant performance downgrading in terms of the quality of selected neighbors.

The rest of the paper is organized as follows. Section 2 presents some related work, the Section 3 introduces the native algorithm TONS, while its evolution TONS2 is described in Section 4. The experiments we have performed to evaluate the effectiveness and the efficiency of TONS2 are presented and discussed in Section 4.1. Finally, some conclusions are drown in Section 5.

2. Related Work

Our study rely on three main elements, namely:

- A BC implements a DL, on a peer-to-peer (P2P) architecture, structured in a chain of data blocks [20], where a distributed consensus protocol enables the BC without trusted third parties and in presence of unreliable actors. BCs properties are transparency, immutability, capabilities of ensuring privacy and maintaining a complete and public transaction history [21] However, BCs are considered resilient to attacks and threats, although BCs have been tampered in the past [22].
- IoT devices have generally limited computational, storage and power capabilities, conversely IoT applications need of high level of connectivity and power to manage the large volumes of data they could generate [23, 24].
- Trust and reputation systems enable qualified parties to interact and cooperate on the basis of the history about their past behaviors [25, 26].

Synergistically, BCs can support trust (i.e., reputation) systems in managing IoT networks [27] to provide them with resilience against a large variety of attacks made easier by device characteristics. For example, TrustChain [28] provides resistance against sibyl-attacks in an online IoT community by adopting a consensus mechanism that can determine the validity and integrity of transactions instead of the more usual Proof-of-Work.

In dynamic federated social IoT environments, where IoT devices can migrate into different environments populated by potential untrusted actors, the consequences of a bad partner choice can result to be particularly risky. To mitigate this problem, in [29] multi-agent and BC technologies exploit reputation scores (witnessed by a BC) to form groups of trusted agent-based IoT devices in each federated environment. The authors of [30] designed a BC-based trust model for IoT that dynamically aggregates information sources to compute certified reputation scores for each entity in a decentralized manner by using a multilevel approach. In [31] a distributed trust model for IoT, supported by a BC, based on connected trust domains to form end-to-end trust relationships between devices is described.

With respect to the computation of the Minimum Spanning Tree (MST) search[1], a number of

[1] A MST is a minimum-weight tree that in a graph, connected, and with undirected arcs, has only the subset of arcs required to connect all vertices with each other by one and only one path.

centralized, semi-distributed or distributed MST algorithm have been proposed [32, 33, 34, 35], also in scenarios including the presence of IoT devices and BCs. For instance, DONS [30] (Dynamic and Optimized Neighboring Search) implements a hybrid architecture where the network topology is managed by an elected peer (i.e., leader) exploiting the neighbor lists to select the best neighbors for each peer by solving a MST based on the message propagation delays. Delays of the broadcast messages can make critical the synchronization of miners to add transaction blocks. In [36] an adaptive broadcasting approach is designed for BC's accounting services, authorization, and authentication combining transmission and data-check times. In [37] the time gap between the propagation of a winning block and the next mining competition is minimized; in particular a message propagation mechanism exploiting the closest neighbors by using the message latency.

Finally, in IoT scenarios, in [38] edge computing is used to mitigate system latency and bandwidth limitation, while edge computing security is provided by a BC that adopts a three-tier network model, named BMEC, exploiting a solution utilizing neural networks. In this case, the MST search is carried out on a weighted indirect graph consisting of edge-based blocks to enhance the BC transaction speed, built based on predetermined priority rules, application type, and past behaviors of edge devices.

3. The TONS algorithm

In this Section the original TONS algorithm will be summarized; however, a complete description of this algorithm can be found in [29].

3.1. The BC Network

Let $NET = \langle N, A, W \rangle$ be a BC network (an undirected, weighted and connected graph), where:

(i) N is a set of agent nodes in NET;

(ii) A is a set of arcs $a_{i,j}$, with $i, j \in N$ representing bidirectional communication between the agents, and associated with a weight $w_{i,j}$;

(iii) W is a set of weights, where $w_{i,j} = \langle t_{i,j}, \tau_{i,j} \rangle \in W$ is a tupla consisting of $t_{i,j} > 0$, named *time-weight*, that is the transmission time required to transmit 1 byte of data from the agents i and j, and $\tau_{i,j}$, named *trust-weight*, that is the mutual trustworthiness between the agents i and j.

