Received ; | Revised ; | Accepted
DOI: xxx/xxxx

ARTICLE TYPE

Cloud plan selection under requirements of multiple applications

Ala Arman | Sara Foresti | Giovanni Livraga | Pierangela Samarati

Computer Science Department,
Università degli Studi di Milano,
26013 Crema, Italy

Correspondence
Email: *firstname.lastname*@unimi.it

Summary

Selecting the right plan is a key issue when moving to the cloud. When multiple applications need to be outsourced at the same time, the problem becomes more complicated, and different challenges might need to be solved. In this paper, we address this problem, characterize different outsourcing scenarios, and propose a brokerage-based approach that, depending on the specific scenario, is able to compute one (or more) ranking(s) over a set of candidate plans, based on how they satisfy the specific application requirements. Our approach is able to combine application requirements to determine a plan that is suitable for all applications, possibly taking into consideration their importance (differentiating the impact that their requirements should have in the selection process), and allows different stakeholders to express such importance through linguistic variables, hence simplifying their definition and capturing the imprecision of human judgment.

KEYWORDS:
Cloud plan, Cloud plan selection, Application requirements, Application importance, Fuzzy numbers

1 | INTRODUCTION

The cloud is more and more becoming the reference paradigm for companies, public organizations, and private individuals for deploying their applications, which can thus be made available to clients over the Internet, hence reducing the economic costs and management burden as compared to a traditional in-house deployment. The cloud market features a multitude of different plans, differing in their characteristics and offered services. When an application needs to be outsourced to the cloud, selecting the right plan among those made available by providers is a key issue, and one plan is typically considered more suitable than another if its characteristics better match the specific requirements of the application itself. For instance, a mission-critical application would most likely need a highly performant cloud plan, while a secure plan would be the best choice for an application managing sensitive data. It is then important to have models that capture and express application requirements, and techniques that assess how much a candidate cloud plan can satisfy them [1, 2, 3, 4, 5, 6, 7].

Whenever moving to the cloud involves different applications, depending on the specific scenario, different challenges that require appropriate solutions can arise. For instance, different applications can have different (and possibly contrasting) needs, and a first question to be answered relates to how many plans the application owner(s) is (are) are willing to rely on: if each application can be outsourced to a specific plan, then the selection process can operate on each application singularly taken, to select the best plan for it. If, on the contrary, a single plan has to be selected for all applications, it is essential to properly combine their requirements, to select a plan that is considered globally acceptable by all applications. Some preliminary efforts have been made in this direction in [4], which however assumes that all applications to be outsourced are equally important, with the consequence that the requirements of all the applications have the same impact on the cloud plan selection process. However, the applications that need to be outsourced might differ, besides on their requirements, also on their perceived importance. In this case, it is essential to take application importance into consideration in the outsourcing process, to ensure that the requirements of more important applications (e.g., a mission-critical application that needs to be up and running 24/7) have more impact on the selection process than those of low-priority ones. The problem can be further complicated by the fact that the perceived importance of an application is clearly subjective, and when different users can have a share in the outsourcing process, they might have different opinions on how much an application (and hence its requirements) should be considered important. This can be the case, for instance, when moving to the cloud is a strategic process for a company: in this case, different stakeholders might rightfully intervene in the outsourcing process. It is then necessary to carefully consider the opinions of all the stakeholders, to correctly estimate the relative importance of the different applications.

In this paper, we build on and extend the preliminary results in [4] to address the problem of outsourcing multiple applications to the cloud. We propose a brokerage-based approach that, taking as input application requirements and a set of candidate cloud plans, evaluates the degree to which each plan satisfies the application requirements and, based on such evaluation, returns to the user a ranking (or a set thereof) of the candidate plans. Based on the computed ranking(s), the user can then select the most suitable plan (or even a combination of plans) for moving to the cloud: the higher the position of a plan in the ranking, the better the plan satisfies the application requirements. With respect to the proposal in [4], the approach we present in this paper aims at addressing different challenges that can be entailed by different outsourcing scenarios, and nicely operates on well-established techniques (e.g., multi-criteria decision making, consensus-based techniques, fuzzy numbers) that are properly combined, in a modular way, depending on the challenges that need to be solved. The contribution of this paper is multi-fold. First, we identify two main scenarios in which the general problem of outsourcing multiple applications can be instantiated, and provide a model for representing cloud plans and application requirements. We consider the basic scenario in which each application can be outsourced to a specific plan, and we extend this scenario towards the selection of single plan for the entire set of applications to be outsourced. We then analyze situations in which applications can have different importance and such importance should be reflected when ranking plans. We study the case in which the importance of the different applications is identified by a set of stakeholders, who rate each application according to their perception. In this respect, we account for the fact that the opinions of different stakeholders might have different relevance (e.g., to reflect their share in the outsourcing process), making some of them more compelling than other ones. We let stakeholders specify their opinions with linguistic labels, to properly take into consideration the imprecision and vagueness naturally implied by human reasoning.

The remainder of this paper is organized as follows. Section 2 discusses the problem addressed and identifies two possible outsourcing scenarios. Section 3 presents our model for representing cloud plans and application requirements. Section 4 illustrates a solution for evaluating cloud plans for each application to be outsourced. Section 5 extends the solution in Section 4 to a global evaluation of the cloud plans for the entire set of applications, by possibly taking their importance into consideration. Section 6 discusses how the importance of an application can be estimated, based on linguistic/fuzzy opinions of a set of stakeholders. Section 7 discusses related work. Finally, Section 8 concludes the paper.

