

An agent-based method for trust graph calculation in resource constrained environments

Borja Bordel^a, Ramón Alcarria^{b,*}, Diego Martín^a and Diego Sánchez-de-Rivera^a

^a *Department of Telematics engineering. Universidad Politécnica de Madrid. bbordel@dit.upm.es, diego.martin.de.andres@upm.es, diegosanchez@dit.upm.es*

^b *Department of Topographic Engineering and Cartography. Universidad Politécnica de Madrid. ramon.alcarria@upm.es*

Abstract. In future resource constrained systems such as Industry 4.0 and Cyber-Physical Systems, thousands of resource constrained embedded devices communicate in order to reach a common objective. Information, then, is processed and transformed several times since it is acquired at physical level until it is received by user applications. Moreover, usually, each component in an current engineered system has almost no information about the other deployed modules. In that way, any malicious component may severely affect systems through the modification of operation data, as high-level applications can check the validity of received information in a hard way. Therefore, future technological components should only communicate with other trustworthy modules. Typically, solutions to address this problem are based on heavy technologies (such as certificates) which do not fulfil the requirements of embedded devices making up future systems. Thus, in this paper, a light computational solution to calculate the graph representing the trustworthy routes through which an application may obtain information is provided. Systems are represented as directed graphs, and untrustworthy branches are identified based on probability offered by a biparametric stochastic process (which is estimated by means an agent-based solution, consisting of a static and a mobile agent). Untrustworthy paths are also pruned following a hierarchical and scalable process. The proposed solution is based on the execution of some “elemental requests” whose response is known. Using the NS3 simulator, virtualization technologies and TAP bridges, the proposed algorithm is validated in a simulation scenario representing a real deployment.

Keywords: Trust, Security, Resource constrained environments, Graphs, Pruning algorithm

© 2020. This manuscript version is made available under the CC-BY-NC-ND 4.0 license
<https://creativecommons.org/licenses/by-nc-nd/4.0/>
 The Version of Record is available online at: <http://dx.doi.org/10.3233/ICA-190604>

*Corresponding author. E-mail: ramon.alcarria@upm.es

1. Introduction

Recently, several different paradigms to support future digital and computational systems have been proposed. Among them, one of most popular proposals nowadays is Cyber-Physical Systems (CPS). CPS are integrations of computation and physical processes [1]. Contrary to other previous works about embedded devices, CPS are focused on communications. Thus, they are primary designed to be networked (usually making up big networks), working all the connected members for a common objective [1]. Around CPS many other definitions and technological paradigms have been reported, as (for example) “Industry 4.0”. Industry 4.0 [2], as defined during the 2014 Hannover Fair, is understood as the new industrial period when CPS, the Internet of Things (IoT) [3] and Next-generation Internet Services [4] are massively employed as supporting technologies for production systems and solutions. Although, as said, different technologies may be employed to deploy Industry 4.0 systems, CPS paradigm is the most employed one [5]. In particular several architectures and reports about the future challenges in this field have appeared [6].

CONFIGURATION LEVEL <i>Supervisory control, feedback to physical world</i>
COGNITION LEVEL <i>Decision support systems</i>
CYBER LEVEL <i>Peer-to-peer monitoring, fleet of machines</i>
DATA-TO INFORMATION CONVERSION LEVEL <i>Machines, components</i>
SMART CONNECTION LEVEL <i>Sensors, data adquisition from machines</i>

Fig. 1. Example of architecture for an Industry 4.0 system [66]

Any case, all these proposals present a very important and common characteristic. Both, CPS and Industry 4.0 systems are characterized by the complexity of the underlying networks, involving thousands of resource constrained devices and other components or modules (such as software applications, middleware, etc.). In a typical Industry 4.0 system (see Figure 1) around five or six different abstraction layers are defined (following the proposed reference architecture for CPS by the National Institute of Standards and Technology [7]). These layers are isolated, and components belonging to each one of them

have almost no information about the organization, modules and behavior of the other layers. Thus, important meta-information about the operation data and provided results is not accessible, especially, from the highest level, where user applications are hosted. Using this approach, different middleware may be deployed (such as those required to execute mobile agents) in an independent manner from heterogeneous hardware and embedded software components that usually made up platforms at low level.

In this context, systems are vulnerable to a new and more general type of attacks known as cyber-physical attacks [8].

These attacks are based on the strong dependencies among components in future technological systems, as well as on the great number of them composing the system, to perform malicious effects in a simple and “noiseless” way [9]. For example, as Industry 4.0 systems are usually employed to perform control policies [10], a secondary and/or small intermediate component in the system may be attacked to provide false information, so user applications operate and decide control actions based on false data. In that way, control loops may magnify the initial effect, and the entire system may fail starting from the infection of a non-relevant or critical module.

Therefore, as most recent technological systems (such as pervasive computing environments) are not centralized, each component (and especially user applications with full system access) should evaluate which components are trustworthy and can be considered to provide trustworthy information. Components only providing false data must be pruned and isolated [11], and modules with a variable behavior have to be requested to evaluate the received information before submitting it to next component.

However, traditionally, trust in technological systems is managed through computationally heavy technologies (in terms of memory and processing time, especially for resource constrained devices) such as certificates, which do not fulfill the special requirements of resource constraint embedded devices making up future engineered systems such as Industry 4.0 solutions (they present a reduced computational power, a limited communication range, etc.).

In that way, this work is motivated by the need of new, fast and lightweight security solutions to evaluate the components’ trustworthiness in Industry 4.0 systems. These new solutions must be adequate for resource constrained devices, which cannot store in a permanent manner (due to memory limitations) trust evaluation algorithms and their related data. Besides, these devices cannot maintain a complete under-

standing or model about the system where they are integrated, so new algorithms must be based on simplified models which allow a total security analysis although they only partially represent the system. Finally, complex statistical methods are not admissible as a large processing time would be required; on the contrary, trust evaluations should consist of simple key performance indicators. Thus, the objective of this paper is to propose a new lightweight computational solution to fulfill this gap.

As a novelty, in our solution Industry 4.0 systems are represented as graphs, where only communication links and cyber-physical components are considered (other elements, such as configuration mechanism are not represented, for example). In that way, complex and large systems may be modeled in a lightweight manner. Then, using simple statistics about exchanged traffic among the different components, trust is evaluated in a fast manner. Besides, trust evaluation algorithms are not permanently stored in embedded devices, but executed through mobile agents, which reduces the resource consumption of this solution. In particular, the proposed solution represents resource constraint environments as directed graphs, which are analyzed in a hierarchical and scalable way using agent-based techniques. In order to do that an epidemic protocol based on a sequence of “elemental requests” is proposed. Depending on the obtained results, branches in the system may be pruned or filters can be installed. When the configuration process ends, each component has calculated a “trust circle” including the components which provide it with correct data. In order to simplify the mathematical notation and the discussion, proposal is particularized to Industry 4.0 systems, although it is applicable in an identical way to any other resource constraint environment or system (such as CPS, IoT, pervasive sensing platforms, etc.).

As main advantages, the proposed solution present a shorter convergence time, as simple traffic statistics are calculated much faster than previous complex statistical methods. Besides, the resource consumption in resource constrained embedded devices is also low, as models based on graphs and pruning techniques are lightweight technologies. Moreover, mobile agents guarantee computational resources are only employed for a short time in an opportunistic manner.

The rest of the paper is organized as follows. Section 2 presents the state of the art on Industry 4.0 solutions, trust management, graph pruning techniques and agent-based solutions for CPS. Although Industry 4.0 are integrations of hardware and embed-

ded software for industrial applications, in this paper we are focusing on system modeling and transformation, using the graph theory and its applications and related technologies (trust calculation, pruning techniques, etc.).

Section 3 describes the main computational contribution, including the mathematical formalization and the epidemic protocol. In this case, as said, the proposal is mainly based on system modeling through graph theory as initial step to perform calculations and transformations over the system using agent-based techniques.

Section 4 describes the performed experimental validation (based on simulations), and Section 5 presents the obtained results. Section 6 concludes the paper.

2. State of the art

This section presents an overview of the related research in the two main fields treated in this work: trust management and graph pruning algorithms.

2.1. Industry 4.0 solutions

Technically, and Industry 4.0 system is an industrial system based on CPS, but considering some special characteristics [12].

First, system heterogeneity, complexity and dynamism are pretty lower than in standard CPS [12]. Although some of the main problems to be addressed in CPS is self-configuration [13] and modeling [14], Industry 4.0 solutions usually consider static CPS, where architecture and components are stable with time. In fact, the use of CPS in Industry 4.0 refers a system where information related to the physical shop floor and the virtual computational space are highly synchronized [15], without considering other relevant characteristics (very relevant in other application scenarios). Industrial systems tend to be more homogenous and statistic than other technological solutions, a requirement that is usually considered in firstly reported Industry 4.0 real deployments [16].

Second, CPS are understood as a platform similar to other paradigms such as Internet of Things (IoT) or Smart Factories [17]. Actually, Industry 4.0 systems are characterized to integrate next-generation digital solutions such as smart networking, new digital business models or flexibly production schemes [12]. In this context, any system addressing these problems is considered an Industry 4.0 solution, alt-

though it does not fulfill traditional characteristic of CPS.

And, finally, Industry 4.0 are resource constrained environments [18]. Standard CPS may be composed of powerful components, but Industry 4.0 only considers pervasive sensing resource constrained platforms, where sensor nodes are employed to obtain real time information about critical infrastructures and processes (enabling new and advanced control techniques and solutions).

These characteristics increase the risk of Industry 4.0 of suffering a cyber-attacks in comparison to standard CPS. First because they are applied to critical systems, second because components are resource constrained and cannot implement complex protection policies [19], third because architecture is static, and attacker could take profit of bottlenecks and other similar problems; and finally because they implement advanced technological solutions which may present unknown security problems.

Nowadays, only initial proposals about Industry 4.0 have been reported; mainly architectures and reports about pending challenges [18]. However, some real deployments may be also found, including process execution systems [20] and security solutions [21].

