The Elimination-Selection Based Algorithm for Efficient Resource Discovery in Internet of Things Environments

Luiz H. Nunes *, †, Julio C. Estrella *, Charith Perera ‡, Stephan Reiff-Marganiec §, Alexandre C. B. Delbem *
* University of São Paulo, Institute of Mathematics and Computer Science, São Carlos-SP, Brazil
Email: {lhnunes, jcezar, acbd}@icmc.usp.br
† Federal Institute of São Paulo, Araraquara-SP, Brazil
Email: lhenriquenunes@ifsp.edu.br
‡ Newcastle University, Newcastle upon Tyne, NE1 7RU - UK
Email: charith.perera@ieee.org
§ University of Leicester, University Road, Leicester, LE1 7RH - UK
Email: srm13@le.ac.uk

***Abstract*—Every day more and more objects are connected to the Internet to sense or actuate in some environment, composing the Internet of Things. IoT platforms will play a key role, as they will be responsible for managing low-level devices and data acquisition processes, and also support the development of new applications. One of the main challenges in IoT platforms will be the search and discovery of resources in large-scale and heterogeneous environments for reuse by other applications to support their specific requirements. In this paper, we propose an elimination-selection algorithm for search and discovery of resources in IoT environments. Our case study considers a real agricultural problem to be solved by the ViSIoT tool. The results show that our approach improves the quality of the proposed solution adding a small time overhead when compared to the TOPSIS algorithm used by ViSIoT.**

***Keywords*-Internet of Things, Multiple Criteria Decision Analysis, Resource Discovery, Resource Search**

I. Introduction

The advances in embedded and sensing technology are contributing to the rapid growth in the number of smart objects connected to the Internet generating the Internet of Things (IoT) paradigm. According to Vermesan et al. [1], *"The Internet of Things could allow people and things to be connected Anytime, Anyplace, with Anything and Anyone, ideally using Any path/network and Any service"*.

Nowadays, the IoT applications span numerous domains such as smart home, smart cities, smart health, smart factories and smart transport. For example, in smart transport, real-time monitoring of parking spaces and traffic congestion are some applications that have being used to optimize driving routes and reduce traffic jams using available mobile and static resources [2]. According to a Gartner Report, in the next few years, the IoT will be part of our lives as more than half of new business processes and systems should incorporate IoT elements [3].

IoT platforms such as GSN [4], OpenIoT [5] and Xively [6] will play a key role as they will be responsible for managing low-level devices, the data acquisition process, and also support the development of new applications [7]. In this sense, one of the main challenges of the IoT platforms will be the search and discovery of resources in large-scale and heterogeneous environments to be reused by other applications regarding their specific requirements and constraints [8].

Several frameworks and middlewares such as ViSIoT [9], CASSARAM [10], Ambient Ocean [11] and CASSF [12] have been proposed for search and discovery of resources in IoT environments in a timely manner considering multiple criteria and their relative priorities. Basically, the search and discovery process of can be divided into two phases: (i) use of a static query to find the resources regarding a set of specific requirements and (ii) apply some multiple criteria decision analysis (MCDA) algorithm according to the relative priorities of each requirement to rank the available resources. These works are just concerned with the time to search and discovery resources and do not properly evaluate the quality of the proposed solution, which can affect the Quality of Experience (QoE) of a user.

In this paper we propose a novel Elimination-Selection (E-S) algorithm to be performed in the second phase of the search and discovery process which increases the quality of the proposed solution. We present a agricultural case study conducted by the Visual Search for Internet of Things (ViSIoT) [9] to compare the Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) algorithm used for search and discovery of resources with the novel E-S algorithm in terms of quality and response time.

The paper is organized as follows: Section II presents a literature review of resource discovery in IoT. Section III describes the background requirements. Section IV presents the E-S algorithm. Section V describes the considered case study. Section VI describes the methodology and configurations used to perform the experiments. Section VII discuss the gathered results. Finally, the conclusions and directions for future work

978-1-5386-4790-5/18/$31.00 ©2018 IEEE

are presented in Section VIII.

II. Related Works

Several approaches consider resource discovery for the different types of IoT context. Römer et al. [13] present a survey describing methods related to the state prediction of a certain resource [14] or aiming to optimize a particular goal like energy or communication [15], [16]. In this section, we present the related works that are concerned to provide IoT resources for third parties.

Carlson and Schrader [11] shows a search engine named Ambient Ocean to discovery and select sensors based on context data. It has a metadata repository which includes information such as resources URI, data-type, title, description, and an optional Web Application Description Language (WADL) document. After an initial query to eliminate the resources that disrespect the user constraints the Weighted Slope One algorithm is used to rank the available solutions. In scenarios where it is hard to model the devices features, Ambient Ocean applies collaborative filters techniques to compute the similarity among users and sensors using historical data.