Besides, let the positive value $ow_{i,j} = t_{i,j} \cdot \tau_{i,j}$ (i.e., the product of the pair of weights above described) be the *overall weight* of the arc $a_{i,j}$ and, similarly, let gow be the *global overall weight* (i.e., the sum of the weights of the arcs of the whole network NET).

Furthermore, it is supposed that each agent $i \in NET$ knows its neighbors $\nu_i = (\nu_{i,1}, .., \nu_{i,l})$ and their associated weights. Therefore, in the adjacency matrix D of size $|N \times N|$ each its element will be set to $ow_{i,j}$ if the arc $a_{i,j} \in NET$, while a sub-network of NET is an undirect graph $NET^*(N^*, A^*, W^*)$, such that $N^* \subseteq N$, $A^* \subseteq A$, and NET^* (where NET^* will inherit the weights of D for the graph NET).

Finally, let a *Spanning Tree* ST of NET be a connected acyclic sub-graph of NET, with $N^* = N$ and $\mid A^* \mid = \mid A \mid -1$, and let a *Minimum Spanning Tree* MST of NET be the unique Spanning Tree having the minimum gow of MST among all the Spanning Trees of NET.

TONS has been developed to realize the *Optimum Neighbor Selection* (ONS_i) finding the subset $h_i = (h_1, ..h_n) \in \nu_i, \forall i \in N$ underlying the BC network, such that $a_{i,h} \in MST_{NET}, \ \forall h \in h_i$. It is executed periodically to maintain updated the list of the agents in NET by exploiting the MST computations by one of agents (i.e., the leader, which is chosen by adopting approaches like [15, 39]) on the behalf of other agents. Specifically, TONS adopts the same approach described in [15], in which the leader collects all the agents' local public views, solves both the network graph MST and the network MST problems, and anonymously shares this information into the network.

To implement our approach, we assume that each interaction occurring between agents (i.e., nodes) can be described from a short report, called *proof*. Moreover, in the BC construction we assume also that in NET the agent a can require to the agent b data for this purpose. In such a scenario, a can either ask to a third node c for a "*recommendation*" r about b (based on the proofs about the past interactions of c with b) or it can directly require from b itself a statement about its competence (both are expressed in the real domain $[0, .., 1]$).

Let Z be the "*Assurance*", a real value ranging in $[0, .., 1]$, evaluating the proofs that c can show to a for assuring the reliability level of the r_c provided. To represent the trust measure of r_c sent by c to the requester a about b, the real value $[0, .., 1] - Z$ is computed. The maximum level of assurance (i.e., 1) will be reached when all the proofs produced by c to generate its recommendations about b are traced through authentic and non-repudiable messages recorded on the BC itself. Conversely, the minimum value of Z will be obtained when all transactions will be without proof.

Finally, a "*feedback*" will by assigned from a to b if if a will require data to b.

3.1.1. The Trust Model

Let T_i, Rep_i, β_i, and S_i be four mappings associated with each agent $i \in N$ in the BC network $NET = \langle N, A, W \rangle$. Each mapping has an agent j as input and a different trust measure ranging in $[0, .., 1]$ (with $0/1$ the minimum/maximum trust value), as output assigned by i to j.

In detail:

- $T_i(j)$ is the overall *trust* assigned by i to its interactions with j.
- $Rep_i(j)$ is the measure of the *reputation* assigned by i to j on basis of recommendations coming from the BC agents.
- $tw_i(j)$ is the *trust weight*, i.e. the weight assigned by i to the reliability of the reputation communicated from j to i. In other words, i computes the overall trust score of j by considering both the trust $T_i(j)$ and the reputation $Rep_i(j)$. The percentage of relevance assigned to trust versus reputation is given by $tw_i(j)$, autonomously computed by i based on the level of assurance Z and the recommendations provided by the contacted agents.
- $S_i(j)$ is the overall "score" (preference) assigned to j by i based on its perceived reliability and reputation of j.