The proposed agent-based solution is primary designed for CPS supporting Industry 4.0 systems. Thus, some additional assumptions may be done (see above) in order to develop our proposal and design the validation methodology (static architecture, low heterogeneity level, etc.). However, the proposed technology could be also applied to other resource constrained systems, including standard CPS (although performance may be different than showed in this work).

2.2. Trust management

Trust management refers all technologies focused on selecting the conditions on which a component is able to interact with other components in the system [22]. Questions about trust decisions such as when an interaction may take place, or with who a component should interact are traditionally addressed in works about trust and multi-agent systems [23]. However, most Industry 4.0 systems are not multi-agent solutions, but intelligent sensing platforms. Thus, although we are using agents as an enabling technology, we cannot consider Industry 4.0 systems as multi-agent scenarios, as it is not guaranteed all elements to be able to make decisions in a flexible and autonomous manner.

Trust management is an interdisciplinary idea, which has been investigated in different areas: from the Internet-of-Things and Peer-to-Peer systems (P2P), to social and autonomic networks.

There is little work on trust management for IoT [24]. Moreover, most of these works are related to the “social IoT”, a concept which refers the idea that common objects may interact with other daily living things and create a social network as people do [25]. In this context, contributions about protection, security and intrusion detection using trust models are divided into two groups.

In the first group, articles deal with trust evaluation. In these papers, researchers try to determine the cyber and physical parameters which should influence trust among IoT components. Typically, direct and indirect observations are employed to extract the required information through some data analytics procedures [26]. Observations may be related to networks parameters (such as the packet loss rate) [27] or may be based on the social network theory [28]. However, taxonomies about uncertainties and trust [20], enhanced processing algorithms [30] and uncertainty models [31] have been also reported. In all these proposals, trust is continuously evaluated and, employing a specific threshold, it is determined whether components communicating with a certain module in the system are trustworthy or not [32].

Sometimes, trust is evaluated through reputation models and frameworks, defining this concept as fuzzy information distributed around all the system [32]. Concepts such as local and global reputation are then proposed [11].

These previous proposals are useful but require a high computation capacity in Industry 4.0 nodes to be executed. Complex statistical frameworks or large databases have to be maintained. Moreover, these works usually try to “infer” the trust a component deserves, so protection actions are not necessary supported by clear evidences.

In the second group of works, papers are focused on recommendation algorithms. In this case, trust is calculated from opinions publicly shared by components in the system [33]. Phenomena, such as the trust reduction when receiving opinions from distant components, are evaluated [34]. In these papers, trust (and distrust) propagation in simple graphs is formalized, defining ideas like the iterative propagation of trust [35].

Social and automatic networks present an intrinsic social behavior [36]. In these scenarios, the concept of trust graphs (graphs representing the trust relationships in a system) is proposed, and (consider-

ing the particular proposed definition for trust) some algebraic properties of trust are obtained and validated [36]. As in social networks, usually, nodes publicly share their opinion; trust is typically based on voting in these papers. As in the first group, these contributions try to detect untrustworthy nodes in the network (although it is not specified how the network is protected or reacts once the malicious node is located).

Different contributions in the context of semantic technologies have appeared in relation with trust. Specific data formats for opinion and trust calculation algorithms (similar to weigh calculation techniques in neural networks [37]) have appeared [38]. Moreover, different formats to represent reputation have been also reported in other fields [39].

Finally, in the field of P2P systems (and ad hoc networks), different types of trust are defined [40], which are employed to propose new ideas such as “trust witnesses”. In these proposals, opinion is usually composed of various parameters (such as trust and confidence) which made up an “opinion space” [41] where algebraic structures [42] (e.g. semirings) are defined. In this field, besides, the idea of “trust graph” as protection or security policy is connected with other techniques as certificate distribution [43] (i.e. trust implies delegation) or reputation-based sharing platforms [44].

All the cited proposals, nevertheless, only aim to evaluate the trustworthiness of nodes (components in the Industry 4.0 system), keeping the same system architecture regardless of the obtained results. In that way, trust graphs are understood as graphs which include information about trust (but which preserves the original topology of the system and presents the same behavior). On the contrary, in our proposal, trust information is employed to dynamically modify the system topology, so during operation no more calculations are required (and resources may be used in a more efficient way). In order to reduce the global system graph to the trust graph, malicious components are pruned using the most appropriate technique. Pruned elements may be included another time in the system if their untrustworthy behavior changes.

2.3. Pruning techniques

Pruning techniques refers those algorithms and solutions focused on detecting and removing branches or nodes (components) in graphs, neural networks, workflows, etc. fulfilling certain conditions [45].

Different graph pruning techniques have been reported in the research literature.

A first type of proposals consists of removing a node without losing connectivity in the graph, by creating direct connections between the points previously connected through the removed node [46]. This approach is usually named as “creation of a dense graph”.

Other proposals consider different characteristics in the nodes (such as if they are dominant) to prune graphs following certain rules (for example, both created sub-graphs must include a dominant node) [47].

A particular case of pruning techniques is which are applied to decision graphs. In this case, when a node is removed, all its successors are also deleted. This technique is usually employed to simplify the decision-making process [48], as the resulting graph may be severely reduced compared to the original [49].

Other type of pruning techniques is focused on removing loops in graphs [50]. This technique, contrary to the previously cited ones, does not remove any node but connections among them.

With the same idea, other proposals have been reported, for example, calculating pruning actions by means of a forward-backward algorithm [51].

Pruning techniques based on graph simplification [40] are unhelpful in relation to Industry 4.0 systems, as with this approach large parts of the system may be get isolated and useless. Thus, with the proposed technique, vertices are not removed and only edges associated with the component under study may be affected. Solutions focused on removing loops are inapplicable, as it is not the focus of the article (in fact, in this work graphs with cycles are considered and admitted). Finally, in Industry 4.0 systems, components cannot be removed by a peer (another module), as this task is reserved to the system administrator. Then, the proposed technology should be focused on pruning graphs by removing edges instead of nodes. The proposed algorithm employed to select the edges to be pruned is explained in the next section.

2.4. Mobile agents and CPS

Agent-based solutions (especially when employing mobile agents) are a very well-known technology. For more than thirty years, different mechanisms and techniques have been reported [52]. This interest for mobile agents is still alive nowadays, and agent-

based solutions for new paradigms such as Cyber-Physical Systems and Internet of Things (IoT) are constantly reported [32][53].

Agent-based solutions for CPS cover a large catalogue of applications: from smart object implementation [54] to human-computer interaction models [55]. However, most common agent-based proposals for CPS are related to security issues [32][56]. Actually, some important authors have evaluated how mobile agents break some traditional assumptions in computer sciences, opening new security issues which prevents the use of this technology in CPS and IoT [57].

On the other hand, CPS are very heterogeneous systems, a characteristic that may also prevent the integration of agents into CPS [58]. However, techniques to overcome the CPS heterogeneity and apply agents and other similar software technologies to CPS have been reported in the last years [59][60]. In order to guarantee the feasibility of the proposed solution, the described technology in this paper should be combined with any of the existing mechanism to implement mobile agents on heterogeneous CPS.

Focused on the topic of this paper, different security solutions for CPS employing mobile agents may be found. Recent proposals addressing this topic include cyber-protection assurance through secure agent implementations (including, for example, a Public Key Infrastructure) [61]. However, most of these solutions employ agents to collect information about the system and make a decision about its operation in a previously defined bifurcation point [62]. Other solutions consider security point injection in workflows, which are managed by mobile agents, or intrusion detection mechanisms [56]. These proposals support the usability of mobile agents in the context of CPS for security applications; but proposed solutions are, nowadays, still very heavy and computationally costly. Moreover, they are focused on very particular scenarios, so more general and lightweight solutions are needed.

In the proposed solution, to create a lightweight mechanism, mobile agents do not collect raw data to be later processed but obtain simple traffic statistics which may be represented using small data structures. Thus, mobile agents are smaller and consume fewer communication and computational resources. Besides, generic parameters are studied, so the proposed technique is adequate for any CPS application scenario.

3. Trust graph calculation

In this Section a mathematical formalization of the proposed contribution is provided. Resource constraint environments (particularized into Industry 4.0 systems) are represented as directed graphs, which are pruned to obtain the trust graph of each component. Data structures and protocols employed to support the proposed algorithm are also described.

Previous works [11] described a simplified calculation process where only systems with one-to-one correspondences between hardware devices and user applications were allowed. In this paper we extend these previous proposals considering arbitrary systems.

Among all questions to be answered as part of the “trust” problem in technological systems, in this work we are dealing with the challenge: which components in the system should a certain element interact with?

3.1. Fundamentals of the mathematical modeling

An Industry 4.0 system $\mathfrak{X}$ is an ordered pair of finite sets $\mathfrak{X} = (C_{\mathfrak{X}}, L_{\mathfrak{X}})$, being $C_{\mathfrak{X}}$ and $L_{\mathfrak{X}}$ non-empty sets. $C_{\mathfrak{X}}$ is named as the collection of system components, whose cardinality is N

$$C_{\mathfrak{X}} = \{c_i \mid i = 1, \dots, N\}$$

and $L_{\mathfrak{X}}$ is the set of communication links, being $\text{card}(L_{\mathfrak{X}}) = K$

$$L_{\mathfrak{X}} = \{l_i \mid i = 1, \dots, K\}$$

Both sets $C_{\mathfrak{X}}$ and $L_{\mathfrak{X}}$ are related by means of three different applications

$$\gamma_{\mathfrak{X}} : L_{\mathfrak{X}} \rightarrow C_{\mathfrak{X}} \times C_{\mathfrak{X}}$$

$$\sigma_{\mathfrak{X}} : L_{\mathfrak{X}} \rightarrow C_{\mathfrak{X}}$$

$$\tau_{\mathfrak{X}} : L_{\mathfrak{X}} \rightarrow C_{\mathfrak{X}}$$

$\gamma_{\mathfrak{X}}$ is the incidence application of the system, and relates each communication link l_i with the order pair of components $\{c_i, c_j\}$ this link connects. The link is outgoing from c_i and incoming in c_j

$\sigma_{\mathfrak{X}}$ is the source application of the system, and associates each link l_i with the component c_i which is the origin of the communication link l_i .