Khodadadi et al. [17] propose the Simurgh framework aiming to simplify management of IoT services, things and humans in IoT environments. They develop a metamodel named Thing Description Document (TDD) to describe the IoT entity properties and the services offered by each entity. The discovery process consists of a syntax based search method which matches the data against the desired search keywords and is split into two phases: (i) a search in the TDD repository is performed to find entities matching the user search criteria, which can be the exact match or partial match when submitting a query; (ii) the second phase consider the first phase result set to perform another search to choose the suitable devices according to a particular task.

Abdelwahab et al. [18] propose a Cloud of Things architecture for sensing resource discovery and devices virtualization in a centralized manner. The resource discovery is based on gossip algorithms to select the smart objects according to the desired capabilities. To perform the objects selection the authors propose the randomized and asynchronous distributed virtualization (RADV) algorithm which is performed in three phases: (i) the virtual domain pruning to remove undesired objects; (ii) each device constructs the benefit matrices locally to maximize the total analysed devices and (iii) solve the assignment problems to find an optimal feasible solution. After each device runs the RADV algorithm, a cloud agent is used to select the solution with maximum benefit.

Perera et al. [10], [19] present the CASSARAM framework to perform the sensor search and selection according to user constraints. The CASSARAM performs the sensor search in two phases:(i) the framework retrieves the data regarding the point-based, proximity-based and user priorities requirements. Then, (ii) the extracted data is normalized, and the Comparative-Priority Based Heuristic Filtering is used to remove the sensors that are far from the ideal point prioritizing the TOP-K selection.

Gao et al. [20] proposes the *Automated Complex Event Implementation System* which acts as a middleware between application and sensor data streams. It uses the Semantic Sensor Network to annotate the available resources allowing to design a semantic information model to represent complex event services and perform the resource discovery and integration of sensor data. The sensor stream discovery aims to find the candidate sensor services based on sensor service descriptions and user request specifications. As the candidate services are retrieved, the Simple Additive Weighting (SAW) algorithm is used to rank the available solutions.

Nunes et al. [9] present the ViSIoT to acts as a middleware between the application and the smart object's resources. Their main contribution is a visual interface to remove the complexity of the sensor discovery in the existent middleware. To select the smart objects the TOPSIS algorithm is used to rank the sensors respecting the user constraints.

Gong et al. [12] propose the Context-aware Sensor Search Framework (CASSF) to select the appropriate sensors in large datasets efficiently. To choose the available objects, they propose the Threshold Algorithm for Sensor Information (TASI) aiming to reduce the computational cost and improve the efficiency of the selection. Also, the comparative-priority based weighted index (CPWI) is used to combine the users priorities and real sensor context property values.

Huang et al. [21] present a service mining framework to discover relationships in IoT context. An ontology represents the available services and describes their spatial-temporal aspects, environment, people, and operations. In the object discovery phase, firstly they retrieve the available objects regarding the user constraints and identify the possible relationships between them. Next, a two-step selection technique is presented to remove the uninteresting services. In the first step, the Correlation Degree filtering is used to measure the relationship strength between two services. After, the interestingness value is computed considering the availability, the domain correlation, and diversity of the available objects.

Nunes et al. [22] present a methodology to evaluate the quality of resource discovery techniques in IoT context. They used the method and synthetic data to evaluate the SAW, TOPSIS and VIsekriterijumska optimizacija i KOmpromisno Resenje (VIKOR) algorithms without considering different user priorities. The result shows that the SAW algorithm has a slightly better quality than the other algorithms. This methodology was slightly changed in Nunes et al. [7] where the user priorities can be modified, and the results showed no statistical difference between the quality of the analyzed algorithms.

Differently, for the works presented in this Section, we are not proposing a new architecture for search and discovery resources or a methodology to compare the quality of solutions. In this paper, we propose an E-S algorithm that can be applied to any system to get the best trade-off between the search and discovery resource process and the QoE offered to a user regarding the quality and time of the proposed solution.

III. Background

The Multiple-criteria decision analysis is used to support the evaluation of multiple conflicting criteria in a decision-making process. There are several algorithms such as the TOPSIS that are used for MCDA and Fast-Non-Dominated Selection for Pareto Optimality. The former provides best options given a set of priorities and constraints, whereas the latter formally choose the best trade-off among the available options. However, there is a limitation in both cases, either you want the algorithm that is faster (TOPSIS) and has no guarantees of getting the best option or get the best solution with the burden of a high complexity algorithm requiring more processing and storage resource consumption. In this section, we describe the concepts used by the E-S algorithm, which combines the best of those algorithms fast selection using TOPSIS and prominent best option by using Fast-Non-Dominated sort.