Moreover, a further mapping RC_i has been defined to represent the *recommendations* received by the agent i. Specifically, a *recommendation* is assumed to be a pair $r = \langle v, l \rangle$, with v (called *recommendation value*) and l (called *recommendation level of assurance*) ranging in the real domain $[0, .., 1]$ (where $0/1$ is the minimum/maximum value). Formally, $RC_i(j, k)$ is a mapping that in input receives two agents j and k and in output returns the recommendation $RC_i(j, k)$ that j provides to i about k, together with a measure of the level of assurance associated with this recommendation.

3.1.2. The ONS Computation

Each node will update its mappings by means of the steps:

- **Step 1: Reception of the Recommendations**. The agent i receives recommendations (encoded in the RC_i mapping) from the other agents in response to previous recommendation requests. Each recommendation provided by j about the agent k is stored in a *recommendation message* m consisting of a tuple $\langle v, l \rangle$, whose elements are stored by i in its mapping $RC_i(j, k).v$ and $RC_i(j, k).l$, respectively.

- **Step 2: Computing the T mapping**. i updates the mapping T for any agent j with which i has interacted and, as a result, i has issued one or more feedback for contributions provided by j. These feedback, stored in the mapping $FEED_i^k(j)$, represent the quality of the collaboration that j provided to i during the n-th by real values ranging in $[0, 1]$ (where $0/1$ means the minimum/maximum "*Quality of Service*"). Based on these feedback, i updates its mapping T_i and calculates the current reliability shown by j by averaging all the feedback concerning it. By denoting with m the number of recent interactions between i and j, the current trust $T_i(j)$ is computed as:

$$T_i(j) = \frac{1}{m} \sum_{k=1}^{m} FEED_i^k(j) \tag{1}$$

  At each step, T_i is updated by averaging the value of T_i at the previous step $t - 1$ and that computed at the new step t, denoted by T_i^t, as:

$$T_i^t(j) = \alpha \cdot T_i^{(t-1)}(j) + (1 - \alpha) \cdot T_i(j) \tag{2}$$

  where α is a real value ranging in $[0, 1]$ and meaning the relevance assigned by i to the past evaluations of T_i with respect to the current one.

- **Step 3: Computation of Rep and β.** The recommendations in the mapping RC_i are used by the agent i to compute the reputations of the other agents. In detail, the reputation of an agent j is computed by i as a weighted mean of all the recommendations received from the other nodes referred to j (let AS be this set), where the weight of each recommendation value is the corresponding level of assurance. Thus $Rep_i(j)$ is given from:

$$Rep_i(j) = \frac{\sum_{k \in AS, k \neq i} RC_i(k, j).v \cdot RC_i(k, j).l}{\sum_{k \in AS, k \neq i} RC_i(k, j).l} \tag{3}$$

where $RC_i(k,j).v$ (resp., $RC_i(k,j).l$) is the value (resp., the level of assurance) of the recommendation that the agent k returned to i about the agent j, and β is the coefficient associated with i in the mapping β_i. The computation of the average level of assurance of the recommendations related to j, denoted by $\beta_i(j)$, is obtained by averaging the level of assurance associated with all the recommendations related to j. Thus:

$$\beta_i(j) = \frac{\sum_{k \in AS, k \neq i} RC_i(k,j).l}{|AS| - 1} \tag{4}$$

- **Step 4: Computation of S**. The agent i computes the overall score $S_i(j)$ in the agent j by considering both the trust $T_i(j)$ and the reputation $Rep_i(j)$, while the value of the mapping $\beta_i(j)$ weights the relevance of the service reliability with respect to reputation:

$$S_i(j) = \beta_i(j) \cdot T_i(j) + (1 - \beta_i(j)) \cdot Rep_i(j) \tag{5}$$

- **Step 5: Sending the mapping S to the leader**. Each agent i sends its mapping S_i to the leader L of NET.

- **Step 6: Computation of ONS**. The leader L exploits Prim's approach [40] to find the MST by computing both the time-weight $t_{i,j}$ and the trust-weight $\tau_{i,j}$ of the arc $a_{i,j} \in A$, as the arithmetic mean between the $S_i(j)$ and $S_j(i)$. Finally, from the MST then L computes its ONS and sends it to the agents belonging to the ONS that, in turn, will forward it to their neighbors.

