Multi-criteria IoT Resource Discovery: A Comparative Analysis

Luiz H. Nunes¹,², Julio C. Estrella¹, Charith Perera³, Stephan Reiff-Marganiec⁴ and Alexandre N. Delbem¹

¹ University of São Paulo (USP), Institute of Mathematics and Computer Science (ICMC), São Carlos-SP, Brazil
² Instituto Federal de São Paulo (IFSP), Matão-SP, Brazil
³ Faculty of Maths, Computing and Technology, The Open University, Walton Hall, Milton Keynes, MK7 6AA - UK
⁴ University of Leicester, University Road, Leicester, LE1 7RH - UK

SUMMARY

The growth of real world objects with embedded and globally networked sensors allows to consolidate the Internet of Things paradigm and increase the number of applications in the domains of ubiquitous and context-aware computing. The merging between Cloud Computing and Internet of Things named Cloud of Things will be the key to handle thousands of sensors and their data. One of the main challenges in the Cloud of Things is context-aware sensor search and selection. Typically, sensors require to be searched using two or more conflicting context properties. Most of the existing work uses some kind of multi-criteria decision analysis to perform the sensor search and selection, but does not show any concern for the quality of the selection presented by these methods. In this paper, we analyse the behaviour of the SAW, TOPSIS and VIKOR multi-objective decision methods and their quality of selection comparing them with the Pareto-optimality solutions. The gathered results allow to analyse and compare these algorithms regarding their behaviour, the number of optimal solutions and redundancy.

Copyright © 2016 John Wiley & Sons, Ltd.

KEY WORDS: Internet of Things, Resource Discovery, Multi Objective, Optimization

1. INTRODUCTION

The Internet of Things (IoT) is an ecosystem that interconnects physical objects with telecommunication networks, joining the real world with the cyberspace and enabling the development of new kinds of services and applications. The IoT world is composed of small sensors and actuators embedded in the objects such as electronic devices (e.g. smartphones or tablets), clothes, alarm systems, cars, domestic appliances and industrial machines, which are capable of interacting with each other and their environment.

Recently, the number of devices has grown rapidly and it is anticipated that between 2015 and 2016 about 20 billion devices will be connected to the Internet creating a market of around 91.5 billion dollars [1]. These things generate an amount of data which cannot be handled in a standalone power-constrained IoT environment. The integration of IoT with cloud computing, named Cloud of Things (CoT), can facilitate unprecedented ubiquitous sensing services and powerful resources to process sensing data streams beyond the capability of individual things [2].

Different domains can benefit from CoT applications such as logistics [3, 4], healthcare [5], smart cities [3], environmental monitoring [6, 7] and assisted driving [4, 8]. However, the CoT poses new challenges as it needs to combine different types of services provided by multiple stakeholders and support a large number of users and devices. One of these challenges is to provide a set of tools...
and environments for development of dynamic applications and ensure their seamless execution to meet the Quality of Context (QoC) and Quality of Service (QoS) requirements imposed by different kinds of applications [9].

While the nature of the CoT makes it suitable for provisioning the aforementioned services, ensuring their QoS and QoC requirements imposes complex challenges such as the resource constrained environment, redundant data, heterogeneity of the sensors nodes, dynamic network topology and size, and an unreliable communication medium. These factors can affect the user experience [10]. In addition, it is common to find two or more conflicting QoS and QoC requirements in this kind of service.

Thus, several papers such as [11, 12, 13] use some kind of Multiple-Criteria Decision Analysis (MCDA) to perform the sensor search and selection to achieve the best trade-off between the QoS and QoC properties. On the other hand, these papers do not show any concern about the quality of the selection presented by these methods regarding aspects like redundancy and dispersion of the selected sensors.

In this paper, we present a qualitative study of three MCDA methods used to establish the relative importance of multiple attributes and alternatives. In particular, we investigate the behaviour of the Simple Additive Weight method (SAW), the Technique for the Order of Prioritisation by Similarity to Ideal Solution (TOPSIS) and ViseKriterijumska Optimizacija I Kompromisno Resenje (VIKOR) under different conditions. We also analyse the quality of the solutions proposed by these methods comparing them with the available Pareto optimal solutions. The scientific contributions of our work can be summarized as:

- We proposed a methodology that can be used to compare different sensor search techniques from a quality of search perspective. We have demonstrated how our proposed methodology works using three different sensor search techniques.
- We examined the impact of ‘number of context-properties’ towards the overall quality of sensor search.
- We examined the impact of ‘number of sensors required to be selected’ towards the overall quality of sensor search.
- We evaluated and compared the overall quality of sensor search between three different Multi-Criteria Decision Analysis (MCDA) techniques, namely Simple Additive Weight method (SAW), the Technique for the Order of Prioritisation by Similarity to Ideal Solution (TOPSIS) and ViseKriterijumska Optimizacija I Kompromisno Resenje (VIKOR).