A. MCDA and TOPSIS algorithm

Multiple-criteria decision analysis is concerned to architect computational and mathematical tools to aid the subjective evaluation of multiple options according to their goals [23]. Usually, an MCDA problem is described by an $M \times N$ matrix named analysis matrix. The element q_{ij} correspond to the performance value of i option regarding the decision criteria c_j, such as represented by Equation 1 [24]. An MCDA problem can be solved in three steps: (i) the normalization of the available options; (ii) the weighting of each criterion and (iii) the decision algorithm execution [25].

$$Q = \begin{array}{c} \\ q_1 \\ q_2 \\ \\ q_m \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & c_3 & & c_n \end{array} \\ \begin{bmatrix} q_{11} & q_{12} & q_{13} & \cdots & q_{1n} \\ q_{21} & q_{22} & q_{23} & \cdots & q_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ q_{m1} & q_{m2} & q_{m3} & \cdots & q_{mn} \end{bmatrix} \end{array} \tag{1}$$

The TOPSIS algorithm aims to find the best solutions, where the best option is nearest to the optimal solution and farthest to the inferior solution [24]. The TOPSIS complexity is $O(n^2)$, where n corresponds to a number of available options. The TOPSIS algorithm can be summarized as:

1) Normalize the analysis matrix Q to Q', according to Equation 2:

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

where *N* corresponds to the number of available options.

2) Using the normalized matrix Q', compute the positive ideal point (p_{+j}) and the negative ideal point (p_{-j}) for each criterion. Equation 3 represent the formulas to compute p_{+j} and p_{-j} for a maximization problem.

$$p_{+j} = \max_i(q'_{ij}) \qquad p_{-j} = \min_i(q'_{ij}) \tag{3}$$

3) Compute the distance of each option to p_{+j} and p_{-j} represented by si_+ and si_- according to Equation 4:

$$s_{i+} = \sqrt{\sum_{j=1}^{n}(q'_{ij} - p_{+j})^2} \quad s_{i-} = \sqrt{\sum_{j=1}^{n}(q'_{ij} - p_{-j})^2} \tag{4}$$

4) Compute the relative closeness (c_{i+}) of each solution to the ideal solution regarding Equation 5:

$$c_{i+} = \frac{s_{i-}}{s_{i+} - si-} \tag{5}$$

5) Sort the options in ascending order according to the c_{i+} value.

B. Pareto Optimality and Fast-Non-Dominated sort

In multiple-objective optimization problems, it is rare the cases where a single and unique option can optimize all objectives functions. It is necessary to identify a set of options with a different trade-off between their goals to solve this kind of problem.

One of the main concept used to compare a different set of options is the Pareto optimality, which uses dominance relationships to identify the optimal solutions [26]. For example, given two solutions x and y, x dominates y ($x \succeq y$) if two conditions are respected [22]:

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 nondominated solutions is also know as Pareto front. The Fast Nondominated Sorting Approach [27] is used to sort the options in fronts regarding their nondomination levels. The algorithm complexity time is $O(mn^2)$ and the storage requirement is $O(n^2)$, where n corresponds to some available options, and m corresponds to the number of criteria. Given a set of options (S), the algorithm can be summarized as:

1) For each option p in S, compute the number of options (n_p) which dominates the option p and the set of solutions (S_p) dominated by solution p. In this step, all options in the first nondominated front (F_1) will have $n_p = 0$.
2) For each option p in F_1, visit the options in S_p and update n_p value as $n_p = n_p - 1$. The options in S_p that present $n_p = 0$ belongs to the second nondominated front (F_2).
3) While there are fronts to be computed, repeat step 2 replacing F_1 and F_2 for F_{n-1} and F_n respectively.

IV. Elimination-Selection

Although the Fast Nondominated Sorting Approach can propose the solutions with the best trade-off, their high complexity of time and storage make it difficult to execute it when a large number of options are available. On the other hand, the MCDA algorithm proposes a set of solutions demanding less computational resources but with low quality.

Thus, we propose the E-S algorithm based on TOPSIS and Fast Nondominated Sorting Approach. In Table I are present the variables used to describe our algorithm.

TABLE I
LIST OF VARIABLE USED IN THE ALGORITHM

Variable	Description
$\mathbb{Q}$	Available options
$\mathbb{N}$	Number of options to be selected
$\mathbb{SR}$	Search rate value
$\mathbb{S}$	Selected options

Algorithm 1 shows the E-S algorithm. It needs three inputs parameters, $\mathbb{Q}$, $\mathbb{N}$ and $\mathbb{SR}$ which corresponds to the set of available resource options, the number of resource options to select and the search rate to be used in the preliminary steps. The search rate parameter is included based on the studies conducted by Nunes et al. [22], [7], observing that as MCDA algorithms select more sensors the better is the quality of the provided solution.

