Future Generation Computer Systems 83 (2018) 116–131

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

An intelligent, uncertainty driven management scheme for software updates in pervasive IoT applications

Kostas Kolomvatsos

Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Ilisia, 15784, Athens, Greece
Department of Computer Science, University of Thessaly, Papasiopoulou 2-4, 35100, Lamia, Greece

HIGHLIGHTS

- We propose an intelligent scheme for software updates in Pervasive Computing and IoT.
- The proposed model monitors a set of network performance metrics.
- An ensemble forecasting scheme provides insights for future values.
- A load balancing mechanism 'distributes' the nodes in the available time slots.
- We propose a decision making mechanism based on a neural network.

ARTICLE INFO

Article history:
Received 4 November 2016
Received in revised form 12 October 2017
Accepted 21 January 2018
Available online 3 February 2018

Keywords:
Internet of Things
Updates management
Ensemble forecasting
Load balancing
Neural networks

ABSTRACT

The era of the *Internet of Things* (IoT) involves a huge number of autonomous devices (nodes) capable of monitoring and interacting with their environment. The autonomous devices are also able of being interconnected, thus, they can exchange data. Pervasive computing applications can be built on top of this infrastructure offering efficient solutions for multiple domains. Nodes can execute intelligent, light-weight processing of the collected data being capable of responding in case of events. Apart from the software necessary to perform the discussed processing tasks, nodes are coming with pre-installed software necessary to perform basic functionalities e.g., communication. When nodes act in dynamic environments, it is necessary to update the software necessary for their functionalities. Updates involve software extensions and patches important to secure a high level performance of the IoT nodes. In this paper, we propose a distributed updates management scheme enhancing the autonomous nature of nodes. Legacy models deal with centralized approaches (i.e., a central server) where complex algorithms are adopted to derive the protocols for the distribution of the updates. In our approach, each node is responsible to, independently, initiate and conclude the update process. The central server is responsible only for indicating when the updates are available to the nodes. Every node monitors a set of performance metrics (either for the node itself or the network) and based on an *intelligent scheme* decides the appropriate time to conclude the update process. We adopt an ensemble forecasting model on top of a pool of estimators and an optimization model to derive the right time for initiating the update process. We are based on the solution of the known Santa Fe bar problem to perform load balancing in the retrieval of the updates. The aim is to have the nodes deciding the conclusion of the update process in different time intervals, thus, to keep the load of the network at low levels. We provide specific formulations and the analysis of our problem while extensive simulations and a comparison assessment reveal the advantages of the proposed solution.

1. Introduction

The *Internet of Things* (IoT) and *Pervasive Computing* (PC) set new challenges in the research community concerning the development of new services and applications. The interconnection of numerous IoT nodes with the help of wireless technologies (e.g., *Wireless Sensors networks* - WSNs) facilitates the creation of new PC services. PC involves the adoption of numerous devices embedded into everyday objects to support intelligent applications. The transition from closed networks to interconnected autonomous nodes that are capable of interacting with their environment and perform simple processing tasks should be supported by intelligent applications increasing the quality of services that end users enjoy. IoT nodes are, usually, characterized by limited

E-mail addresses: kostasks@di.uoa.gr, kolomvatsos@cs.uth.gr.

https://doi.org/10.1016/j.future.2018.01.036
0167-739X/

computational capabilities (i.e., constrained devices) that pose a set of requirements for any novel application. These nodes come with the software/firmware of the respective manufacturers. Two main challenges may positively affect their performance: (i) the application of software updates that will extend the provided functionalities; (ii) the application of firmware updates that will cover any gaps present in the already installed firmware, thus, securing the IoT nodes (e.g., against attacks). It is imperative to apply software/firmware updates as well as patches and extensions during nodes' interaction with the environment. The challenge is to apply these updates in the entire network. Once nodes are in operation, they will start applying software updates that should be concluded as soon as possible. However, every node performs some tasks significant for supporting applications, thus, applying updates should not disturb, if possible, nodes from their initial goal. Updates must be delivered in a way that conserves the limited bandwidth and intermittent connectivity of a device and eliminates the possibility of compromising functional safety.