The paper is organized as follows: Section 3 describes the analysed Multiple-criteria decision-making algorithms and the methodology used to evaluate them. The results are then discussed in Section 4. Section 5 presents a literature review of existing approaches for sensor search and selection. Finally, the conclusions and directions for future work are presented in Section 6.

2. BACKGROUND

One of the most accepted definitions of what is IoT is described by Vermesan et al. (2011) [14]: “The IoT aims to allow people and things to be connected at any time or place with anything or anyone by any path, network or service”. Usually, the IoT is considered as a three-layer architecture, as represented by Figure 1a, showing the perception layer, network layer and application layer [15, 2]. On the other hand, some authors such as Khan et al. (2012) [16], Aazam and Huh (2014) [2] and Fersi (2015) [17] consider two extra layers named as middleware and business, as show in Figure 1b. The main layers objectives can be summarized as:

- **Perception layer:** It is main function is to perceive and collect the real environment information and bring them to the virtual environment. Sensors, bar code labels bar, radio-frequency identification devices (RFID), GPS and cameras are concentrated in this layer [2]. These devices can be described by metadata or specific languages such as SensorML,
OGC/SWE, SSN W3C, HyperCat and semantic models to enable its use by the up layers [18].

- **Network layer**: It is responsible for transporting data from the perception layer to be processed. The transmission medium can use wired networks or wireless networks such as 3G, UMTS, Wifi, Bluetooth, infrared and ZigBee depending directly on the types of sensor devices and the environment in which they are deployed [16].

- ** Middleware layer**: Its goal is to offer services and store data received from the network layer. Their services must process the information and make automated decisions based on their results [16, 2]. Currently, there are several middleware solutions such as GSN [19] and openIoT [20] to support the management of sensor networks. Usually these solutions are able to abstract the sensors available in the perception layer and offer their resources as a service to end users.

- **Application layer**: It presents data from the network layer or middleware layer. This layer must be concerned to present the information according to the specifications or constraints of an user [2].

- **Business layer**: It is responsible for system management including its applications and services. It defines the business models, graphics and execution flows based on data received from the application layer. The success of IoT depends directly on establishing good business models to analyze the results and determine future business strategies [16].

Nowadays, there is much research conducted for the different layers of the IoT architecture, which aim to solve problems related to interoperability, scalability, reliability, data management, privacy and security. One of the most significant challenges involves the middleware layer. Specifically, concentrating on how to support the search and selection of sensors regarding the QoS and QoC properties determined by a user [9].

### 3. MULTIPLE CRITERIA DECISION ANALYSIS

Multiple criteria decision analysis (MCDA) refers to making decisions in the presence of multiple, usually conflicting, criteria [21]. MCDA algorithms aims to aid in the judgement of the decision making team using a set of objectives and criteria, estimating their relative importance weights and, establishing the contribution of each option regarding to each performance criterion [22].

An MCDA problem can be described using an analysis matrix \( (M \times N) \) in which element \( q_{ij} \) represents the performance of the option according to the decision criterion \( c_j \) in different and non-comparable units and scales, as represented in Equation 1. The evaluation matrix is used to represent the relative performance of \( q'_{ij} \) using a value/utility function to enable comparisons between the different criteria [23].
All MCDA algorithms explicitly define their options and contributions to each criterion, but differs in how they combine the input data. Although MCDA problems are found in different contexts, they usually share common features such as multiple attributes/criteria often forming a hierarchy, conflict among criteria, hybrid nature, uncertainty, large scale and assessments that may not be conclusive [21]. Sections 3.1 to 3.3 describe three MCDA methods, namely SAW, TOPSIS and VIKOR algorithms for MCDA. Section 3.4 proposes the use of the Pareto-Optimality based on the proposed criterion to evaluate the applied MCDAs to select a subset of sensors.

3.1. SAW

The Simple Additive Weighting (SAW) method is one of the most popular MCDA methods [24, 25]. It provides the additive properties to calculate the final score of alternatives used for weight determinations and preferences, which is the basis of other MCDA methods such as the Analytic Hierarchy Process (AHP) and Preference ranking organization method for enrichment evaluation (PROMETHEE) [24]. According to [25], SAW is used in several application domains such as supply chain management, personnel selection problems, project manager selection and facility location selection.