Algorithm 1 Elimination-Selection algorithm

1: **function** E-S($\mathbb{Q}, \mathbb{N}, \mathbb{SR}$)
2: $list \leftarrow TOPSIS(\mathbb{Q})$ ▷ Rank all options according to TOPSIS algorithm
3: $\mathbb{S} \leftarrow fast\_nondominated\_sort(list[1..\mathbb{N} \times \mathbb{SR}])$
4: **return** $\mathbb{S}[1..\mathbb{N}]$ ▷ Return the $\mathbb{N}$ first options regarding their front index
5: **end function**

Firstly, the TOPSIS algorithm is used to rank and sort the $\mathbb{Q}$ set. Next, the fast nondominated sort algorithm is executed considering the $\mathbb{N} \times \mathbb{SR}$ first options. The $\mathbb{SR}$ parameter plays a key role in this step, as it is used to increase the odds of getting an optimal solution proposed by TOPSIS algorithm and minimize the time and storage complexity to perform the fast non-dominated sort algorithm. Finally, the $\mathbb{N}$ options that belong to the first fronts are returned.

V. CASE STUDY

Weather events such as high temperature, rainfalls, and relative humidity play a vital role in crop yields as it increases the crop vulnerability to diseases, pest infestations and choking weeds [28]. Nowadays, the information retrieved through IoT paradigm provide to farmers insights of their crop situation, which can be used to perform strategic decisions to prevent crop damages [29].

Precipitation, temperature, humidity, wind and sky coverage are some of the climatological factors which directly impact the productivity of crops. Precipitation probably is most important factor for crop development due to their influence on other weather variables. Also, high indices of precipitation may cause waterlogging and increase pest infestations [28]. Crop species also are affected by air temperature and humidity which usually are expressed as a range of minimum and maximum expected values and influences the growth and production rates. The sky coverage defines the proportion of clouds in the sky impacting in the global solar radiation in the crop field and changes the plant's metabolic process [30]. Finally, the wind velocity affects the distribution of seeds, polymerization, and pesticide appliance in the crop.

Based on Doblas-Reyes et al. [30] and Rosenzweig et al. [28] Table II reflects these factors and their estimated range values to start a corn crop.

TABLE II
WEATHER CONDITIONS REQUIRED TO SEED A CORN CROP

Factor	Value
Temperature (t)	59 F <t <91.4 F
Humidity (h)	h >50 %
Sky coverage (sc)	sc >50 %
Wind (w)	w <20 km/h
Precipitation (p)	0 mm <p <50 mm

VI. EVALUATION METHODOLOGY

This Section presents the research methodology used in the experiments. We adapted the evaluation method proposed by Nunes et al. [22] to compare the TOPSIS and the E-S algorithm from a quality of search and time perspective.

Figure 1 shows the workflow used in our experiments. (i) The resource repository and the user context properties are used as input for the static query phase, in which the resources that not meet the user conditions will be discarded. (ii) The resulted resource list is used as input to perform the chosen MCDA algorithm, and a ranked list is returned. (iii) The Pareto Optimal Solutions Check compute the number of optimal solutions in each Pareto front.

Two metrics are used to evaluate the quality of the proposed solutions. We considered the time in seconds (s) to propose the solution, and the Overall non-dominated vector generation ratio (ONVGR) [31]. The ONVGR shows the proportion of the number of solutions suggested by the MCDA methods by the number of optimal solutions in the Pareto front in each front. As closer the ONVGR value is to one, better is the solution proposed in that front.

The experiment environment is composed of a physical machine responsible for executing the algorithm. Table III describes the hardware used in the experiments.

TABLE III
PHYSICAL ENVIRONMENT

Hardware/Software	Specification
Processor	AMD Processor Vishera 4.2 Ghz
Memory	32 GB RAM DDR3 Corsair Vegeance
Hard Disk	HD 2TB Seagate Sata III 7200RPM
Operating System	Ubuntu Server 14.04 64 Bits LTS
Java	JDK 1.7
Database	MongoDB 3.0

The experimental methodology was based on four factors: i) the number of sensors descriptions, ii) the MCDA algorithm, iii) the number of selected sensors and iv) the number of criteria. Table IV shows the used experimental factors and levels, where the combination of the levels of each factor gives a total of 6 experiments. Each experiment was replicated one hundred times. It is important to highlight in the E-S algorithm we consider the value of $\mathbb{SR}$ as two just as a proof of concept.