Finally, $\tau_{\mathfrak{X}}$ is the target application of the system, and associates each link l_i with the component c_j which is the destination of the communication link l_i .

In that way, an Industry 4.0 system may be understood as a directed or oriented graph $G_{\mathfrak{X}}$ characterized by any of the two following tuples

$$G_{\mathfrak{X}} = (V_G, E_G, \gamma_G)$$

$$G_{\mathfrak{T}} = (V_G, E_G, \sigma_G, \tau_G)$$

V_G is the set of vertices of the graph, and E_G is the set of edges. γ_G is the incidence application of the oriented graph, and σ_G and τ_G are the source and target functions, as defined in the mathematical graph theory.

The graph $G_{\mathfrak{T}}$ is said to represent the Industry 4.0 system $\mathfrak{T}$ if there exists a pair of bijections ($\mathfrak{f}_V, \mathfrak{f}_E$) $\mathfrak{f}_V : C_{\mathfrak{T}} \rightarrow V_G$, $\mathfrak{f}_E : L_{\mathfrak{T}} \rightarrow E_G$ fulfilling that $\forall c_i, c_j \in C_{\mathfrak{T}}$ and $\forall l_i \in L_{\mathfrak{T}}$ such $\gamma_{\mathfrak{T}}(l_i) = \{c_i, c_j\}$ and $\sigma_{\mathfrak{T}}(l_i) = c_i$ and $\tau_{\mathfrak{T}}(l_i) = c_j$. It is proved that $\exists v_i, v_j \in V_G$ and $e_i \in E_G$ such $\gamma_G(e_i) = \{v_i, v_j\}$ and $\sigma_G(e_i) = v_i$ and $\tau_G(e_i) = v_j$ if, and only if, $v_i = \mathfrak{f}_V(c_i)$ and $v_j = \mathfrak{f}_V(c_j)$ and $e_i = \mathfrak{f}_E(l_i)$.

In order to clarify this point, Figure 2 shows a graphic representation of both functions. As can be seen, these functions allow transforming a technological system into a graph and vice versa, so (hereinafter) we are using indistinctly both notions.

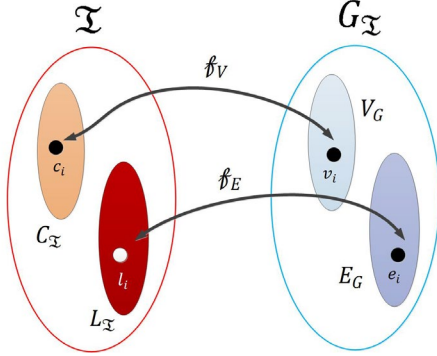


Fig. 2. Correspondence between mathematical graph and Industry 4.0 system

Moreover, vertices and edges in the $G_{\mathfrak{T}}$ graph may inherit some properties from components and communication links in the Industry 4.0 system. In particular, each vertex has associated a collection of three relevant parameters (ID , $tier$, Op):

- ID is a unique identifier (a binary word). It can be managed by a global institution such as the IEEE or by a local administrator. Ensuring the adequate identification of vertices is a key factor in the proposed solution.
- $tier$ is an integer number indicating the abstraction layer to which the component (vertex) belongs. Usually, such as in the OSI reference model, layers closer to the physical layer (hardware devices) present lower

numbers than layers closer to final applications (users).

- Op is a list describing (naming) all the operations the component may perform. It can be composed of alphanumerical names, logical operation identifiers, etc. Hereinafter, $M = card\{Op\}$, being $card\{\cdot\}$ an operator to indicate the calculation of the cardinality's set; and M the number of considered operations in our model.

On the other hand, edges are also characterized by two different parameters (ID , $trust$):

- ID maintains the same meaning as explained for vertices
- $trust$ represents a number indicating the trustworthiness of the messages being transmitted through the link. In general, this value is calculated by the source vertex. If desired, this value may be understood as a weight of the communication link. In that case $G_{\mathfrak{T}}$ turns into a labeled graph or, being more specific, (as $trust$ should take values from an ordered set, typically the real numbers) $G_{\mathfrak{T}}$ turns into a weighted graph.

Figure 3 shows a graph representing a simple Industry 4.0 system.

In a generic case, thousands of components and communication links make up a resource constraint environment -or Industry 4.0 system- (and, thus, the same number of vertices and edges composes the equivalent graph). It is not strange if some of these components or links (due to different reasons such as interferences, cyber-attacks, etc.) behave in an untrustworthy way. The objective of every module in the system, then, should be to avoid as much as possible relating or employing these components or links.

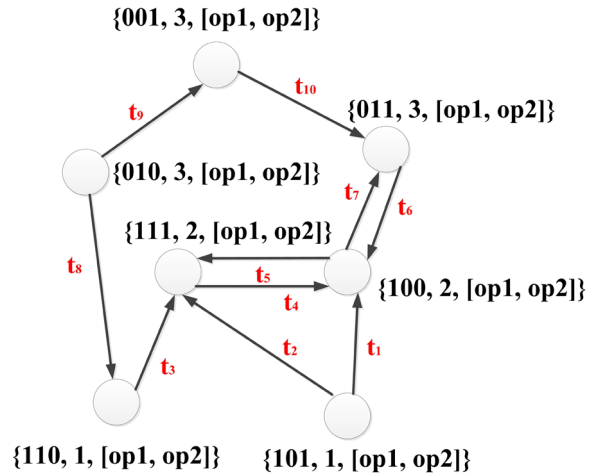


Fig. 3. Example of a graph representing an Industry 4.0 system

In this context it is defined the absolute trust graph T_{c_i} of a component c_i as the subgraph of $G_{\mathcal{T}}$ composed by the vertices and edges with a trustworthy behavior from the point of view of c_i . Figure 4 shows a trust graph in color green.

The final objective for the component c_i is to route its messages and packets and, in general, perform its actions using only elements belonging to this trust graph. The main obstacle in this challenge is the fuzzy management of these systems, where each component only has information about the modules that surround it (i.e. with which it shares a communication link). Therefore, it is especially important to define the trust circle of a component.

The trust circle $\mathcal{C}_{c_i}$ of a component c_i is a set containing all the components adjacent to it with a trustworthy behavior from the point of view of c_i . Two components are considered adjacent if their corresponding vertices in $G_{\mathcal{T}}$ are also adjacent in mathematical sense. Figure 4 shows a trust circle in color purple.

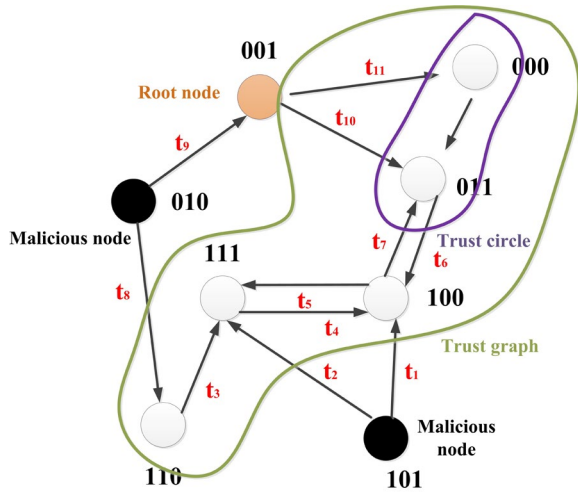


Fig. 4. Example of a graph with the proposed definitions for trust management

With these ideas, each component should be able to calculate its own trust circle (as it has information about the adjacent nodes), and a mechanism by which this calculation also causes the obtaining of the trust graph must be defined (using, probable, a distributed procedure).

In all cases, however, the proposed “absolute definitions” are very restrictive; as components and

communication links generally show different behaviors when different operations are performed. Thus, in order to maximize the use of the available resources in the system, components may employ a relative definition on the proposed concepts.

In that way, the relative trust graph $T_{c_i}^{op_j}$ of a component c_i relative to the operation op_j is defined as the subgraph of $G_{\mathcal{T}}$ composed by the vertices and edges with a trustworthy behavior from the point of view of c_i when performing the operation op_j . The absolute trust graph is related to the relative trust graphs, as it is made of the elements present in all relative trust graphs. In particular:

$$\forall v_i \in V_G, v_i \in T_{c_i} \Leftrightarrow v_i \in V_{T_{c_i}^{op_j}} \quad j = 1, \dots, M$$

$$\forall e_i \in E_G, e_i \in T_{c_i} \Leftrightarrow e_i \in E_{T_{c_i}^{op_j}} \quad j = 1, \dots, M$$

The relative trust circle is defined in a similar way.

From the computational point of view, the calculation algorithms, the obtained absolute trust graph and the relative trust graphs depend all in the same way on the number of elements (components and links) in the system. Thus, they present the same complexity order in respect to the system topology and size (and, then, the same scalability properties, bottlenecks, etc.). Later, these dependencies will be discussed. However, if relative definitions are employed, the calculation process depends also on the number of available operations. In fact, computationally, there is no difference between calculating the absolute trust graph or any of the possible relative trust graphs, only the employed protocols or data format may change. Thus, the manner of obtaining all the possible relative trust graphs is to repeat the calculation for every available operation. In that way, the corresponding algorithm depends linearly on the number of available operations. This situation should be considered when estimating the refreshing period or the required computational time to perform the calculations.

Hereinafter we are describing a trust graph calculation process, which may be applied to obtain the global graph or any relative graph.