SAW uses an evaluation score to rank each available option. The score is obtained using a normalized criteria value multiplied by a weight. The options are ranked in descending order according to their final score, which is the sum of the scores for individual criteria [26]. SAW algorithm can be summarized by the following three steps [23]:

1. Normalize the analysis matrix \( Q \) described in Equation 1 to \( Q' \) according to Equation 2 if the criterion should be maximized or the Equation 3 if the criterion should be minimized.

\[
q'_{ij} = \begin{cases} 
\frac{q_{ij} - q_{j\text{min}}}{q_{j\text{max}} - q_{j\text{min}}} & \text{for a criterion to be maximized} \\
\frac{q_{j\text{max}} - q_{ij}}{q_{j\text{max}} - q_{j\text{min}}} & \text{for a criterion to be minimized}
\end{cases}
\]

(2)

(3)

2. Compute the score vector \( \phi \) of each available option. Each score \( q'_i \) can be calculated using Equation 4, where \( w_j \) corresponds to the criterion weight and \( N \) represents the number of criteria in the evaluation matrix.

\[
\phi(q'_i) = \sum_{j=1}^{N} w_j \cdot q'_{ij} \]

(4)

3. Sort options \( q_i \) in decreasing order according to the score \( \phi(q'_i) \) to get the ranking of suitable options.

3.2. TOPSIS

TOPSIS explores the attribute information to provide a set of ranked alternatives and requires independent that attribute preferences. The application domains that uses the TOPSIS method has been Supply Chain Management and Logistics, Design, Engineering and Manufacturing Systems, Business and Marketing Management, Health, Safety and Environment Management, Human Resources Management, Energy Management, Chemical Engineering and Water Resources Management [27]. TOPSIS sorts a set of options according to the Euclidean distance from the ideal and negative-ideal solutions. Each option is normalized using a specific criterion value. The ideal
A DEMONSTRATION OF THE SOFTW. PRACT. EXPER. CLASS FILE

solution represents the most desirable level of each criterion across the options under consideration, while the negative-ideal solution reflects the worst-desirable level of each criterion. The options are ranked regarding their closeness to the ideal solution and farness to the negative-ideal solution [23]. The TOPSIS algorithm can be summarized in the following steps [28]:

1. Normalize the analysis matrix $Q$ to $Q'$ according to the Equation 5:

$$q'_{ij} = \frac{q_{ij}}{\sqrt{\sum_{i=1}^{N}(q_{ij})^2}}$$

where $N$ represents the number of options in the evaluation matrix.

2. Determine the positive ideal points ($p_{+j}$) and the negative ideal points ($p_{-j}$) of all objective functions using the analysis matrix. For a maximization criterion, the positive ideal and the negative ideal points can be calculated using Equations 6 and 7 respectively:

$$p_{+j} = \max_i(q'_{ij})$$

$$p_{-j} = \min_i(q'_{ij})$$

3. Compute the distances to the positive ideal solution and ($s_{i+}$) and the negative ideal solution ($s_{i-}$). The distance of each option $q'_i$ to the ideal solution $p_{+j}$ and the ideal negative solution $p_{-j}$ is given by Equations 8 and 9:

$$s_{i+} = \sqrt{\sum_{j=1}^{n}(q'_{ij} - p_{+j})^2}$$

and

$$s_{i-} = \sqrt{\sum_{j=1}^{n}(q'_{ij} - p_{-j})^2}$$

4. Calculate the relative closeness to the ideal solution. The relative closeness of $q_i$ to $p_{+j}$ and $p_{+j}$ represented by ($c_{i+}$) can be calculated according to Equation 10.

$$c_{i+} = \frac{s_{i-}}{s_{i+} - s_{i-}}$$

5. Sort options $q_i$ in increasing order according to the relative closeness to $c_{i+}$.

3.3. VIKOR

The basic concepts of VIKOR is a compromise programming used to get the most satisfactory option by the results of the individual and group regrets. This method has been widely used in several applications fields, such as: location selection, environmental policy and data envelopment analysis [29].

VIKOR introduces the multicriteria ranking index based on the particular measure of closeness to the ideal solution. The alternatives are evaluated according to all established criteria and ranks them according to: i) the minimal distance to the ideal point, ii) the maximum group utility for the majority and, iii) the minimum individual regret of the opponent. VIKOR algorithm can be summarized according to the follow steps [23], [28]:

1. Determine the best and the worst values for all criteria in $Q$. For a maximization criterion, the best and worst criteria values represented by $q^*_j$ and $q^-_j$ can be calculated respectively according to Equations 11 and 12:

$$q^*_j = \max_i(q'_{ij})$$

$$q^-_j = \min_i(q'_{ij})$$

2. Compute the utility measure and the regret measure. The utility measure represented by $S_i$ is used to show the average gap of our options and can be calculated according to Equation 13, where $w_j$ corresponds to the criteria weights, expressing their relative importance. A regret measure represented by $R_i$ is used to show the maximal gap for improvement priority and it can be calculated according to Equation 14.
3. Compute the group utility represented by \( Q_i \) of each solution. The \( v \) parameter is used to represent the weight of the strategy of "the majority of criteria". Equation 15 is used to calculate \( Q_i \).

\[
Q_i = \frac{v \cdot (S_i - S^*)}{(S^- - S^*)} + \frac{(1 - v) \cdot (R_i - R^*)}{R^- - R^*},
\]

where

\[
S^* = \min_j S_i \quad \text{and} \quad S^- = \max_j S_i \\
R^* = \min_j R_i \quad \text{and} \quad R^- = \max_j R_i \\
v = 0.5
\]

4. Sort options \( q_i \) in decreasing order according to the values \( S_i \), \( R_i \) and \( Q_i \). The results are three ranking lists.