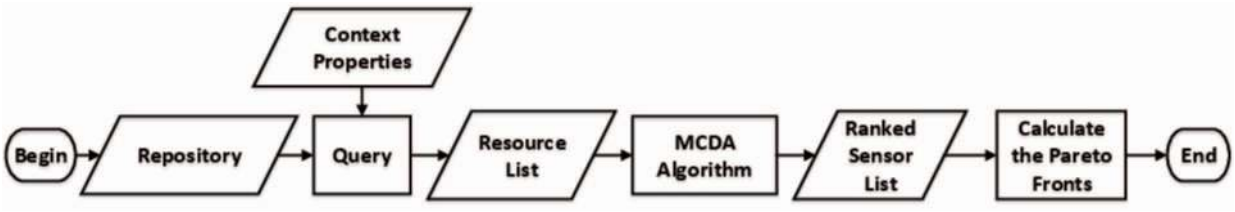


Fig. 1. Evaluation Workflow. Adapted from Nunes et al. [22]

TABLE IV
FACTORS AND LEVELS USED IN THE EXPERIMENT

Factor	Level
Number of Resource Descriptions	209,555
MCDA Method	TOPSIS and E-S
Number of Selected Sensors	2095
Number of User Constraint	2, 4 and 6

The sensor descriptions used as algorithm input are retrieved from OpenWeatherMap[1] and their current context values used in this experiment. Considering the constraints described in Table II, the objectives functions of our case study are defined in Table V. As a proof of concept, the objectives functions of the context properties where a range of values may be considered as optimal are modeled as a parabola. The point that presents the maximum value of this objective function is given for the mean value of the range. The context properties t vs h, t vs h vs sc vs w and t vs h vs sc vs w vs p vs dt are used for two, four and six user constraints respectivelly.

TABLE V
OBJECTIVES FUNCTIONS TO SEED A CORN CROP

Context property	Objective Function
Temperature	$max(-0.01777778t^2 + 10.512t - 1552.9364)$
Humidity	$\max(h)$
Sky coverage	$\max(sc)$
Wind	$\min(w)$
Precipitation	$max(0.0016p^2 + 0.08p)$
Datetime (dt)	$\max(dt)$

VII. RESULTS

In this section, we present the results of the experiments regarding time and quality of selection relating to the number of imposed conditions. Figure 2 shows the number of available resources that meet the conditions described in Table II after the static query phase. We observe when two conditions are considered, just 46,755 available resources meets the constraints which correspond to approximately 22,3% of the 209,555 resources. As more conditions are imposed the number of available resources decreases as shown for four and six criteria, which correspond to approximately 6,9% of available resources for both.

Figure 3 presents the ONVGR value according to the number of imposed conditions. The ordinate axis presents the ONVGR value and the abscissa axis the number of fronts used to propose the solution. The green line represents the optimal

[1]OpenWeatherMap - http://bulk.openweathermap.org/sample/hourly_16.json.gz

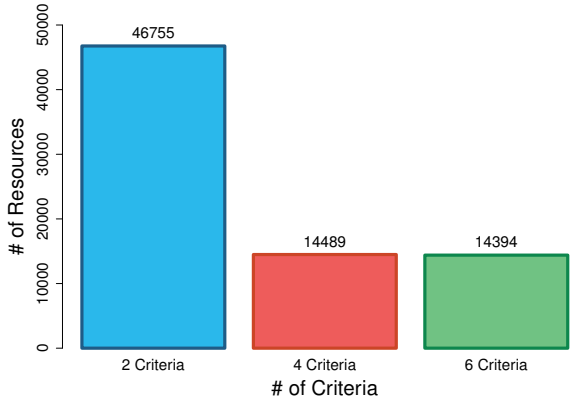
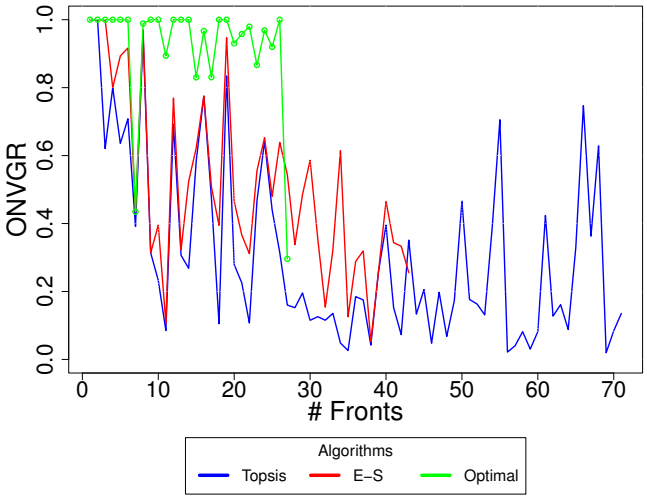


Fig. 2. Number of resources considering the conditions to seed a corn crop

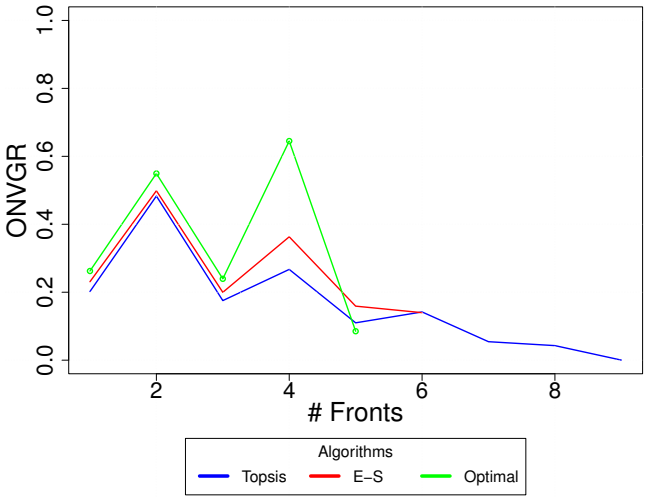
selection considering the resources retrieved from the static query phase. The optimal solution is computed considering the 209,555 resources in the repository. The blue and red lines correspond to the solutions presented by the TOPSIS and E-S algorithm respectively.