At each step, the agent i exploits the mapping S to select the most suitable candidates to require for collaboration. We highlight that the usage of TONS introduces the time cost required from the computation of the trust measures, that is generated realized during the transactions performed in the network.

4. The TONS2 algorithm

The TONS algorithm has been designed to compute the ONS from the MST, but it is an expensive procedure in terms of computational, storage and power resources, which is an aspect relevant in presence of IoT devices and, particularly, when they are light, and poorly equipped.

From the experiments carried out to test TONS, we observed that the set of best neighbor agents was not subject to sudden changes in its composition. Consequently, an updated version of this algorithm, called TONS2, was developed to reduce the computational, storage and power costs required by TONS to the IoT devices, but without significantly impacting on the quality of the neighbor selection process. To this purpose, in TONS2 an heuristic strategy has been introduced to compute a new ONS only:

(i) after each p consecutive blocks validated in the DL, with $p > 1$, (we call "*batch*" this sequence of blocks to validate);

(ii) when the overall score of each agent belonging to the agent set decreases more than a fixed percentage;

(iii) when the communication time occurring between two agents increases more than a fixed percentage;

(iv) when a time threshold, meaning the maximum time allowed to complete a batch, is elapsed.

Therefore, let Δ_S and Δ_{time} be the thresholds defining the percentages of maximum loss of score and communication quality respectively, and let τ be the maximum time imposed to complete a batch.

It is evident as TONS2 could save IoT resources by confirming the agent set without modifying the trust-based ONS search algorithm of TONS. In detail, for each batch the leader will execute the algorithm of TONS with the first block to validate. Then for each subsequent block in the batch to be validated, the leader will preemptively check whether the overall score and the weight-time parameter (both referred to the previous process of block validation) will respect the constraints imposed by Δ_S and Δ_{time}, and the time ξ is not elapsed. If the constraints are met then the leader will initiate a new block validation process, otherwise it will proceed to update all mappings (see below) and will perform the selection of a new set of agents.

Unfortunately, this procedure may cause a side effect on agent trust values due to the fact that the same set of agents will validate multiple consecutive blocks in the DL. In fact, by choosing the same set of agents several times, they will gain an important advantage over the rest of the agent community, having more opportunities to increase their trustworthiness. This implies more chances to be chosen for new block validation processes than the other agents.

Summarizing, the TONS2 strategy will be the following:

- Let x be the loop counter of the batch ranging in $[1, .., p]$. For $x = 1$ the mappings W, T, RC, Rep, $FEED$ and S are copied in W^*, T^*, RC^*, Rep^*, $FEED^*$ and S^*.
- After each other validation block process in the batch only W^*, T^*, RC^*, Rep^*, $FEED^*$ and S^* will be updated as described in Section 3.1.2 and for each pair of agents i and j, belonging to the exploited agent set, then:
  - If $S_i(j)^* >= (1 - \Delta S) \cdot S_i(j)$ && ξ is not elapsed && $x < p$, then a new interaction in the batch will be performed and x will be set to $x + 1$;
  - If $S_i(j)^* < (1 - \Delta S) \cdot S_i(j)$ | | ξ is elapsed | | $x = p$, then the batch will end (i.e., will be completed) and a new ONS computation will be executed by the leader;
- When in the batch there are not new blocks to validate then the mapping T, S and W will be updated as follows: $T = T + (T - T^*)/x$; $S = S + (S - S^*)/x$; $W = W^*$.

In terms of computational complexity, the batch procedure of TONS2 and TONS, for each new iteration in the batch after the first block, have complexities of $O(|N|^2)$ and $O(|N|^3)$, respectively. Therefore, each time that in a batch the ONS computation is not performed because the agent set is confirmed, a significant amount of resources is saved.

The batch procedure above describes is formally presented as a pseudo-algorithm in Algorithm 1.

Algorithm 1 The pseudocode of the Batch procedure of TONS2.