5. Propose as a compromise solution the alternative \( q_i \), which is ranked the best by the measure \( Q(\text{minimum}) \) if the following two conditions are satisfied:

**C1.** Acceptable advantage:

\[ q_{i+1} - q_i \geq DQ, \]

where \( DQ = \frac{1}{(N - 1)} \); and \( N \) is the number of options

**C2.** Acceptable stability in decision making:
The alternative \( q_i \) must also be the best ranked by \( S \) or/and \( R \).
If one of the conditions is not satisfied, then a set of compromise solutions is proposed, which consists of:

- Alternative \( q_i \) and \( q_{i+1} \) if only condition C2 is not satisfied, or
- Alternative \( q_i, q_{i+1}, \ldots, q_n \) if condition C1 is not satisfied; and \( q^n \) is determined by the relation \( q_n - q_i < DQ \) for maximum \( n \).

In summary, the methods presented in this Section are used to compute the relative importance of multiple criteria and solutions based on an weighting strategy. They have been successfully applied to several real-world scenarios, where multiple conflicting objectives should be satisfied. Next Section describe how the quality of the selection of sensors provided by each algorithm can be evaluated. We also, present all performed experiments and the environment where they were executed.

### 3.4. Proposal of Evaluation of MCDA methods

This Section presents the research methodology used in the experiments. As a base of our study we assume the SAW algorithm used by Gao et al. [13] and compare it with other popular MCDA (TOPSIS and VIKOR) algorithms. Our evaluation approach is based on a set of sensor data that will be ranked according to an MCDA method and context properties. The desired number of sensors are retrieved from the top of the ranked list and the Pareto-optimal fronts are calculated. Figure 2 synthesizes the whole processing proposal for evaluating MCDA.

The Pareto-optimality criterion [30] is used to compare the quality of the solutions obtained by each method. It uses the dominance concept to determine when a solution is better then other. For example, given two solutions \( x \) and \( y \), \( x \) dominates \( y \) (\( x \succeq y \)) if two conditions are respected:

1. The \( x \) solution is better than \( y \) in at least one objective function;
2. The $x$ solution is at least equal to $y$ in all objective functions;

The set of non-dominated solutions is named Pareto-optimal set, which represents the set of optimal available solutions for the problem. The Pareto fronts is the set of values of the objective functions of the Pareto-optimal solutions set. The solutions that are dominated only for the Pareto-optimal solutions are located in the second Pareto front. The number of Pareto fronts that are used in an experiment are directly proportional to the number of non-dominated solution. In this sense, our evaluation process will consider the number of used sensors in the Pareto-optimal set and the number of Pareto fronts used by each MCDA solution. The Pareto fronts are computed through the fast-non-dominated-sort algorithm described by Deb et al. (2002) [31].

We considered two metrics to evaluate the MCDA methods: i) the number of fronts, which indicates the MCDA method with more non-dominated solutions and; ii) the Overall non-dominated vector generation ratio (ONVGR) [32] metric which shows the number of optimal solutions in the Pareto front as a proportion of the number of solutions proposed by the MCDA methods in each front. As closer the ONVGR value is to one better is the solution proposed in that front.

The test environment is composed by one physical machine. Table I describe the hardware and software specification used to perform the experiments.

<table>
<thead>
<tr>
<th>Hardware/Software</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processador</td>
<td>AMD Processor Vishera 4.2 Ghz</td>
</tr>
<tr>
<td>Memory</td>
<td>32 GB RAM DDR3 Corsair Vengeance</td>
</tr>
<tr>
<td>Hard Disk</td>
<td>HD 2TB Seagate Sata III 7200RPM</td>
</tr>
<tr>
<td>Operating System</td>
<td>Linux Ubuntu Server 14.04 64 Bits LTS</td>
</tr>
<tr>
<td>Java</td>
<td>JDK 1.7</td>
</tr>
<tr>
<td>Database</td>
<td>MongoDB 3.0</td>
</tr>
</tbody>
</table>

The experimental methodology was based on four factors: i) the number of sensors descriptions, ii) the MCDA method, iii) the number of selected sensors and iv) the number of context properties required. In this context, the term context properties will be used to refer to the analysed sensor criteria. Table II shows the used experimental factors and levels, where the combination of the levels of each factor gives a total of 45 experiments. We assume that sensor descriptions such as sensor capabilities and measurements (e.g. frequency and power consumption) are based on the 4027A Series from Bird Technologies∗. Similarly, we assume that context data related to each sensor are

retrieved from OpenWeatherMap\textsuperscript{†} and their current properties values used in this experiment (e.g. battery, price, drift and response time) are assumed to be retrieved by software systems that manage such data and are available to be used.