3.2. Probability and trust calculation models

Being A the adjacent matrix of $G_{\mathcal{T}}$, the number of paths $p_{i,j}$ from vertex v_i to vertex v_j may be calculated as

$$p_{i,j} = \sum_{r=1}^K [A^r]_{i,j}$$

In its basic form, the adjacent matrix A indicates if two vertexes are connected by an edge or not. If vertexes v_i and v_j are connected by an edge, the position (i, j) in A is set to the unit; zero if they are not connected. Besides, in general, the r -th pow of A indicates if vertexes are connected through r edges. Thus, the position (i, j) in A^r is set to the unit if vertexes v_i and v_j are connected by r edges; zero if they are not connected.

Then, to calculate the number of paths between two vertexes $p_{i,j}$ in a directed graph with K edges, it must be evaluated if these vertexes are connected through all possible numbers of edges r ; incrementing a counter each time vertexes are connected.

Equally, the number of paths that connect vertex v_j to vertex v_i may be obtained as

$$p_{j,i} = \sum_{r=1}^K [A^r]_{j,i}$$

Thus, the number of paths p that allow connecting vertices v_i and v_j in a bidirectional way (i.e. cycles centered in v_i and passing through v_j) is

$$p_j^i = p_{i,j} \cdot p_{j,i}$$

However, Industry 4.0 systems are characterized by presenting a lot of redundancies. Thus, a direct communication between particular low-level components and user applications is rather rare. Usually, connections are established with an entire platform containing several devices with similar functions. These platforms belong to a unique abstraction layer, so all devices (vertices) with similar functions present the same value for *tier* parameter. Considering that $tier_{v_j} = n$ and a total of T vertices present also the value $tier = n$, the number of paths p^i whose trustworthiness have to be evaluated before obtaining the trust graph of vertex v_i is

$$p^i = \sum_{r=1}^T p_r^i$$

Formally, the probability of a path pt to be trustworthy may be calculated as the combination of the different edges (links) and vertices (components) making up the path to be trustworthy. As, in general, the trustworthiness of every edge and vertex is an independent event, the final probability can be obtained through a product (E is the number of edges composing the path)

$$P[pt \text{ trustworthy}] =$$

$$= \prod_{j=1}^E P[e_j \text{ trustworthy}] \prod_{j=1}^{E-1} P[v_j \text{ trustworthy}]$$

This approach, nevertheless, implies vertex v_i to know every edge and component in the path, so it can obtain the global probability. In practical Industry 4.0 systems this assumption is in general not true, so vertex v_i can only evaluate the trustworthiness of edges and vertices adjacent to it.

Let $\mathfrak{N}_{v_i}$ be the set of all vertices adjacent to v_i in $G_{\mathfrak{T}}$. Moreover, let $\mathfrak{N}_{v_i}^s$ be the set of all vertices adjacent to v_i , so the edge joining these vertices starts in v_i . Finally, $\mathfrak{N}_{v_i}^t$ is the set of all vertices adjacent to v_i , so the edge joining these vertices finishes in v_i . It can be easily proved that $\mathfrak{N}_{v_i} = \mathfrak{N}_{v_i}^s \cup \mathfrak{N}_{v_i}^t$.

Thus, in this work, it is said a vertex $v_j \in \mathfrak{N}_{v_i}^s$ is trustworthy along v_i if, and only if, paths starting in v_i and passing through v_j are trustworthy. Equally, it is said a vertex $v_j \in \mathfrak{N}_{v_i}^t$ is trustworthy along v_i if, and only if, paths finishing in v_i and passing through v_j are trustworthy.

In order to evaluate the trustworthiness of a path, whose elements (edges and vertices) are unknown, components are provided with the infrastructure that is shown in Figure 5.

Each node maintains a *Trust repository* (TR) where trust information about the adjacent vertex and links is stored. In particular, the trust circle $\mathfrak{C}_{v_i}$ of the vertex v_i and the probabilities for the vertex' neighbors $\mathfrak{N}_{v_i}$ to be trustworthy are contained in the TR.

Information in the TR is generated and updated by a *Trust Agency* (TA). The TA represents a multi-agent system, including two different types of agents. Among its functions, TA must control the generation, behavior, temporization, design, composition, etc. of agents (but is not an agent). Basically, it orchestrates the agents' operation. This component is unique, contrary to agents that are diverse, and isolates the agent coordination from details about trust graph calculation encapsulated into agents (specifically, into MaA).

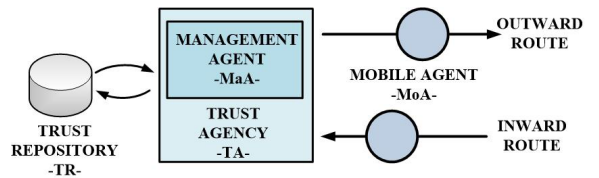


Fig 5. Trust calculation platform

The *management agent* (MaA) is a static agent which controls the trust calculation process. It gets activated each time trust graphs have to be calculated or refreshed. Basically, it constructs the ephemeral data structures necessary to perform the calculation, activates the *Mobile Agents* (MoA) to visit the target vertices and computes the trust graph based on the behavior of the different paths when performing some simple operations named as “elemental requests”. These elemental requests are regular operation requests which have to be answered following a pre-programmed pattern. If received responses do not fit these patterns, the vertices through which messages are sent and/or received are suspected to be untrustworthy (see Section 3.3).

The use of mobile agents presents, in this context, three important advantages.

First, the use of mobile agents enables a fast a lightweight trust evaluation, as resource constrained nodes only must store and execute the proposed algorithm (which is also simple and computationally low cost) in an ephemeral and opportunistic manner. In that way, we are meeting the most important requirement and characteristic of Industry 4.0 systems (see Section 2).

Second, univocal vertices identification is enabled. In message passing solutions, nodes are usually identified by global communication identifiers which are location-dependent and ephemeral (such as IP addresses). Thus, any component moving around the system may appear with different identifiers at different moments which makes difficult to evaluate its trustworthiness in an efficient manner. Contrary to this approach, mobile agents may access to unique identifiers stored in components in a very easy manner. These identifiers do not change with the component location, and trustworthiness is then evaluated in a much more efficient way, even when components get in and get out the system dynamically.

Third, and finally, system scalability is enhanced. As trust evaluation is orchestrated by the TA, traffic engineering policies and other similar solutions to avoid network congestion synchronizing the operation of all components in the systems may be applied. On the contrary, traditional asynchronous message-passing solutions cause a traffic flood generating the system congestion when the number of interconnected components goes up.

Two additional considerations have to be taken into account.

First, in order to respond to the elemental requests (ERQ) with the previewed patterns, programming into components in the Industry 4.0 system should be

modified. In order to avoid this problem, the TA also includes some MoA which visit the components of the system and respond to the ERQ with the pre-programmed patterns. These agents are triggered by the MaA, and also collect information about the transmission and execution process in order to improve the trust graph calculation. Agents are routed by the system following the standard routing rules, and get hosted in the vertices fulfilling a certain requirement (e.g. belonging a certain abstraction layer). If necessary, MoA can be replicated during routing to reach multiple destinations.

And, secondly, it has to be considered that absolute results are almost impossible to obtain. In that way, a trustworthy component will never return 100% of correct responses, random errors also occur and they do not have to affect the trustworthiness of nodes. In that way, two thresholds are defined θ_{min} and θ_{max} , being $0 \leq \theta_i \leq 1$. Thus, an element is considered trustworthy if, and only if, $P[trustworthy] \geq \theta_{max}$. Besides, an element is considered untrustworthy if, and only if, $P[trustworthy] \leq \theta_{min}$. For any other value between thresholds, different traffic engineering techniques (where trust probabilities may be considered as weights) or traffic filters may be considered.

Although traditional expert-created weights could also be employed, some important differences with probabilities must be considered. First, probabilities represent a real, dynamic and measurement-supported estimation; while expert-created weights tend to be fixed and based on previous experiences whose applicability may be limited. Thus, efficiency of traffic engineering techniques is improved when probabilities are employed as weight. And, second, probabilities have a direct physical meaning, while weights may present different consequences. That makes easier the configuration of probability-based solutions than expert-created weights mechanism. Furthermore, as probabilities may be calculated in a systematic and automatic manner, human intervention may be totally removed (which is not possible when using expert-created weights).

On the other hand, a final risk must be discussed. While traveling through the graph, agents could be infected, duplicated or (in general) turned into malicious objects. In that way, cybercriminals could introduce confusion in the system. Then, a solution to protect agents would be required. However, different characteristics make this risk almost negligible, and no additional mechanism to protect agents is actually needed. Namely:

Contrary to cyber-physical components, that are persistent elements in the system architecture, proposed agents are ephemeral objects with a very short lifetime. Thus, infected or duplicated agents cannot operate indefinitely, and their temporal impact is very limited.

Besides, agents may be provided with a (light-weight) digital signature to guarantee the integrity and authenticity of ERQ and patterns. Moreover, as agents are ephemeral objects, this signature does not have to be very complex, and proposed solution may be still supported by resource constrained devices.

3.3. Trust calculation process

Considering all the previous descriptions, the trust graph calculation process is as follows.

First, the MoA of v_i vertex gets activated and triggers the MoA to reach the vertices characterized by a value $tier = n$ (dst field). The selected value of $tier$ parameter determines the “depth” of the calculated trust graph (which may include only one abstraction layer or -even- all of them).

$$MoA = \{dst: \{tier: n\}, sq_{number}, code\}$$

The mobile agent is transmitted by means of an epidemic protocol through the system: each component receiving the agent transmits it to all its neighbors. Moreover, each mobile agent is marked with a unique *sequence number* (and, of course, the code to be executed). In order to avoid duplicated transmissions, a vertex only transmits agents with new se-

quence numbers (in other cases transmission is discarded).

Once the MoA arrives to a vertex fulfilling the requirements described in the dst field, it gets installed.