Require: $W, T, RC, Rep, FEED, S, \Delta_S, \Delta_{time}, \xi, p, t_{batch}$
$W^* \leftarrow W, T^+ \leftarrow T, RC^* \leftarrow RC, Rep^* \leftarrow Rep, FEED^* \leftarrow FEED, S^* \leftarrow S$
$x \leftarrow 1$
$control \leftarrow TRUE$
Computing MST
Computing OSN
while $x < p$ && control **do**
    Validation of a Block to add in the DL
    Updating W^*, T^*, RC^*, Rep^*, $FEED^*$, S^*
    for $i = 1 \rightarrow \forall$ agent into the set **do**
        $t_{batch} \leftarrow time(\,)$
        for $j = 1 \rightarrow \forall$ agent into the set $-\{i\}$ **do**
            if $S_i^*(j) < (1 - \Delta S) \cdot S_i(j) \,||\, W^*.\tau_{i,j} < (1 - \Delta_{time}) \cdot W.\tau_{i,j} \,||\, t_{batch} < \xi$ **then**
                $control \leftarrow FALSE$
            end if
        end for
    end for
    $x \leftarrow x + 1$
end while
$T \leftarrow T + (T - T^*)/(x - 1)$
$S \leftarrow S + (S - S^*)/(x - 1)$
$W \leftarrow W^*$
return

4.1. Experiments

This Section describes the experiments we performed to evaluate the advantages given by TONS2 with respect to TONS. The interested reader can found a more detailed analysis of the effectiveness and efficiency of TONS in [16].

To carry out these experiments we adopted the same computational configuration, i.e. an ASUS PC equipped with an Intel i7 CPU (8-Cores, 7GHz), 32 GB DDR4-SDRAM, 1 TB of SSD and Windows-11 OS. Several experiments by using random network models belonging to the Barabási–Albert (BA) and Erdős–Rényi (ER) network models [17, 18] have been realized. Moreover, as in [16] we varied the number of nodes to analyze, the percentage of misbehaving nodes randomly generated, but also adding the dimension of the batch. Δ_S and Δ_{time}, have been set to 5, τ has been set to 600 seconds that is a value suitable for the adopted simulated DL, and the batch size varied from 3 to 5.

The results obtained bv the performed comparison between the TONS and TONS 2 algorithms are presented in the Tables 1 and 2 and graphically depicted in Figures 1 and 2 for the BA and RE network models and different batch dimensions. To permit a proper evaluation of the results, the average times (in seconds) have been reported for only those simulations that completed all the blocks assigned to them. Such results show that the advantage in time introduced by

number of nodes	150	300	500	1000
average neighbors	2	2	5	7
TONS (3 blocks)	61.74	111.39	183.52	302.24
TONS2 (k=3)	24.81	40.98	98.60	186.73
TONS (4 blocks)	84.49	150.92	239.64	409.89
TONS2 (k=4)	34.49	55.82	123.29	267.57
TONS (5 blocks)	102.18	189.96	307.07	501.18
TONS2 (k=5)	40.89	70.43	196.40	304.88

Table 1
Results (in seconds) of TONS and TONS2 on the BA random network model with different sizes and batch dimensions.

TONS2 with respect to TONS increases with both the network size and the batch dimensions. This confirmed our expectations.