The criteria and objectives functions used to maximize \((\max(c_j))\) or minimize \((\min(c_j))\) the criteria follow this order: \(\max(\text{battery}), \min(\text{price}), \min(\text{drift}), \max(\text{frequency}), \min(\text{energy consumption}), \min(\text{response time})\).

4. EVALUATION RESULTS AND LESSONS LEARNED

In this Section, we present the gathered data of the performed experiments. In order to make the data visualisation and their meaning easier, we will present the results of each method regarding the number of context properties.

4.1. Evaluation Results

We will analyse the results regarding the SAW, TOPSIS and VIKOR methods. Two graphics represent the number of used fronts and the ONGVR metric. To represent the number of used front the graphic have two ordinate axis. The abscissa axis has the indexes of the Pareto fronts from the first front to the last one. The left ordinate axis presents the number of solutions retrieved by each method (different colours lines) from each Pareto front. The right ordinate axis corresponds to number of Pareto fronts needed to cover a given subset of sensors. To represent the ONVGR metric a graphic with one ordinate axis and one abscissa axis is used. The ordinate axis corresponds to the ONVGR value and the abscissa axis has the indexes of the Pareto fronts.

4.1.1. Selection using six context properties: Figure 3 presents the quality behaviour of the selection of 1,000 (Figure 3.a), 5,000 (Figure 3.b) and 10,000 (Figure 3.c) of available sensors considering six context properties (as defined in Section 3.4). The number of Pareto front slightly increases as the number of selected sensors is raised. Also, the number of optimal sensors available in each front increases according to the number of selected sensors. The MCDA methods concentrates the major part of the solutions in the first fronts due to a high number of conflicts between the used criteria.

Figure 4 shows the ONVGR value in each front of the selection of 1,000 (Figure 4.a), 5,000 (Figure 4.b) and 10,000 (Figure 4.c). The ratio value in the first fronts increases proportional to the number of selected sensors. On the other hand the ratio value shows a high loss of optimal sensors, as the ratio values changes from 0.2 to 0.6 in the worst and best scenarios respectively.

The MCDA methods does not use the Pareto optimality concept to select the sensors. They aim to select sensors that present a certain level of stability between the context properties values. While, the Pareto optimality solutions do not care about the stability between the context properties values but try to get the greatest number of context properties with the best possible values.

\textsuperscript{†}OpenWeatherMap - http://openweathermap.org/
Regarding the MCDA methods, the TOPSIS method presented the worst solution as it shows the lowest ratio value of the analysed MCDA methods in all scenarios. In addition, the SAW method is slightly better than VIKOR method when 1% of the sensors were desired as it uses less fronts, while the proposed solutions when 5% and 10% of the selected sensors were equivalent.

4.1.2. Selection using four and five context properties: Figures 5 and 6 presents the quality behaviour of the selection of 1,000 (Figure 5.a and 6.a), 5,000 (Figure 5.b and 6.b) and 10,000 (Figure 5.c and 6.c) of available sensors considering four and five context properties respectively. Analogous to Section 4.1.1, the number of Pareto front and the number of optimal solutions increases proportional to the number of selected sensors. For four and five context properties the number of Pareto fronts is twice as the results presented in Section 4.1.1 and are not so different, it varies from 6 to 16.

Figure 5. Pareto fronts for four context properties

Figure 6. Pareto fronts for five context properties

Figure 7 and 8 show the ONVGR value in each front of the selection of 1,000 (Figure 7.a and 8.a), 5,000 (Figure 7.b and 8.b) and 10,000 (Figure 7.c and 8.c). Although the number of solutions found by each method and the number of Pareto solutions are different from Section 4.1.1, the ratio between the number of selected sensors and the number of Pareto solutions are practically the
same as presented in Figure 4. They also show a low ratio value that changes approximately from 0.2 to 0.6 in the best and worst scenarios respectively.

Considering the MCDA methods, the solution proposed by the SAW method is again slightly better than the solution proposed by the VIKOR method when 1% of the sensors were desired; while the proposed solutions when 5% and 10% of the selected sensors were equivalent. On the other hand, TOPSIS presents a lower quality solutions as it shows a minor number of sensors in the top first fronts.

4.1.3. Selection using three context properties: Figure 9 presents the quality behaviour of the selection of 1,000 (Figure 9.a), 5,000(Figure 9.b) and 10,000 (Figure 9.c) of available sensors considering three context properties. As seen in Section 4.1.2, the number of Pareto front increases proportional to the number of selected sensors. This observation are justified because with less context properties we also reduce the number of context properties conflicts, the number of Pareto optimal solutions per front and the number of solutions found per front, which increases the probability for finding solutions with a higher level of stability.