Figure 3.a the ONVGR value for the optimal selection considering two conditions are 1.0 for the first fronts and floats between 0.8 and 1.0 for the intermediary fronts, requiring a total of twenty-seven fronts. The float in the optimal selection value occurs due to the query phase, which eliminates the options that do not attend the imposed conditions. On the other hand, the ONVGR value for the TOPSIS and E-S algorithm is closer to 1.0 for fewer fronts in the beginning, and this value floats between 0.15 and 0.6 for the major part of the intermediary fronts. The high ONVGR value fluctuation for TOPSIS and E-S algorithms occurs due to the small number of solutions in each front. The E-S algorithm presented a higher ONVGR value than the TOPSIS algorithm. Consequently, it uses fewer fronts providing a better set of options.

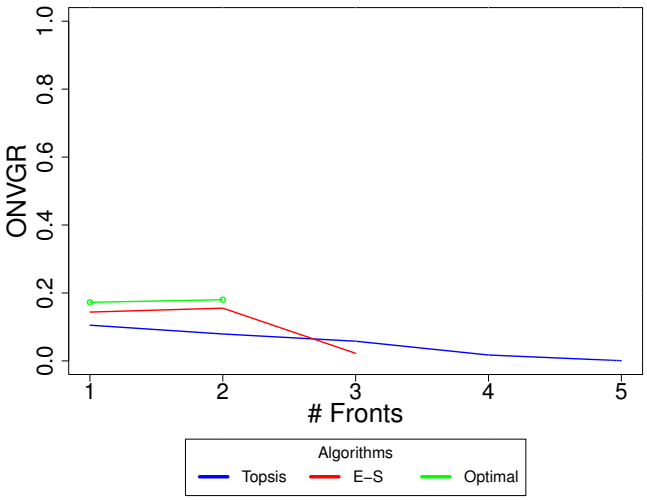
In Figure 3.b the ONVGR value for the optimal selection considering four conditions are lower than two conditions as several solutions are removed in the query phase and the number of conflicts among the conditions increases. As the number of conflicts increases, the number of the available solution in each front also increase, and consequently, the number of fronts used to propose a solution decrease. Thus, for the optimal selection, the ONVGR value floats in general between 0.2 and 0.6 using just five fronts. The E-S algorithm presents a quality of solution closer to the optimal solution, where the ONVGR value floats in general between 0.2 and 0.5 and uses six fronts providing a better solution than TOPSIS



(a) 2 Criteria



(b) 4 Criteria



(c) 6 Criteria

Fig. 3. ONVGR value according to the number of conditions

algorithm which presents an ONVGR value between 0.1 and 0.5 and uses nine fronts.

Figure 3.c the ONVGR value for the optimal selection considering six conditions are closer to 0.2 and just uses two fronts showing a behavior similar to Figure 3.b. The E-S algorithm presents an ONVGR value slightly lower than the optimal solution and uses three fronts. The TOPSIS algorithm shows a worse solution than the E-S algorithm with an ONVGR value around to 0.1 and using five fronts.

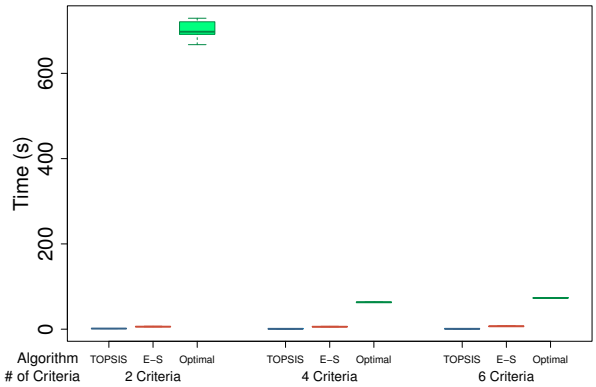


Fig. 4. Time to select the resources regarding the conditions

Figure 4 presents the time in seconds to execute the algorithms. We observe the time to compute the Optimal solution is greater than the TOPSIS and E-S approach, especially when two criteria are used as it presents the major number of available resources between all scenarios. The time to compute the optimal solution drastically decreases for four and six criteria due to the small number of sensors to be analyzed. Similar to the optimal solution, the time to perform the TOPSIS algorithm slightly decreases from two to four and six criteria as the number of resources to be analyzed is smaller. As expected, the E-S algorithm presents a higher time than TOPSIS algorithm as more steps are needed to select the sensors. The mean time overhead of the E-S algorithm about TOPSIS algorithm is closer to 4.0, 7.0 and 8.5 times for two, four and six criteria respectively.