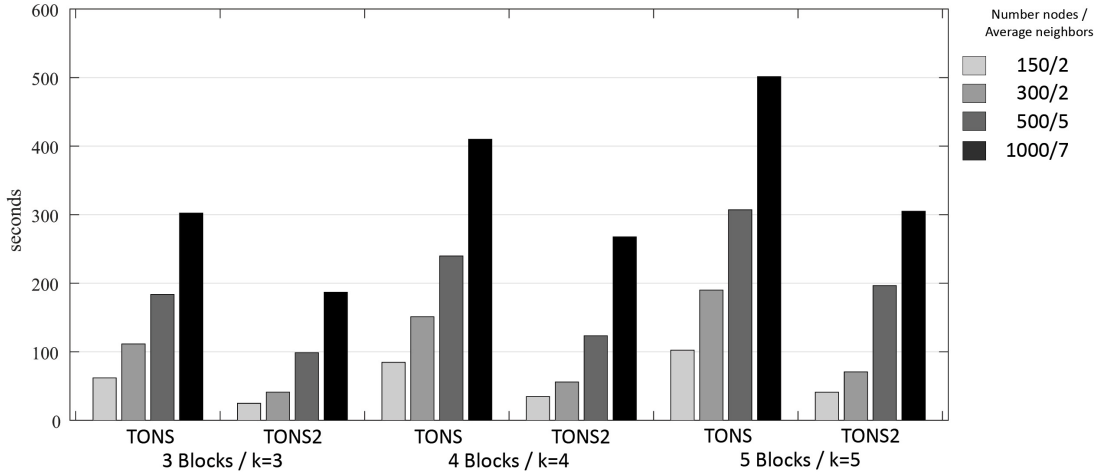

Figure 1: Results (in seconds) of TONS and TONS2 on the BA random network model with different sizes and batch dimensions.

5. Conclusions

The consistency of a DL is a fundamental aspect of BC systems where the propagation of shared data in the shortest way and the optimization for the neighbor selection for each agent play a relevant role, particularly in IoT scenarios.

In our previous work [16] we presented TONS (**T**rust-based **ONS**) an optimized BC network algorithm that allows agents to communicate with an optimized selection of neighboring agents, ensuring that these are the most trustworthy agent nodes in the BC. Therefore, the TONS algorithm identifies the optimal neighbor selection based on both the delivery time rates and

number of nodes	150	300	500	1000
connection probability	**0.02**	**0.015**	**0.01**	**0.007**
TONS (3 blocks)	65.19	105.87	174.51	301.86
TONS2 (k=3)	28.08	41.67	105.46	177.11
TONS (4 blocks)	86.92	141.16	232.68	402.48
TONS2 (k=4)	34.56	48.81	128.37	186.47
TONS (5 blocks)	108.65	176.45	290.85	503.10
TONS2 (k=5)	42.60	56.41	186.47	320.18

Table 2
Results (in seconds) of TONS and TONS2 on the ER random network model with different sizes and batch dimensions.

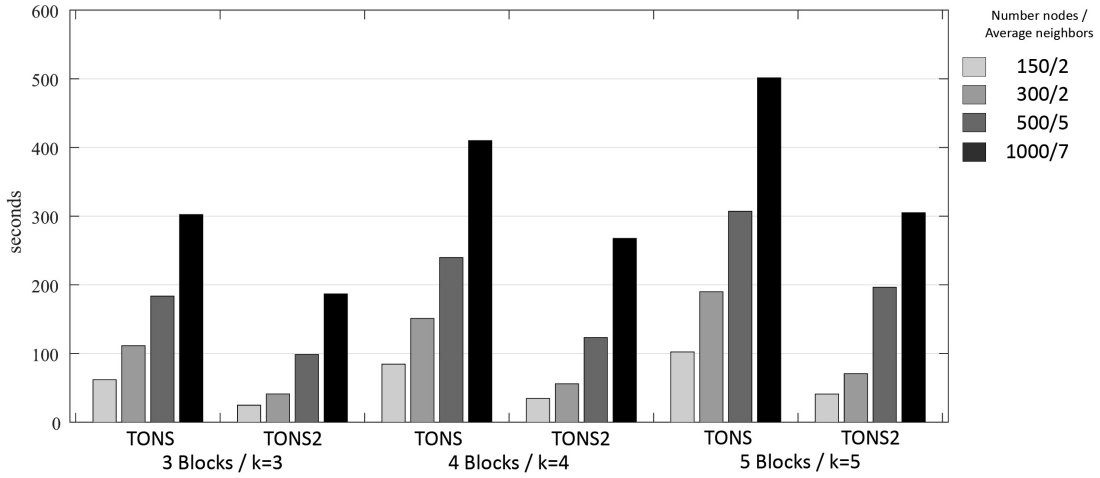

Figure 2: Results (in seconds) of TONS and TONS2 on the ER random network model with different sizes and batch dimensions.

the trustworthiness of the agent nodes in the network.