Figure 10 shows the ONVGR value in each front of the selection of 1,000 (Figure 10.a), 5,000(Figure 10.b) and 10,000 (Figure 10.c). The ratio value in the first fronts are slightly higher
than the ratio values presented in Section 4.1.2 due to the reduction of the number of criteria. In this sense, the ratio values changes from approximately 0.4 to 0.8 in the worst and best scenarios respectively.

Moreover, when the MCDA methods are analysed the quality of the solution proposed by the SAW method was slighter better than the quality of the solution proposed by VIKOR method when 1%, 5% and 10% of the available sensors were selected as the SAW solution uses less Pareto fronts. Similar to Section 4.1.2, the TOPSIS method presented the solution with low quality as it had less solutions than the SAW and VIKOR methods in the top first fronts.

4.1.4. Selection using two context properties: Figure 11 presents the quality behaviour of the selection of 1,000 (Figure 11.a), 5,000(Figure 11.b) and 10,000 (Figure 11.c) of available sensors considering two context properties. Analogous to Section 9, the number of Pareto fronts and the number of optimal solutions increases directly proportional to the number of selected sensors.

Figure 12 shows the ONVGR value in each front of the selection of 1,000 (Figure 12.a), 5,000 (Figure 12.b) and 10,000 (Figure 12.c). Due to the minimal number of criteria conflicts, the ratio value tends to get closer to one as it less optimal solutions are available. In this sense, the ratio
changes from approximately 0.8 to 1 in the worst and best scenarios respectively, which shows that all optimal solutions are selected.

Furthermore, the SAW method presented again the solution with better quality independently of the selected sensors numbers as it presented a high ONVGR value and uses less fronts than VIKOR and TOPSIS. The solution presented by VIKOR was quite similar to the solution presented by the SAW method, but it solution uses more fronts than SAW. The TOPSIS method presented the poorest solution, as it shows a higher number of fronts and a low ONGVG value in the top first fronts.

4.2. Lessons Learned

In this section, we have compared the behaviour and quality of different MCDA methods for sensor search and selection. Firstly, it is important to highlight the number of optimal solutions available in each scenario. As expected, the number of optimal solutions increases proportional to the number of fronts. It occurs due to the non-dominated solution concept used to compute the optimal solutions set in each front. In this sense, the number of optimal solutions is not influenced by the number of selected sensors.

On the other hand, the number of selected sensor affects the number of optimal solutions that are founded by the MCDA algorithms. The influence of the number of selected sensors can be justified, because it increases the chances of the MCDA find the optimal sensors set. In all scenarios, the ONVGR metric clearly shows the significant increase of the ratio between the number of optimal sensors and the number of sensors found by each MCDA algorithm when more sensors are selected.

The context properties also influence the number of optimal solutions obtained by each MCDA algorithm. The number of context properties is directly proportional to number of optimal solutions available in each front. It is because as more context properties are used the number of conflicts between the criteria increases and consequently the number of non-dominated solutions increases. In other words, we reduce the chances to find a small set of solutions which present the best trade-off between the analysed context-properties.

Also, the ONVGR metric allows to compare how the number of context properties influence the number of optimal selected sensors. Although the number of selected sensors in each front is different for six, five or four context properties the ONVGR value is practically the same for all and indicates that a low number of optimal sensors is founded in each one. When three or two context properties are used, the ONVGR value is higher for all scenarios and consequently a higher number of optimal sensors is founded when less context properties is used.

Regarding the analysed MCDA methods it is possible to observe that for all analysed scenarios the SAW method, which uses regular arithmetical operations of multiplication and addition to rank the options, presented at least an equal number of fronts and the ONVGR value than TOPSIS and VIKOR method. The VIKOR method, which apply the compromise programming concept providing a maximum group utility for the majority and a minimum of an individual regret for the opponent, presented a solution pretty closer to the proposed solution by SAW algorithm but in some scenarios it solution has more fronts. Finally, the TOPSIS method which ranks the solutions according to the distance to the ideal solution and the greatest distance from the negative-ideal solution without consider the relative importance of these distances, presented the poorest solution as in the major part of the scenarios it uses more fronts and presented a low ONVGR value than SAW and VIKOR methods.

5. RELATED WORK

Today there are several approaches that enable the sensor management. Perera et al. [33] and Römer et al. [34] present surveys that describes several techniques, methods, models, features, systems, applications, and middleware solutions related to the IoT context. These surveys shows that the algorithms used to perform the sensor search and selection can be splitted in two groups: prediction models and keyword or context information. In this Section we present the main work related to each group.