In summary, the E-S algorithm provides a better quality of solution than the TOPSIS algorithm, independent of the number of imposed conditions in a fair time. On the other hand, while the time to compute the solution for the TOPSIS algorithm decreases when more criteria are used and fewer resources are considered, in the E-S algorithm this time increases proportionally to the number of imposed conditions due to the complexity of the fast nondominated sort algorithm. It is important to highlight the trade-off between the quality and time of a solution in the E-S algorithm is directly related to the $\mathbb{SR}$, which will determine the number of extra computations to be performed compared to TOPSIS algorithm. It should also be noted that while time is of importance, the quality is often the overriding factor: if the "thing" is being used over a longer period or provides essential data that decisions are based on, then it can be worth spending a slightly longer time to select the best possible fitting "thing".

VIII. Conclusion

The search and discovery of resources are one of the main challenges in IoT environments. Thus, several works have been proposed to solve this problem promptly, but in the majority, these proposals do not consider the need to provide the expected quality of the solution. In this paper, we proposed an E-S algorithm that can improve the quality of the solution based on the previous works, which adds a small overhead to compute the solution. We take the TOPSIS algorithm used in ViSIoT tool and the fast non-dominated sort algorithm as a base of comparison with the E-S algorithm and apply them in an agricultural case study using a real dataset. The results have shown the E-S algorithm can improve the quality of the proposed solution; although it adds a certain overhead when compared to TOPSIS algorithm. This overhead is at least ten times inferior to the fast non-dominated sort. In future work, we will look at optimizing the E-S approach, especially for the number of extra sensors that must be selected and alternatives for TOPSIS algorithm in E-S.

Acknowledgment

We thank Coordination of Improvement of Personal Higher Education (CAPES) and São Paulo Research Fundation (FAPESP), for the support of this research. We also thank ICMC-USP and LaSDPC for offering the necessary equipments for this study.

References

[1] O. Vermesan, P. Friess, P. Guillemin, S. Gusmeroli, H. Sundmaeker, A. Bassi, I. S. Jubert, M. Mazura, M. Harrison, M. Eisenhauer *et al.*, "Internet of things strategic research roadmap," *Internet of Things: Global Technological and Societal Trends*, vol. 1, pp. 9–52, 2011.

[2] O. Vermesan and P. Friess, *Internet of things-from research and innovation to market deployment*. River Publishers Aalborg, 2014.

[3] V. Woods and R. van der Meulen, "Gartner says by 2020, more than half of major new business processes and systems will incorporate some element of the internet of things," Avaliable in http://www.gartner.com/newsroom/id/3185623, January 2016, last access: 27/12/2016.

[4] K. Aberer, M. Hauswirth, and A. Salehi, "Middleware support for the internet of things," *Proceedings of 5. GI/ITG KuVS FachgespraechDrahtlose Sensornetze*, pp. 15–19, 2006.

[5] J. Soldatos, M. Serrano, and M. Hauswirth, "Convergence of utility computing with the internet-of-things," in *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, July 2012, pp. 874–879.

[6] L. Inc., "Xively is the public cloud specifically built for the internet of things," Avaliable in https://xively.com/whats_xively/, July 2014, last access: 04/02/2017.

[7] L. H. Nunes, J. C. Estrella, A. N. Delbem, C. Perera, and S. Reiff-Marganiec, "The effects of relative importance of user constraints in cloud of things resource discovery: A case study," in *Proceedings of the 9th International Conference on Utility and Cloud Computing*, ser. UCC '16. New York, NY, USA: ACM, 2016, pp. 245–250.

[8] P. Barnaghi and A. Sheth, "On searching the internet of things: Requirements and challenges," *IEEE Intelligent Systems*, vol. 31, no. 6, pp. 71–75, Nov 2016.

[9] L. H. Nunes, J. C. Estrella, L. H. V. Nakamura, R. M. D. O. Libardi, C. H. G. Ferreira, L. Jorge, C. Perera, and S. Reiff-Marganiec, "A distributed sensor data search platform for internet of things environments," *International Journal of Services Computing (IJSC)*, vol. 4, no. 1, pp. 1–12, 2016.

[10] C. Perera, A. Zaslavsky, C. Liu, M. Compton, P. Christen, and D. Georgakopoulos, "Sensor search techniques for sensing as a service architecture for the internet of things," *IEEE Sensors Journal*, vol. 14, no. 2, pp. 406–420, 2014.