To improve the computational complexity of TONS, we added in TONS an heuristic strategy to avoid of computing a new set of agents for each new block to validate. To test the benefits provided from this new version of TONS, called TONS2, we compared these two algorithms by executing some simulations on BA and ER network models for different dimensions of the batch of validating blocks. The obtained results have shown that TONS2 is capable to save important amount of computational, storage and power resources (a desired feature in an IoT context) with a minimal performance detriment, particularly important in presence of IoT devices poorly equipped.

References

[1] P. Asghari, A. M. Rahmani, H. H. S. Javadi, Internet of things applications: A systematic review, Computer Networks 148 (2019) 241–261.

[2] S. Li, L. D. Xu, S. Zhao, The internet of things: a survey, Information systems frontiers 17 (2015) 243–259.

[3] J. E. Ibarra-Esquer, F. F. González-Navarro, B. L. Flores-Rios, L. Burtseva, M. A. Astorga-Vargas, Tracking the evolution of the internet of things concept across different application domains, Sensors 17 (2017) 1379.

[4] Y.-K. Chen, Challenges and opportunities of internet of things, in: 17th Asia and South Pacific design automation conference, IEEE, 2012, pp. 383–388.

[5] E. M. Migabo, K. D. Djouani, A. M. Kurien, The narrowband internet of things (nb-iot) resources management performance state of art, challenges, and opportunities, IEEE Access 8 (2020) 97658–97675.

[6] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, H. Wang, Blockchain challenges and opportunities: A survey, International journal of web and grid services 14 (2018) 352–375.

[7] M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, H. Janicke, Blockchain technologies for the internet of things: Research issues and challenges, IEEE Internet of Things Journal 6 (2018) 2188–2204.

[8] X. Wang, X. Zha, W. Ni, R. P. Liu, Y. J. Guo, X. Niu, K. Zheng, Survey on blockchain for internet of things, Computer Communications 136 (2019) 10–29.

[9] L. Kiffer, R. Rajaraman, A. Shelat, A better method to analyze blockchain consistency, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018, pp. 729–744.

[10] S. Jiang, J. Wu, Approaching an optimal bitcoin mining overlay, IEEE/ACM Transactions on Networking (2023).

[11] T. Wang, C. Zhao, Q. Yang, S. Zhang, S. C. Liew, Ethna: Analyzing the underlying peer-to-peer network of ethereum blockchain, IEEE Transactions on Network Science and Engineering 8 (2021) 2131–2146.

[12] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, et al., Hyperledger fabric: a distributed operating system for permissioned blockchains, in: Proceedings of the thirteenth EuroSys conference, 2018, pp. 1–15.

[13] R. G. Brown, J. Carlyle, I. Grigg, M. Hearn, Corda: an introduction, R3 CEV, August 1 (2016) 14.

[14] A. Kertesz, H. Baniata, Consistency analysis of distributed ledgers in fog-enhanced blockchains, in: International European Conference on Parallel and Distributed Computing (Euro-Par 2021), volume 27, 2021.

[15] H. Baniata, A. Anaqreh, A. Kertesz, Dons: Dynamic optimized neighbor selection for smart blockchain networks, Future Generation Computer Systems 130 (2022) 75–90.

[16] G. Fortino, F. Messina, D. Rosaci, G. M. L. Sarnè, Using trust measures to optimize neighbor selection for smart blockchain networks in the iot, IEEE Internet of Things Journal 10 (2023) 21168–21175.

[17] R. Albert, A.-L. Barabási, Statistical mechanics of complex networks, Reviews of modern