At the same time, MaA triggers a timer. When time is over it generates a collection of elemental requests

$$ERQ = \{vertex_{src}, rq_{msg}, sq_{number}, timestamp\}$$

Each elemental request describes the source vertex (the vertex through which the root vertex v_i sends the request), which must fulfill that $vertex_{src} \in \mathfrak{N}_{v_i}^s$; the body (or message) of the request; and the sequence number, indicating the MoA which should answer the request (various calculation processes may be performed in parallel). In this case, as different paths may pass through a same vertex and various sequences of ERQ may be triggered, every elemental request (independently of the sequence number) is routed and transmitted. In order to make the proposed solution scalable, an epidemic protocol is also employed in this case. As the graph might include cycles, elemental requests are provided with a *timestamp*. Requests older than a pre-defined limit are discarded (so the system does not get overloaded).

Table 1. Double entry table constructed by MaA

Percentage of trustworthy received messages $\Psi[s, t, x = true]$		Target vertex			Probability of trustworthiness
		v_1^t	...	$v_{card(\mathfrak{N}_{v_i}^t)}^t$	
Source vertex	v_1^s	$\psi_{1,1}$	...	$\psi_{1,card(\mathfrak{N}_{v_i}^t)}$	$\frac{1}{card(\mathfrak{N}_{v_i}^t)} \sum_j \psi_{1,j}$
	...	...	...	...	...
	$v_{card(\mathfrak{N}_{v_i}^s)}^s$	$\psi_{card(\mathfrak{N}_{v_i}^s),1}$	...	$\psi_{card(\mathfrak{N}_{v_i}^s),card(\mathfrak{N}_{v_i}^t)}$	$\frac{1}{card(\mathfrak{N}_{v_i}^t)} \sum_j \psi_{card(\mathfrak{N}_{v_i}^s),j}$
Probability of trustworthiness		$\frac{1}{card(\mathfrak{N}_{v_i}^s)} \sum_j \psi_{j,1}$	...	$\frac{1}{card(\mathfrak{N}_{v_i}^s)} \sum_j \psi_{j,card(\mathfrak{N}_{v_i}^t)}$	

When a ERQ is received by a MoA (hosted in a vertex belonging to the n -th tier) it responds following a pre-defined pattern. For example, if MoA is visiting a temperature sensor, it generates an answer (known as “elemental answer”, EA) containing a preset value instead of the real temperature which is

unknown by upper components. The EA also includes the original ERQ.

$$EA = \{ERQ, ans_{msg}\}$$

The EA are sent to the root vertex v_i using also an epidemic protocol, in order to traverse all the possible paths. Each MoA gets hosted in the target vertices during a certain time (answering ERQ), and, when this time is over, they return to the root vertex.

When the root vertex receives the EA, it completes the information with the target vertex information (the vertex through which the root vertex v_i receives the response), $vertex_{trg} \in \mathcal{N}_{v_i}^t$.

$$EA = \{ERQ, ans_{msg}, vertex_{trg}\}$$

When v_i vertex has received all the information, the MaA constructs a double entry table (see Table 1). As EA should follow the pre-defined patterns, the MaA may evaluate the correctness (the trustworthiness) of the received information. Each cell in the double entry table is, then, completed with the corresponding percentage of trustworthy received messages. Each row is associated with a source vertex and each column with a target node.

The resulting double entry table, then, may be understood as a biparametric stochastic process $\Psi[s, t, x]$. Both parameters, s and t , represent the source s and target t vertices of the path to be studied. Random variable X is a binary variable indicating the event of the path starting in s and finishing in t to trust ($x = true$) or not ($x = false$). The created table would contain the process realization for $x = true$.

Then, each realization $\psi_{i,j}$ may be understood as the percentage of correct answers received through paths starting in v_i^s and finishing in v_j^t . In order to obtain the probability of trustworthiness of each source and target node, the statistical mean value is obtained (having all cases the same probability) considering percentages for all paths starting or finishing in a certain vertex.

As percentages in the same row or column are already calculated considering the same denominator (the number of performed ERQ), it is guaranteed the global probability add up to the unit. Thus, all incoming edges (on the one hand), and all outgoing edges (on the other hand) have associated probabilities meeting the basic probabilistic theory assumption (probabilities of all considered events add up to the unit). Edges in the global trust graph are not required to fulfill any requirement a priori, as they are (in the general case) independent components.

The obtained information (as it only describes the beginning and end of every path), it is not enough to obtain directly the trust graph. Nevertheless, as said in Section 3.2, the first step in the trust graph calculation process is to obtain the trust circle of v_i . In order

to do that, the trustworthiness of the individual nodes has to be obtained. As said, a vertex is trustworthy if, and only if, paths starting or finishing in it are also trustworthy. Thus, considering the created table contains samples of an unknown random variable (as in any other regular laboratory experiment), the most appropriate way of estimating the trustworthiness of vertices is the sample mean (see Table 1). That is, the probability of a vertex to be trustworthy it is obtained as the sample mean of the probabilities associated with each one of the paths starting or finishing in that node.

$$\bar{X} = \frac{1}{n} \sum_j X_j$$

Considering an epidemic protocol transmits in all directions with the same probability, every sample in the sample mean is weighted by the same coefficient.

Computation of these values is a very easy and fast procedure, so sources are much better used than in traditional statistical frameworks for trust management. It must be noted (besides) that, in all this calculation process, the trustworthiness of a link is considered together the trustworthiness of the target vertex of that link.

Once the probability of being trustworthy is calculated for every adjacent vertex to the component under study, various possibilities appear. First, if the probability to be trustworthy of a vertex is above the threshold θ_{max} , the component is considered trustworthy and all paths starting (or finishing) in this vertex are included in the trust graph. In the same way, if the probability to be trustworthy of a vertex is below the threshold θ_{min} , the component is considered untrustworthy and all paths starting (or finishing) in this vertex are removed from the trust graph.

In order to remove untrustworthy paths, the initial and/or final vertices have to be pruned. In Industry 4.0 systems, components cannot remove any peer, as these privileges are reserved to the system administrator. In that way, pruning algorithm only removes the edge that connects the root vertex v_i with the initial/final of the path to be pruned. No additional edges are created to maintain the connectivity. If the pruned path was the only outgoing/ingoing path, an alarm may be triggered to inform user and/or administrators about the situation.

For any other value θ that $\theta_{min} \leq \theta \leq \theta_{max}$, a new calculation process should be triggered. Considering the information provided by MoA, the MaA decides if untrustworthiness elements are located in the downlink (from root vertex to vertex hosting the MoA) or in the uplink. In the first case, source vertex

is requested to calculate its own trust graph. In second case, target vertex is equally requested. If a decision cannot be made about this issue, both vertices are directly requested to evaluate the trust graph.

Once requested nodes have finished the calculation process (following the same described procedure), they notify the root vertex this event with an acknowledgement message (ACK). This ACK has to indicate also if some paths have been pruned or no changes have been performed. In the first case, the probability to be trustworthy of the affected vertices is refreshed following the same procedure described above. If obtained probabilities are unclear another time, these values are considered, anyway, as the final results of the calculation process. In the second case no more actions are performed.

In any case, probabilities and trust graph and trust circle are refreshed periodically, in a continuous actualization process. The first time this process is triggered is after all ACK are received, and all untrustworthy paths pruned. From this point, the procedure is repeated periodically.

With the obtained results, the intermediate values are employed as weights to perform traffic engineering in the system and/or to include annotations (using, for example, Semantic Annotations) about the trustworthiness of the received messages.

The use of epidemic protocols makes the proposed technology very scalable, as the system get “flooded” in a fast way. However, depending on the system topology (more than on the number of components in the system), if several paths start/end in the same vertex and an untrustworthy component is located very deeply in the system, the calculation process may imply several steps and require more processing time than expected.

4. Experimental validation

In order to illustrate the behavior of the proposed algorithm, in this section we perform and provide some relevant simulations.

Simulations were performed using the (commercial) NS3 simulator, which implemented a platform based on TAP bridges and virtualization. The proposed platform turns independent the application layer and the network layer. The network layer is supported by NS3 simulator, where different libraries describing gossip protocols, mobile communications, wireless links, etc. are available. On the other hand, application layer is based on Linux paravirtualization,

as native C programming is the most adequate to implement.

In particular, NS3 scenarios recreate very similar conditions to real networks, and using the described platform, nodes in these simulation scenarios may exchange data and implement applications as real hosts do.

The simulation scenarios are supported by TAP bridges, which connect virtual Linux machines generated using LXC technologies and the Libvirt library with the NS3 simulator, and ghost nodes, the virtual entity that represent the virtual machine in the network simulator. All parameters are provided using a configuration script, where nodes, network interfaces, connections and applications are described. Although this approach is more complex than the configuration of regular NS3 simulations, it is a more flexible design. TAP bridges are configured as “ConfigureLocal”. In that way, TAP bridges are generated in a row, inheriting the configuration properties one from the other. In order to orchestrate in the most efficient way the different parallel processes that require this type of simulations, a desktop application was created (see Figure 6).

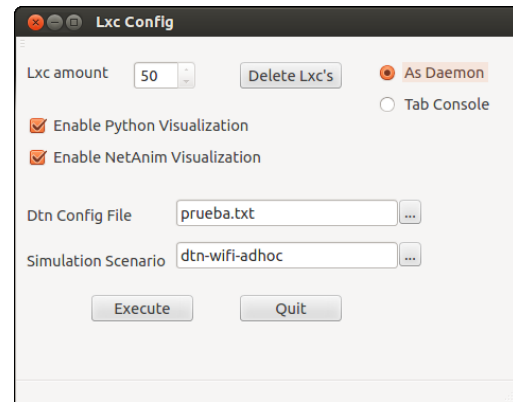


Fig. 6. Graphic management application

The employed simulation scenario consisted of an Industry 4.0 system composed of 250 components, distributed as follows:

- Physical layer (175 devices). It includes 100 smart transducers (temperature sensors and buzzers) based on Atmel microprocessors. Moreover, 70 Programmable Logic Controller (PLC) supported by Samsung Artik devices are simulated. Finally, 5 robotic arms are simulated.
- Control and data analytics (75 components). 25 network devices among routers, bridges

and hubs were considered. Besides 50 different middleware were also included (most of them focused on data translation).