Elahi et al. [35] presents a primitive called sensor ranking to perform the sensor search in an efficient way. The main idea of sensor ranking primitive is to explore the periodicity presented by
the sensor in some cases using prediction models that rank the sensors according to the probability to meet a user query. The Single-Period and the Multi-Period predictions models are used in this paper and the gathered data allow to observe a performance improvement to select the sensors. Ostermaier et al. [36] present a search engine for the Web of Things called Dyser to conduct searches in scalable environments with highly dynamic content. Dyser is able to collect and store data and information from sensors to allow search based on metadata. It also extends the work presented by Elahi et al. [35] using the Aggregated Prediction Model. The results showed that the algorithms presented a better quality selection when compared with the random model.

Truong et al. [37] also extends the work presented by Elahi et al. [35] and propose a prediction model based on fuzzy logic named Time-Independent Prediction Model. This model is able to detect anomalies about sensor behavior using metrics of density and stability. The density metric is used to estimate the probability of a certain value belong to a specific sensor while the stability metric estimates the stability of these sensors in the past. The combination of these metrics allow to rank the sensors and check their state. Thus, the solution presented is able to reduce the necessary communication for sensor search and selection.

Carlson and Schrader [38] present a search engine named Ambient Ocean to search and select sensors using context information. The search engine uses metadata, which is stored in a global repository, to establish the sensors context and carry out the search in a more efficient and effective manner. Ambient Ocean uses multi-task similarity models based on the Weighted Slope One algorithm to select the sensors. In scenarios where the characteristics of the sensors are difficult to model, collaborative filtering techniques are employed to compute similarities between users or sensors based on information history.

Ding et al. [39] propose a hybrid search engine to IoT environments, able to perform searches using quantitative values, keywords and spatio-temporal relations. The architecture of this search engine is based on a bottom-up model with three layers, the first layer is responsible for sensing and monitor the equipment. The second layer is responsible to store the data in a distributed form. The third layer provides optimized access to data from the sensors. The search for keywords and quantitative values is optimized by a B+ tree and the search base on time-space relationships uses a R tree. This search engine allows the discovery of the objects state at run-time as the sensors sends continuous data to the storage layer, which index these data according to data-structure used.

Guinard et al. [40] propose a module for the integration architecture named SOCRADES, which aims to enable ubiquitous integration services running on embedded with other business processes devices. The proposed module is based on the model Publish/Subscribe and uses a global repository to store meta-data about the available devices. The repository works with a monitor that is responsible to update the devices states and their QoS attributes. The sensor search is made by keywords and is sorted according to the QoS attributes prioritized by the user. Kothari et al. [41] presents an architecture denominated DQS-Cloud to optimize the sensor search, provide resilience to faults and QoS degradation and also optimize system performance managing sensor data streams. The sensor search is based on keywords and considers the QoS attributes specified by users. Moreover, in order to reduce communication overhead, the authors proposes an optimization mechanism to reuse sensors flows to similar requests. The results showed that the optimization module is able to reduce the bandwidth and processing rate of the providers.

Shah et al. [11] presents a search mechanism based on Coordinate Virtual System to find process in P2P networks. A coordinate is assigned to a node representing a physical location in relation to other nodes. The sensor search uses keywords and the returned sensors are ranked according to the euclidean distance to the QoS attributes specified by the user. A qualitative approach shows that the proposed search mechanism was the only one able to perform a precision query at real time. Ruta et al. [42] proposes a framework to manage semantic notations of data streams, devices, high level events and services. The requests uses the CoAP protocol based on the RESTful architectural style, which allow to use inference to support the sensor search and their compositions. A data mining mechanism was used to retrieve the sensor search in real time to improve the sensor selection. The sensor selection is based in the Concept Covering inference followed by a ranking algorithm.
Perera et al. [12] proposes a framework named CASSARAM, which performs the sensor search and selection regarding the QoS attributes specified by the user. The selection process is divided in two phases. In the first phase, the static sensor attributes, such as manufacturer or type, are used to limit the user space search. In the second phase, the result query of the first phase is evaluated in a multi-dimensional space where each axis corresponds to a QoS user attribute. The sensors are indexed regarding the Comparative Priority-based Weighted Index and ranked according to their euclidean distance to the optimal point. The authors also proposes a heuristic named Comparative Priority-based Heuristic Filtering, which removes the sensors that are far from the ideal point prioritizing the TOP-K selection. The results shows that using up to 10,000 sensors, the framework presents a satisfactory performance with a high precision.

Gao et al. [13] proposes the Automated Complex Event Implementation System to manages different run-time data streams. The sensors and their data streams are described according to SSN ontology and stored in a repository with their QoS attributes. The system acts such as a middleware between a sensor data stream and an application. The middleware are able to perform the sensor search and select using the Simple-Additive-Weighting algorithm to find the best trade-off between the specified QoS attributes.