physics 74 (2002) 47.
[18] S. Chatterjee, P. Diaconis, Estimating and understanding exponential random graph models, The Annals of Statistics 41 (2013) 2428–2461.
[19] J. F. Mikalsen, Firechain: An efficient blockchain protocol using secure gossip, Master's thesis, UiT Norges arktiske universitet, 2018.
[20] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system, Decentralized Business Review (2008) 21260.
[21] Q. Zhou, H. Huang, Z. Zheng, J. Bian, Solutions to scalability of blockchain: A survey, Ieee Access 8 (2020) 16440–16455.
[22] X. Li, P. Jiang, T. Chen, X. Luo, Q. Wen, A survey on the security of blockchain systems, Future Generation Computer Systems 107 (2020) 841–853.
[23] A. Reyna, C. Martín, J. Chen, E. Soler, M. Díaz, On blockchain and its integration with iot. challenges and opportunities, Future generation computer systems 88 (2018) 173–190.
[24] X. Zhou, W. Liang, I. Kevin, K. Wang, L. T. Yang, Deep correlation mining based on hierarchical hybrid networks for heterogeneous big data recommendations, IEEE Transactions on Computational Social Systems 8 (2020) 171–178.
[25] Z. Yan, P. Zhang, A. V. Vasilakos, A survey on trust management for internet of things, Journal of network and computer applications 42 (2014) 120–134.
[26] G. Fortino, L. Fotia, F. Messina, D. Rosaci, G. M. L. Sarné, Trust and reputation in the internet of things: State-of-the-art and research challenges, IEEE Access 8 (2020) 60117–60125.
[27] R. Kumar, R. Sharma, Leveraging blockchain for ensuring trust in iot: A survey, Journal of King Saud University-Computer and Information Sciences (2021).
[28] Z. Huang, X. Su, Y. Zhang, C. Shi, H. Zhang, L. Xie, A decentralized solution for iot data trusted exchange based-on blockchain, in: 2017 3rd IEEE International Conference on Computer and Communications (ICCC), IEEE, 2017, pp. 1180–1184.
[29] G. Fortino, F. Messina, D. Rosaci, G. M. L. Sarné, Using blockchain in a reputation-based model for grouping agents in the internet of things, IEEE Transactions on Engineering Management 67 (2019) 1231–1243.
[30] B. Shala, U. Trick, A. Lehmann, B. Ghita, S. Shiaeles, Blockchain and trust for secure, end-user-based and decentralized iot service provision, IEEE Access 8 (2020) 119961–119979.
[31] R. Di Pietro, X. Salleras, M. Signorini, E. Waisbard, A blockchain-based trust system for the internet of things, in: Proceedings of the 23nd ACM on symposium on access control models and technologies, 2018, pp. 77–83.
[32] C. F. Bazlamaçcı, K. S. Hindi, Minimum-weight spanning tree algorithms a survey and empirical study, Computers & Operations Research 28 (2001) 767–785.
[33] S. Ruzika, H. W. Hamacher, A survey on multiple objective minimum spanning tree problems, in: Algorithmics of Large and Complex Networks, Springer, 2009, pp. 104–116.
[34] G. Pandurangan, P. Robinson, M. Scquizzato, et al., The distributed minimum spanning tree problem, Bulletin of EATCS 2 (2018).
[35] P. C. Pop, The generalized minimum spanning tree problem: An overview of formulations, solution procedures and latest advances, European Journal of Operational Research 283 (2020) 1–15.
[36] F. Y.-S. Lin, C.-H. Hsiao, Y.-F. Wen, Y.-C. Su, Adaptive broadcast routing assignment algorithm for blockchain synchronization services, in: 2018 Tenth International Conference

on Ubiquitous and Future Networks (ICUFN), IEEE, 2018, pp. 487–492.
[37] W. Bi, H. Yang, M. Zheng, An accelerated method for message propagation in blockchain networks, arXiv preprint arXiv:1809.00455 (2018).
[38] G. Li, X. Ren, J. Wu, W. Ji, H. Yu, J. Cao, R. Wang, Blockchain-based mobile edge computing system, Information Sciences 561 (2021) 70–80.
[39] R. Antwi, J. D. Gadze, E. T. Tchao, A. Sikora, H. Nunoo-Mensah, A. S. Agbemenu, K. O.-B. Obour Agyekum, J. O. Agyemang, D. Welte, E. Keelson, A survey on network optimization techniques for blockchain systems, Algorithms 15 (2022) 193.
[40] R. C. Prim, Shortest connection networks and some generalizations, The Bell System Technical Journal 36 (1957) 1389–1401. doi:10.1002/j.1538-7305.1957.tb01515.x.