- User applications (25). Finally, 25 user applications obtaining information from physical platform and representing it on a graphic interface are simulated.

Although, in a real scenario, components would present heterogeneous characteristics, this situation is a pending challenge which is not addressed in this paper. In order to evaluate the performance of the proposed solution, all components are provided with a mobile agent framework: Mobile-C [67]. Agents are described using XML documents containing the proposed algorithms in C code. In a real application, the described solution should be combined with other software engineering mechanisms to enable the implementation of mobile agent on heterogeneous CPS [60].

Two different scenarios were created with these devices, each one with a different system topology: (1) total connectivity and (2) branched graph.

In the described scenarios, the proposed algorithm for trust graph calculation was implemented and deployed. In particular, user applications were programmed to include this solution and perform calculations.

The total simulated time was two days (48 hours), for which a 15 hours simulation was required. Each simulation was repeated five times in order to compensate the possible random effects.

At $t_0 = 0$ five random components were programmed to behave in an untrustworthy way. During simulation, other components could also turn untrustworthy depending on random effects. Different (3) cyber-attacks were also considered and planned at $t_1 = 12h$, $t_2 = 24h$ and $t_3 = 36h$.

The numeric values assigned to the parameters described in the proposed technology are included in Table 2. Values are only illustrative, and they are not optimized or specifically designed for any particular situation. These two values may be modified in an independent manner, as they are two non-related variables. Although both parameters may be fixed as desired, in order to maximize the success rate, it is advisable to study the application scenario before selecting values. However, some important general remarks should be done.

In extremely secure or trustworthy scenarios, θ_{max} and θ_{min} parameters would tend to be very close and have low values. In that way, only components with several negative reports will be declared

malicious. On the other hand, in high-risk scenarios, both values will be high; as only one doubt about the trustworthiness of components is considered enough to prune this path. Using this two configurations, the calculation time will be severely reduced, as almost no recalculations will be done (θ_{max} and θ_{min} are very close, see Section 3), and paths will be classified during the first evaluation.

However, when no information about the scenario is available, all possibilities should be considered to have the same probability, and θ_{max} and θ_{min} parameters should be homogeneously distributed. In this case, several recalculations will be done, and required time to obtain a stable trust graph is higher. This “worst case” is considered in this experimental validation. For any other situation, performance will be improved.

Table 2. Numeric values of the solution parameters

Parameter	Value	Comments
θ_{max}	0.75	Values in the upper quartile are considered to be trustworthy
θ_{min}	0.25	Values in the lower quartile are considered to be untrustworthy
Timestamp limit	180 s	ERQ older than three minutes are discarded

Simulations are employed to perform four experiments.

In the first experiment, we evaluate the convergence time of the proposed solution and the correctness of the calculated trust graphs.

In the second experiment, obtained results from the proposed solution are compared to results obtained from state-of-the-art proposals. In particular, although no similar solution has been reported, other techniques with similar purposes may be found. In this case we have selected one of the most popular proposals: reputation and trust models [26][27]. This approach employs direct and indirect observations to evaluate if cyber-physical components are malicious and, then, isolate them.

In the third experiment, the solution performance is evaluated in a dynamic Industry 4.0 system (where components get in and out the system) and compared to an equivalent traditional message-passing solution. Basically, the mean converge time is evaluated. Components may appear at any available location (logical or physical) randomly. Different behavior patterns are considered. In particular, the probability a component to leave or enter the system follows a gaussian distribution with time, whose mean value

(or average stay time) is varied in each experiment realization.

Finally, in the fourth experiment, the global system overload caused by trust calculation and graph pruning is measured.

5. Results

In respect to the first experiment, figure 7 shows the evolution in the use of the untrustworthy paths. As can be seen, the use rate of the untrustworthy paths follows, for both system topologies, a sigmoid-like evolution.

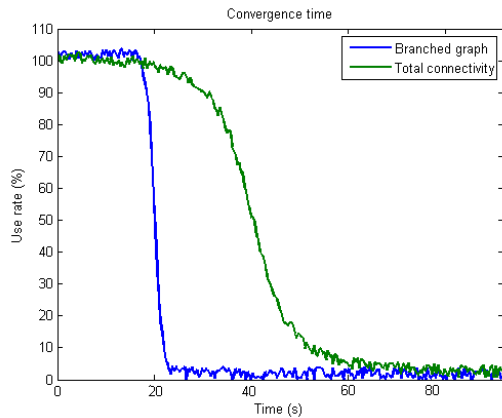


Fig. 7. Use of the untrustworthy paths

If a branched graph topology is considered, only twenty (20) seconds are required to isolate (almost totally) the untrustworthy paths. However, as the number of edges and links goes up, more time is required to traverse all possible paths, and more calculation time is needed to obtain the final trust graph. In that way, as can be seen, around sixty (60) seconds are necessary to locate all the untrustworthy paths in a system with total connectivity. Although this period may be too high for traditional Internet scenarios, Industry 4.0 applications inherit some special characteristics from current industrial systems. In particular, broadband communications are not usually deployed and components tend to exchange few messages (as production tasks usually take some time to finish). Thus, the potential impact of untrustworthy behaviors goes up much more slowly than in traditional Internet scenarios.

It must be noted that, once equilibrium is reached it is stable. Small variations of the curves are due to the statistical (stochastic) nature of the proposed technology.

Results in Figure 8 shows the success rate in locating and pruning the untrustworthy paths depending on the system topology and the type of untrustworthiness considered.

As can be seen, in general systems with a branched topology allow a higher success rate. In these graphs, in fact, traversing all possible paths is easier, as it is possible to define an ordered way to analyze the topology. Any case, in all cases success rate is above 80%. As can be seen, this difference is obvious for initial and random trustworthiness, but smaller for cyber-attacks. Any case, all values are enough high to consider the proposed solution a successful technique.

Initial untrustworthiness is detected in a more successful way, and untrustworthiness appearing in a random way are the most difficult to locate. Cyber-attacks, as they are usually pretty “noisy”, they are detected and untrustworthy paths pruned in a more successful manner.

It is also important to remark that the proposed technique is applicable to both scenarios (branched graph and total connectivity graph), so Figure 8 does not represent a comparison to show the obtained improvement. Below we are doing this analysis.

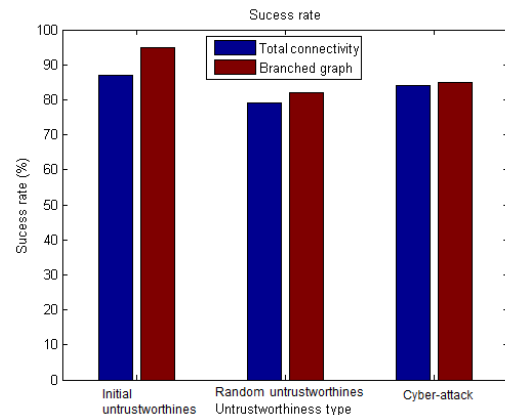


Fig. 8. Success rate

Once presented results from first experiment, we are discussing results from the second experiment. Figure 9 shows a comparison between results obtained from our proposal, and existing solutions' performance [27][63]. It is important to remark that both approaches are not equivalent, as application scenarios are not exactly the same, but this initial comparison may be employed to obtain first evidence about the obtained improvements.

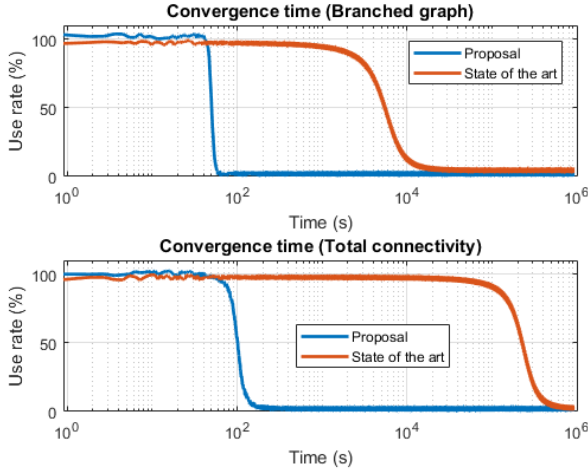


Fig. 9. Use of the untrustworthy paths (comparison to the state of the art)

In order to show results in the most adequate manner, X-axis is represented in logarithmic form. Using this method, differences between the proposed solution and existing technologies are clearly shown, but comparisons between results for the different scenarios are hard to be analyzed. Considering this fact, some relevant initial conclusions may be done. First, the proposed solution is up to three magnitude orders faster than existing technologies [27] based on reputation and trust models. The main cause for this improvement is the different and new approach to evaluate the path trustworthiness. While in trust and reputation models [63] observations are passively collected during system operation, in the proposed mechanism a proactive evaluation procedure is considered. Thus, previous proposals [63] require a lot of time (up to fifty hours, depending on the number of carried out transactions among components per time unit) to collect enough information to make a decision and prune untrustworthy paths. On the contrary, proposed mechanism performs an exhaustive proactive evaluation procedure which detects malicious components and paths in few minutes.

And, second, the scalability of the proposed technique is much better than existing message-passing approaches. In fact, as can be seen in Figure 9, when a branched graph increases its interconnections and turns into a total connectivity graph, previous mechanisms increment the required time to locate all malicious components and untrustworthy paths in two magnitude orders. Nevertheless, the proposed solution only duplicates that time. Therefore, using the proposed solution, the scalability curve has transformed from an exponential function to a linear law.

Previous experiment compares the proposed solution to existing trust evaluation technologies. On the contrary, third experiment compares the proposed agent-based solution to the equivalent message-passing algorithm. Besides, in this case, a more realistic scenario is considered, where components leave and enter the system dynamically. Figure 10 shows the obtained results for this experiment.