[11] D. Carlson and A. Schrader, "Ambient ocean: A web search engine for context-aware smart resource discovery," in *Internet of Things (iThings), 2014 IEEE International Conference on, and Green Computing and Communications (GreenCom), IEEE and Cyber, Physical and Social Computing(CPSCom), IEEE*, Sept 2014, pp. 177–184.

[12] H. Gong, Y. Shi, and G. Li, "Context-aware sensor search framework in semantic web of things," in *2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, vol. 01, Aug 2016, pp. 94–98.

[13] K. Romer, B. Ostermaier, F. Mattern, M. Fahrmair, and W. Kellerer, "Real-time search for real-world entities: A survey," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1887–1902, nov 2010.

[14] B. Elahi, K. Romer, B. Ostermaier, M. Fahrmair, and W. Kellerer, "Sensor ranking: A primitive for efficient content-based sensor search," in *Information Processing in Sensor Networks, 2009. IPSN 2009. International Conference on*, April 2009, pp. 217–228.

[15] F. H. Bijarbooneh, W. Du, E. C. H. Ngai, X. Fu, and J. Liu, "Cloud-assisted data fusion and sensor selection for internet of things," *IEEE Internet of Things Journal*, vol. 3, no. 3, pp. 257–268, June 2016.

[16] M. Ruta, F. Scioscia, and E. D. Sciascio, "A mobile matchmaker for resource discovery in the ubiquitous semantic web," in *2015 IEEE International Conference on Mobile Services*, June 2015, pp. 336–343.

[17] F. Khodadadi, A. V. Dastjerdi, and R. Buyya, "Simurgh: A framework for effective discovery, programming, and integration of services exposed in IoT," in *2015 International Conference on Recent Advances in Internet of Things (RIoT)*. Institute of Electrical & Electronics Engineers (IEEE), apr 2015.

[18] S. Abdelwahab, B. Hamdaoui, M. Guizani, and T. Znati, "Cloud of things for sensing as a service: Sensing resource discovery and virtualization," in *2015 IEEE Global Communications Conference (GLOBECOM)*, Dec 2015, pp. 1–7.

[19] C. Perera, A. Zaslavsky, P. Christen, M. Compton, and D. Georgakopoulos, "Context-aware Sensor Search, Selection and Ranking Model for Internet of Things Middleware," in *IEEE 14th International Conference on Mobile Data Management (MDM)*, Milan, Italy, jun 2013.

[20] F. Gao, M. I. Ali, and A. Mileo, "Semantic discovery and integration of urban data streams," in *CEUR Workshop Proceedings*, vol. 1280, 2014, pp. 15–30.

[21] B. Huang, A. Bouguettaya, H. Dong, and L. Chen, *Service Mining for Internet of Things*. Cham: Springer International Publishing, 2016, pp. 566–574.

[22] L. H. Nunes, J. C. Estrella, C. Perera, S. Reiff-Marganiec, and A. C. Botazzo Delbem, "Multi-criteria iot resource discovery: a comparative analysis," *Software: Practice and Experience*, pp. n/a–n/a, 2016, spe.2469.

[23] A. Mardani, A. Jusoh, and E. K. Zavadskas, "Fuzzy multiple criteria decision-making techniques and applications two decades review from 1994 to 2014," *Expert Systems with Applications*, vol. 42, no. 8, pp. 4126 – 4148, 2015.

[24] G. Tzeng and J. Huang, *Multiple Attribute Decision Making: Methods and Applications*, ser. A Chapman & Hall book. Taylor & Francis, 2011.

[25] S. Carladous, J. M. Tacnet, J. Dezert, D. Han, and M. Batton-Hubert, "Evaluation of efficiency of torrential protective structures with new bf-topsis methods," in *2016 19th International Conference on Information Fusion (FUSION)*, July 2016, pp. 2267–2274.

[26] A. L. Jaimes, S. Z. Martınez, and C. A. C. Coello, "An introduction to multiobjective optimization techniques," 2009.

[27] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, Apr 2002.

[28] C. Rosenzweig, A. Iglesias, X. Yang, P. R. Epstein, and E. Chivian, "Climate change and extreme weather events; implications for food production, plant diseases, and pests," *Global change & human health*, vol. 2, no. 2, pp. 90–104, 2001.

[29] A. Huang, "Transforming the agricultural industry," Avaliable in https://www.ibm.com/blogs/internet-of-things/agricultural-industry/, August 2016, last access: 27/12/2016.

[30] F. Doblas-Reyes, A. Garcia, J. Hansen, L. Mariani, A. Nain, K. Ramesh, L. Rathore, and R. Venkataraman, "Weather and climate forecasts for agriculture," *Guide to agricultural, meteorological practices*, p. 57, 2003.

[31] Y. Collette and P. Siarry, *Multiobjective Optimization*. Springer Berlin Heidelberg, 2004. [Online]. Available: http://dx.doi.org/10.1007/978-3-662-08883-8