Various models have been proposed for the application of updates in WSNs [1–14]. To the best of our knowledge, the vast majority of them deal with centralized systems. A central authority/server is responsible to disseminate the updates to the network. Usually, remote wireless reprogramming is applied for all nodes to distribute the updates. Due to the increased size or the multiple assignments of the updates, a set of techniques have been proposed to alleviate the burden of the numerous broadcasted messages. *Incremental updates*, *compression*, *diffusion* and various dissemination protocols are proposed to support the most effective solution. However, these techniques have a number of disadvantages. Incremental updates require an efficient management mechanism to maintain the updates history while compression and, accordingly, decompression require complex processes executed by the central server and nodes. The additional overhead for the management of incremental and compression/decompression schemes becomes significant, thus, it limits the central server's and nodes' performance. In addition, incremental techniques should incorporate mechanisms for handling the heterogeneity of nodes. The design of a reliable dissemination protocol that could handle such heterogeneity is a real challenge. Heterogeneity makes the definition of a common affordable overhead very difficult as it depends on the state of each node. Diffusion requires an increased number of messages especially in dense networks that will lead the nodes to spend their resources for communication. New trends in the updates management involve the maintenance of a virtual machine in the nodes responsible to host the software that will update a part of the firmware [15,16]. Such approaches are capable of manipulating the local processing of an IoT node and its interactions with other nodes. They are mainly based on the *Constrained Application Protocol* (CoAP) defined to allow applications to interact with physical objects. In any case, a centralized approach suffers from the following drawbacks: (i) the central server is the main responsible for delivering the updates, thus, it should apply complex algorithms and protocols to conclude the update process without affecting the network performance; (ii) the central server cannot be aware of the real number of nodes connected in the network, thus, cannot be aware if all nodes have received and applied the updates; (iii) when the central server distributes the updates, nodes should interrupt their processing tasks to apply the updates disturbing them from the initial goal; (iv) the network is flooded by update distribution messages limiting the available bandwidth and reducing the network performance.

In this paper, we try to alleviate the aforementioned problem and propose a distributed scheme for applying updates. We propose a model that builds on top of the autonomous nature of nodes. Our model does not focus on *how* updates will be applied locally (e.g., through parts or as a whole) but *when* nodes should initiate the update process (i.e., retrieval and application/installation of the update). The proposed scheme could be combined with any model implemented for installing updates locally. Our scheme alleviates the central system from the burden of finding the appropriate means for delivering the updates. Through our approach, the central server should not be aware of any information related to the nodes and the underlying network and acts only as a 'repository'. The central server sends a lightweight message, indicating the presence of an update. We consider two schemes: (i) the server defines a deadline till which the update should be applied (e.g., a security or a firmware update — the more critical the update, the more strict/short the deadline); (ii) the server does not impose a deadline (e.g., for a software extension), however, the sooner a node applies the update, the better for its performance. Nodes undertake the responsibility of retrieving and applying the updates when they observe that it is the appropriate time to perform the process. Nodes are capable of collecting information related to the performance of the network. Specific performance metrics could be taken into account like the *bandwidth*, the *error rate*, the *latency* and so on. Our mechanism aims to intelligently support the nodes and provide a decision making scheme that finds the right time for initiating the update process. The update process is initiated when the performance of the network is of high quality to efficiently support a fast and secure conclusion. Apart from the network performance, each node takes into consideration the current effort it spends in its main tasks.

In our approach, we contribute with:

- a novel, performance-aware mechanism for deciding the appropriate time to conclude an *update process* $\mathcal{P}$. The proposed mechanism takes into consideration not only the load of each node but also the performance of the network. Nodes initiate and conclude $\mathcal{P}$ only when they see that the network's performance is at acceptable levels to support the efficient application of the updates. Moreover, we propose a distributed model that enhances the autonomous nature of nodes, thus, there is no need to involve the central update server in the process and adopt complicated protocols and communication schemes;
- a load balancing scheme adopted to alleviate the burden of the network when nodes activate $\mathcal{P}$. The load balancing aspect of our work secures that bottlenecks in the network will be eliminated when an update is available in the central server. This approach secures the efficient and uninterrupted application of the updates;
- an ensemble forecasting scheme for having future insights of the network performance (applied on top of multiple performance metrics). The ensemble forecasting model helps us to avoid the disadvantages of every single forecasting scheme while involving multiple parameters that depict various aspects of the network's performance (e.g., bandwidth, latency). Based on this scheme, we are capable of deciding the right time to conclude $\mathcal{P}$ taking into consideration the future behaviour of the network;
- a decision making mechanism for deriving the appropriate time when $\mathcal{P}$ will be activated. The decision mechanism is based on the current view on the network's and nodes' performance but also on their future behaviour as estimated by the ensemble forecasting model. The proposed mechanism is realized through an *Artificial Neural Network* (ANN). The ANN results the final decision about the immediate initiation of $\mathcal{P}$ or not.

The proposed model leads the nodes to enjoy the best possible performance to support $\mathcal{P}$. As the best performance, we define the highest possible network performance and the limited load of nodes. In addition, our model maximizes the performance of