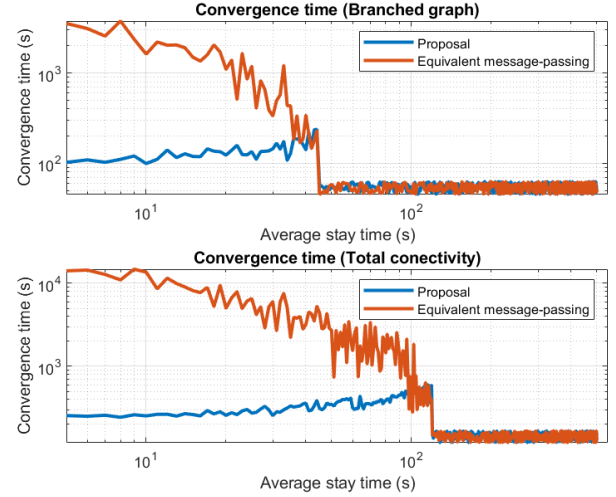


Fig. 10. Convergence time (comparison to message-passing algorithm)

As can be seen, both solution (agent-based and message-passing) have a pretty similar behavior when components remain in the same position for a long period (long enough to evaluate trust completely). In that case, both algorithms behave similar, and converge time is around values showed in Figure 7. However, this situation totally changes when components leave and enter dynamically the system. In that case, the use of agents is a key feature for a fast and efficient trust evaluation.

In fact, as ERQ do not include any information about the node identity, in the equivalent message-passing algorithm, identifiers from the communication subsystem must be employed. These identifiers are ephemeral, and trust is not correctly evaluated until each component remains enough time in one specific location. This requirement is more costly as components change their location more frequently, as can be seen in Figure 10. Converge time for the message-passing algorithm is up to two magnitude orders higher. On the contrary, agents are hosted by the component (vertex) under study, so it may be univocally recognized using serial numbers or other similar information which is globally unique. In that way,

even if components change their location, mobile agents recognize the components; and converge time is independent from the location and only depends on the time the components remain connected to the system. Any case, when components are moving around the system, converge time increases around 400% (although the specific amount depends on the behavior pattern of components).

Finally, results from the fourth experiment are analyzed to prove the proposed solution is lightweight. Two different indicators are measured to evaluate the overload in the system due to trust calculation: memory usage and consumed processing time. Table 2 shows the obtained measured.

Table 2. Resource consumption

Scenario	Period	Overload indicators	
		Memory usage	Processing time
Branched graph	Initial calculation	9.8%	11.2%
	Actualization	6.7%	2%
Total connectivity	Initial calculation	12.2%	16.1%
	Actualization	8%	3.9%

As can be seen, in all cases, the overload is higher during the initial calculation than during the (continuous) actualization processes. Any case, the mean aggregated overload is lower than 10% (around 9%). This value goes down to 5% (a reduction around 50%) during the continuous actualization process. In respect to memory usage, the overload is around 10%, although it may reach up to 12% for total connectivity graphs during the initial calculation. These values are a little bit higher for processing time, appearing an overload around 16% for total connectivity graphs during the initial calculation.

These values, for small resource constraint devices, show that the proposed technology is lightweight, as said before. In fact, previously reported lightweight security (encryption) techniques are proved to consume similar amounts of resources (an overload around 10%) [64]. Besides, other similar solutions (based on trust and reputation models) [63] are explicitly said to be implemented in powerful software components, as algorithms are complex and costly.

Actually, if common simple data gathering programs and algorithms are considered in sensor nodes, the generated overload by our solution is almost negligible in absolute values. However, if complex algorithms and costly software is being already executed,

this overload could be excessive; but this situation is inevitable for any proposed security solution.

6. Conclusions

In this work, a trust management technology to protect resource constraint systems against malicious components is proposed.

A mathematical formalization describing engineered resource constraint systems (particularized in Industry 4.0 systems) as directed graphs is described. Using the proposed notation and formalization, a model for trust in Industry 4.0 systems, defining important concepts such as the trust graph (absolute or relative) and the trust circle, is proposed. A probabilistic model based on a biparametric stochastic process is also described.

A platform based on two different types of agents support the calculation of the trustworthy paths in the graph describing an Industry 4.0 system. Untrustworthy paths are pruned and situations where a clear decision cannot be made, are solved using traffic engineering techniques.

Simulation results showed a success rate above 80% in all cases. Besides, a maximum calculation time of 600 seconds is required to locate all possible untrustworthy paths in total connectivity graphs. Comparisons to the state-of-the-art solutions show an improvement in the required calculation time and scalability. Besides, the generated system overload proves the proposed solution to be lightweight.

From the engineering point of view, different challenges must be faced when implementing the proposed solution. First, specific programming solutions for CPS must be employed, in order to solve problems such as timing (in our case the specific C-based language provided with microcontrollers was employed). Second, sometimes, mathematical expressions cannot be directly applied due to hardware characteristics (for example, no floating-point operation could be allowed). Then, adaptations are required. And, third, solutions to send mobile agents to all nodes under study must be coordinated with self-configuration technologies. That's basic to share network addresses, communications ports, etc., but if built-in communication protocols are employed, it may require an additional effort.

Finally, the system topology is a basic variable in the proposed technology, so future works should consider this variable in the calculation process in order to optimize as much as possible the use of sys-

tem resources (which, in Industry 4.0 systems, use to be scarce). In this sense, the proposed solution could be improved from a computational point of view if visibility graphs are employed [65]. Future works will consider also this technology.

Future works will also describe some specific solution for situations where the trustworthiness of paths is unclear.

Acknowledgements

Borja Bordel has received funding from the Ministry of Economy and Competitiveness through SEMOLA project (TEC2015-68284-R).

References

- [1] Lee EA. Cyber-physical systems-are computing foundations adequate. In: Position Paper for NSF Workshop On Cyber-Physical Systems: Research Motivation, Techniques and Roadmap (Vol. 2) 2006
- [2] Varghese A, Tandur D. Wireless requirements and challenges in Industry 4.0. In: 2014 International Conference on Contemporary Computing and Informatics (IC3I); 2014. p. 634-638.
- [3] Robles T, Alcarria R, Martín de Andrés D, Navarro M, Calero R, Iglesias S, et al. An IoT based reference architecture for smart water management processes. *JoWUA*. 2015; 6(1):4-23.
- [4] Martín de Andrés D, Alcarria R, Robles T, Morales Dominguez A. A Systematic Approach for Service Prosumerization in IoT Scenarios. In: Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS); 2013. p. 494-499.
- [5] Mosterman PJ, Zander J. Industry 4.0 as a cyber-physical system study. *Software & Systems Modeling*. 2016; 15(1):17-29.
- [6] Lee J, Bagheri B, Kao HA. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing Letters*. 2015; 3:18-23.
- [7] Bordel Sánchez B, Alcarria R, Martín D, Robles T. TF4SM: a framework for developing traceability solutions in small manufacturing companies. *Sensors*. 2015;15(11):29478-29510.
- [8] Yampolskiy M, Horváth P, Koutsoukos XD, Xue Y, Sztipanovits J. A language for describing attacks on cyber-physical systems. *International Journal of Critical Infrastructure Protection*. 2015;8:40-52.
- [9] Yampolskiy M, Horvath P, Koutsoukos XD, Xue Y, Sztipanovits J. Taxonomy for Description of Cross-Domain Attacks on CPS. In: Proceedings of the Second ACM International Conference on High Confidence Networked Systems; 2013. p. 135-142.
- [10] Wang, X., Zhang, G., Neri, F., Jiang, T., Zhao, J., Gheorghe, M., ... & Lefticaru, R. (2016). Design and implementation of membrane controllers for trajectory tracking of nonholonomic wheeled mobile robots. *Integrated Computer-Aided Engineering*, 23(1), 15-30.
- [11] Bordel B, Alcarria R, Sánchez-de-Rivera D. Detecting Malicious Components in Large-Scale Internet-of-Things Systems and Architectures. In: World Conference on Information Systems and Technologies; 2017. Springer, Cham. p. 155-165.
- [12] Jazdi, N. (2014, May). Cyber physical systems in the context of Industry 4.0. In Automation, Quality and Testing, Robotics, 2014 IEEE International Conference on (pp. 1-4). IEEE.
- [13] Borja Bordel, Ramón Alcarria, Diego Martín, Tomás Robles, Diego Sánchez de Rivera: Self-configuration in humanized Cyber-Physical Systems. *J. Ambient Intelligence and Humanized Computing* 8(4): 485-496 (2017)
- [14] Borja Bordel Sánchez, Ramón Alcarria, Álvaro Sánchez-Picot, Diego Sánchez de Rivera: A Methodology for the Design of Application-Specific Cyber-Physical Social Sensing Co-Simulators. *Sensors* 17(10): 2177 (2017)
- [15] Hofmann, E., & Rüsç, M. (2017). Industry 4.0 and the current status as well as future prospects on logistics. *Computers in Industry*, 89, 23-34.
- [16] Sommer, L. (2015). Industrial revolution-industry 4.0: Are German manufacturing SMEs the first victims of this revolution?. *Journal of Industrial Engineering and Management*, 8(5), 1512-1532.
- [17] Shrouf, F., Ordieres, J., & Miragliotta, G. (2014, December). Smart factories in Industry 4.0: A review of the concept and of energy management approached in production based on the Internet of Things paradigm. In *Industrial Engineering and Engineering Management (IEEM)*, 2014 IEEE International Conference on (pp. 697-701). IEEE.
- [18] Lee, J., Bagheri, B., & Kao, H. A. (2015). A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing Letters*, 3, 18-23
- [19] Tang, Y., Sun, Q., Liu, R., & Wang, F. (2018). Resource Leveling Based on Line of Balance and Constraint Programming. *Computer - Aided Civil and Infrastructure Engineering*, 33(10), 864-884.
- [20] Bordel, B., Alcarria, R., de Rivera, D. S., & Robles, T. (2018). Process execution in Cyber-Physical Systems using cloud and Cyber-Physical Internet services. *The Journal of Supercomputing*, 1-43.
- [21] Sánchez, B. B., Alcarria, R., de Rivera, D. S., & Sánchez-Picot, A. (2016). Enhancing Process Control in Industry 4.0 Scenarios using Cyber-Physical Systems. *JoWUA*, 7(4), 41-64
- [22] Chen, X., & Wang, L. (2017). A cloud-based trust management framework for vehicular social networks. *IEEE Access*, 5, 2967-2980.
- [23] Wang, S., Wan, J., Zhang, D., Li, D., & Zhang, C. (2016). Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination. *Computer Networks*, 101, 158-168.
- [24] Yan Z, Zhang P, Vasilakos AV. A survey on trust management for Internet of Things. *Journal of network and computer applications* 2014;42:120-134.
- [25] Geetha S. Social internet of things. *World Scientific News*. 2016:41-76.
- [26] Bao F, Chen R, Chang M, Cho JH. Hierarchical trust management for wireless sensor networks and its applications to trust-based routing and intrusion detection. *IEEE transactions on network and service management*. 2012;9(2):169-183.