<table>
<thead>
<tr>
<th>Paper</th>
<th>Search Technique</th>
<th>Search Method</th>
<th>Selection Method</th>
<th>QoS</th>
</tr>
</thead>
<tbody>
<tr>
<td>[35]</td>
<td>Prediction Model</td>
<td>Sensor State</td>
<td>Single-Period and Multi-Period Prediction model</td>
<td>No</td>
</tr>
<tr>
<td>[36]</td>
<td>Prediction Model</td>
<td>Sensor State</td>
<td>Aggregated Prediction model</td>
<td>No</td>
</tr>
<tr>
<td>[37]</td>
<td>Prediction Model</td>
<td>Sensor State</td>
<td>Time-Independent Prediction Model</td>
<td>No</td>
</tr>
<tr>
<td>[38]</td>
<td>Prediction Model</td>
<td>Context Information</td>
<td>Weighted Slope One Algorithm</td>
<td>No</td>
</tr>
<tr>
<td>[39]</td>
<td>Index</td>
<td>Context Information</td>
<td>B+ tree and R tree</td>
<td>No</td>
</tr>
<tr>
<td>[40]</td>
<td>Score and Ranking</td>
<td>Keywords</td>
<td>Not specified</td>
<td>Yes</td>
</tr>
<tr>
<td>[41]</td>
<td>Score, Ranking and Similarity</td>
<td>Keywords</td>
<td>Not specified</td>
<td>Yes</td>
</tr>
<tr>
<td>[42]</td>
<td>Inference and Ranking</td>
<td>Keywords</td>
<td>Concept Covering</td>
<td>No</td>
</tr>
<tr>
<td>[11]</td>
<td>Score and Ranking</td>
<td>Keywords</td>
<td>Euclidean Distance</td>
<td>Yes</td>
</tr>
<tr>
<td>[12]</td>
<td>Score and Ranking</td>
<td>Context Information</td>
<td>Euclidean Distance and Comparative</td>
<td>Yes</td>
</tr>
<tr>
<td>[13]</td>
<td>Score and Ranking</td>
<td>Context Information</td>
<td>Priority-based Heuristic Filtering</td>
<td>Yes</td>
</tr>
</tbody>
</table>

The Table III summarizes the main characteristics of the works presented in this Section. [35], [36], [37] e [38] uses prediction models and do not consider QoS attributes to choose the sensors as they are just interesting in the sensor state. [39] also do not consider QoS attributes but worry about to offer efficient data structures to store the sensor state at run-time. On the other hand, [40] e [41] highlight the importance to select the sensors based on their QoS properties, but do not present a specific method for sensor search and selection. [11], [12] and [13] uses methods to score and ranking their sensors. [11] and [12] use the Euclidean distance of the sensor to the optimal point to score and rank while [13] applies the SAW method.

Briefly, these works present different mechanisms to perform the sensor search and selection considering QoS properties. However, they do not evaluate and compare the quality of the proposed solutions of a specific technique according to the number of desired QoS properties. Thus, to fulfill this gap we have proposed a methodology to enable the comparison of different sensor search techniques from the quality of search perspective. In addition, a case study considering three sensor search techniques is presented to demonstrate our methodology.

6. CONCLUSION

The integration of IoTs with cloud computing composes the CoT and poses several new challenges. One of these challenges focus on the sensor search and selection field regarding a set of context properties explicit desired by an user. In this paper, we examined some multi objective decision analysis methods applied in different scenarios. Specifically, we have analysed the Simple Additive Weight method (SAW), the Technique for the Order of Prioritisation by Similarity to Ideal Solution
We described a methodology that uses the Pareto optimality concept to enable comparisons between the solutions proposed by these methods. The gathered results allowed to observe the impacts of ‘the number of context properties’ and ‘the number of desired sensors’ towards the quality of the final solution based on the Pareto optimality concept.

Using the overall non-dominated vector generation ratio (ONVGR) we observed when six, five and four context-properties are used, the proportion of optimal sensors retrieved is low because of the high number of context-properties conflicts, and also is practically the same for all. In contrast, when three or two context-properties are used the proportion of optimal sensors retrieved increases significantly due to the small number of context-properties conflicts.

In addition, the ONVGR value is directly proportional to the number of selected sensors. In other words as more sensors are selected more optimal sensors are used. Regarding the analysed MCDA methods, we could observe that the SAW presented the solution with better quality as their ONVGR significantly due to the small number of context-properties conflicts.

REFERENCES
17. Fersi G. Middleware for internet of things: A study. 2015 International Conference on Distributed Computing in Sensor Systems, Institute of Electrical & Electronics Engineers (IEEE), 2015; doi:10.1109/dcss.2015.43. URL http://dx.doi.org/10.1109/DCCSS.2015.43.