- [27] Bao F, Chen R. Trust management for the internet of things and its application to service composition. In: IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks (WoWMoM); 2012. p. 1-6.
- [28] Daly EM, Haahr M. Social network analysis for information flow in disconnected delay-tolerant MANETs. *IEEE Transactions on Mobile Computing*. 2009;8(5):606-621.
- [29] Zhang M, Selic B, Ali S, Yue T, Okariz O, Norgren R. Understanding Uncertainty in Cyber-Physical Systems: A Conceptual Model. In: *Modelling Foundations and Applications: 12th European Conference, ECMFA 2016; 2016 July 6-7; Vienna, Austria: Springer; 2016*. p. 247-264.
- [30] Hasan S, Curry E. Approximate semantic matching of events for the internet of things. *ACM Transactions on Internet Technology (TOIT)*. 2014;14(1):2.
- [31] Aggarwal CC, Ashish N, Sheth A. The internet of things: A survey from the data-centric perspective. In: *Managing and mining sensor data*. Springer US; 2013. p. 383-428.
- [32] Chen D, Chang G, Sun D, Li J, Jia J, Wang X. TRM-IoT: A trust management model based on fuzzy reputation for internet of things. *Computer Science and Information Systems*. 2011;8(4):1207-1228.
- [33] Truong NB, Um TW, Lee GM. A reputation and knowledge based trust service platform for trustworthy social internet of things. In: *19th conference in Innovations in Clouds, Internet and Networks (ICIN); 2016 March 1-3; Paris, France*. 2016
- [34] Xiao H, Sidhu N, Christianson B. Guarantor and reputation based trust model for social internet of things. In: *International Wireless Communications and Mobile Computing Conference (IWCMC); 2015 Aug 24-28; Dubrovnik, Croatia; 2015*. p. 600-605.
- [35] Guha R, Kumar R, Raghavan P, Tomkins A. Propagation of trust and distrust. In: *Proceedings of the 13th international conference on World Wide Web; 2004 May 17-22; New York, USA; ACM; 2004*. p. 403-412.
- [36] Jiang T, Baras JS. Graph algebraic interpretation of trust establishment in autonomic networks. In: *Preprint Wiley Journal of Networks; Wiley; 2009*.
- [37] Trajdos, P., & Kurzynski, M. (2018). A Correction Method of a Binary Classifier Applied to Multi-label Pairwise Models. *International journal of neural systems*, 28(09), 1750062.
- [38] Golbeck J, Parsia B, Hendler J. Trust networks on the semantic web. In: *Proceedings of the 7th International Workshop on Cooperative information agents (CIA-2003)*. 2003:238-249.
- [39] Shmatikov V, Talcott C. Reputation-based trust management. *Journal of Computer Security*. 2005;13(1):167-190.
- [40] Ziegler CN, Lausen G. Spreading activation models for trust propagation. In: *Proceedings of IEEE International Conference on e-Technology, e-Commerce and e-Service; 2004 March 28-31; Taipei, Taiwan. IEEE; 2004*. p. 83-97.
- [41] Theodorakopoulos G, Baras JS. On trust models and trust evaluation metrics for ad hoc networks. *IEEE Journal on selected areas in Communications*. 2006;24(2):318-328.
- [42] Rostami, S., & Neri, F. (2016). Covariance matrix adaptation pareto archived evolution strategy with hyper-volume-sorted adaptive grid algorithm. *Integrated Computer-Aided Engineering*, 23(4), 313-329
- [43] Blaze M, Feigenbaum J, Lacy J. Decentralized trust management. In: *Proceedings of the IEEE Symposium on Security and Privacy; 1996 May 6-8; California, USA. IEEE; 1996*. p. 164-173.
- [44] Cornelli F, Damiani E, di Vimercati SDC, Paraboschi S, Samarati P. Choosing reputable servants in a P2P network. In: *Proceedings of the Eleventh International World Wide Web Conference; 2002 May 07-11; Honolulu, USA. ACM; 2002*. p. 376-386.
- [45] Liu, F., & Er, M. J. (2012). A novel efficient learning algorithm for self-generating fuzzy neural network with applications. *International journal of neural systems*, 22(01), 21-35.
- [46] Kretschmar H, Stachniss C, Grisetti G. Efficient information-theoretic graph pruning for graph-based SLAM with laser range finders. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS); 2011 Sep. 25-30; San Francisco, USA. IEEE; 2011*. p. 865-871.
- [47] Dai F, Wu J. Distributed dominant pruning in ad hoc networks. In: *Proceedings of the IEEE International Conference on Communications; 2003 May 11-15; Anchorage, Alaska, USA. IEEE; 2003*. p. 353-357.
- [48] Rostami, S., Neri, F., & Epitropakis, M. (2017). Progressive preference articulation for decision making in multi-objective optimisation problems. *Integrated Computer-Aided Engineering*, 24(4), 315-335.
- [49] Takimoto E, Warmuth MK. Predicting nearly as well as the best pruning of a planar decision graph. *Theoretical Computer Science*. 2002;288(2):217-235.
- [50] Navigli R, Velardi P, Faralli S. A graph-based algorithm for inducing lexical taxonomies from scratch. In *International Joint Conference on Artificial Intelligence; 2011 July 16-22; Barcelona, Spain. 2011*. p. 1872-1877.
- [51] Sixtus A, Ortmanns S. High quality word graphs using forward-backward pruning. In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing; 1999 March 15-19; Phoenix, Arizona, USA. IEEE; 1999*. vol 2. p. 593-596.
- [52] Kotz, D., & Gray, R. S. (1999). Mobile Agents and the Future of the Internet. *Operating systems review*, 33(3), 7-13.
- [53] Godfrey, W. W., Jha, S. S., & Nair, S. B. (2013, April). On a mobile agent framework for an internet of things. In *Communication Systems and Network Technologies (CSNT), 2013 International Conference on* (pp. 345-350). IEEE.
- [54] Fortino, G., Guerrieri, A., Russo, W., & Savaglio, C. (2014, May). Integration of agent-based and cloud computing for the smart objects-oriented IoT. In *Computer Supported Cooperative Work in Design (CSCWD), Proceedings of the 2014 IEEE 18th International Conference on* (pp. 493-498). IEEE.
- [55] Cimellaro, G. P., Mahin, S., & Domaneschi, M. (2019). Integrating a Human Behavior Model within an Agent - Based Approach for Blasting Evacuation. *Computer - Aided Civil and Infrastructure Engineering*, 34(1), 3-20.
- [56] Bahrami, S. (2017). Design of a Mobile Agents Based Solution to Distributional Management of Computer Networks, Taking into Account the Security Mechanisms. *Engineering, Technology & Applied Science Research*, 7(5), 2026-2029.

- [57] Vigna, G. (Ed.). (2003). *Mobile agents and security* (Vol. 1419). Springer.
- [58] Derler, P., Lee, E. A., & Vincentelli, A. S. (2012). Modeling cyber-physical systems. *Proceedings of the IEEE*, 100(1), 13-28.
- [59] Rajhans, A., Bhawe, A., Ruchkin, I., Krogh, B. H., Garlan, D., Platzer, A., & Schmerl, B. (2014). Supporting heterogeneity in cyber-physical systems architectures. *IEEE Transactions on Automatic Control*, 59(12), 3178-3193.
- [60] Lin, J., Sedigh, S., & Hurson, A. R. (2011, May). An agent-based approach to reconciling data heterogeneity in cyber-physical systems. In *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW)*, 2011 IEEE International Symposium on (pp. 93-103). IEEE.
- [61] Chess, D., Harrison, C., & Kershenbaum, A. (1996, July). Mobile agents: Are they a good idea?. In *International Workshop on Mobile Object Systems* (pp. 25-45). Springer, Berlin, Heidelberg.
- [62] Wan, J., Chen, M., Xia, F., Di, L., & Zhou, K. (2013). From machine-to-machine communications towards cyber-physical systems. *Computer Science and Information Systems*, 10(3), 1105-1128.
- [63] Bordel, B., Alcarria, R., De Andrés, D. M., & You, I. (2018). Securing Internet-of-Things systems through implicit and explicit reputation models. *IEEE Access*, 6, 47472-47488.
- [64] Bordel, B., Orúe, A. B., Alcarria, R., & Sánchez-de-Rivera, D. (2018). An intra-slice security solution for emerging 5G networks based on pseudo-random number generators. *IEEE Access*, 6, 16149-16164.
- [65] Ahmadlou, M., & Adeli, H. (2012). Visibility graph similarity: A new measure of generalized synchronization in coupled dynamic systems. *Physica D: Nonlinear Phenomena*, 241(4), 326-332.
- [66] Lee J, Bagheri B, Kao HA. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing Letters*. 2015;3:18-23.
- [67] Chen, B., Cheng, H. H., & Palen, J. (2006). Mobile - C: a mobile agent platform for mobile C/C++ agents. *Software: Practice and Experience*, 36(15), 1711-1733.
- [68]