2 | REFERENCE SCENARIO AND PROBLEM STATEMENT

The reference scenario of this paper is characterized by a user (or a set thereof) who manages a set A of applications, and who wishes to outsource these applications to the cloud. In doing so, it is essential to take the specific requirements of the applications to be outsourced into consideration in cloud plan selection: a mission/business critical application, for instance, would need a performant plan with excellent levels of guaranteed availability, while an application treating sensitive information would require a plan that enforces strong security mechanisms. Since the plans in the cloud market differ from one another in terms of their characteristics (e.g., guaranteed availability, security mechanisms, performance), given a set P of candidate plans, our problem requires to evaluate the application requirements with respect to the plans in P, to the aim of assessing how much their characteristics 'match' the requirements of the applications in A (and hence, how much such plans are suitable for outsourcing).

To address the problem illustrated above, in this paper we propose a brokerage-based approach for ranking a set P of cloud plans, based on how they satisfy the requirements of a set A of applications. The broker, acting as a proxy between the user and the candidate plans, takes as input the applications' requirements and the plans' characteristics and, after matching requirements to characteristics, returns to the user a ranking (or a set thereof, as will be explained in the following) of the plans in P, where higher positions correspond to better satisfaction of the requirements.

Depending on the willingness of the user to manage multiple contracts with different providers, we can have two different instantiations of the general cloud plan selection problem (resulting in two different outsourcing scenarios), differing in the

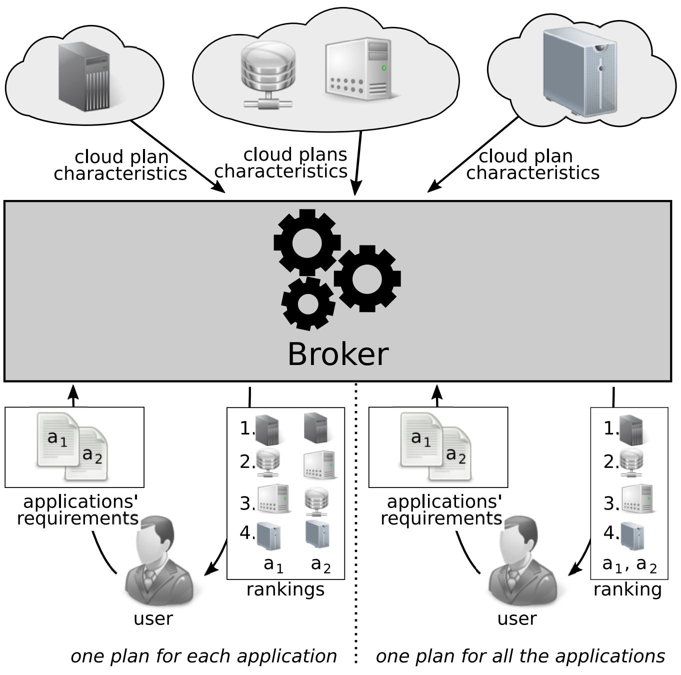

cloud plan characteristics
cloud plans characteristics
cloud plan characteristics
Broker
applications' requirements
1. 2. 3. 4.
a₁ a₂
rankings
user
applications' requirements
a₁, a₂
ranking
user
one plan for each application
one plan for all the applications

FIGURE 1 Reference scenarios

number of rankings that are to be returned to the user. In particular, if the user is willing to select one plan (i.e., the best possible solution) for each application in A, our problem requires to consider the requirements of each application singularly taken, and to produce a (possibly different) ranking of the plans in P for each application. On the contrary, if the user is willing to rely on a single plan (e.g., a plan that is highly acceptable by all applications globally considered), the different application requirements should be balanced, and our problem requires to produce a single ranking of the plans in P. In this paper, we consider both the possibilities, and hence we consider the following two scenarios, graphically illustrated in Figure 1.

- *One plan for each application* (Section 4). This is the simplest scenario, in which the applications in A can be considered independently from one another. Given an application $a \in A$ and the set P of candidate plans, the goal is to compute a ranking among the plans in P based on how such plans satisfy the requirements of the single application. Clearly, if two applications $a_i, a_j \in A$ have different requirements, the rankings computed for a_i will be different from that computed for a_j. The left-hand side of Figure 1 illustrates this scenario, where two rankings are returned to the user for two applications that are moved to the cloud.
- *One plan for all applications* (Sections 5 and 6). While the first scenario is undoubtedly natural and can model many real-world scenarios (e.g., when the complete satisfaction of the requirements of an application a is mandatory for a to be outsourced), it inevitably causes a higher management overhead at the user side (e.g., due to the possible need to manage different contracts, one for each selected provider/plan). In situations where this is not acceptable, the user might be interested in relying on a single plan for outsourcing all the applications in A. Given the entire set A of applications and the set P of candidate plans, the goal is to compute a ranking among the plans in P based on how such plans satisfy the requirements of *all* applications in A. Indeed, since the requirements' applications can be different (and possibly contrasting), this approach requires to balance the satisfaction of the requirements of the different applications. The right-hand side of Figure 1 illustrates this scenario, where only one ranking is returned to the user outsourcing two applications.

Both these scenarios have advantages and disadvantages. While the first ensures that all applications be outsourced to the best possible plan, it can require the user to rely on and interact with multiple providers, with a possible management overhead. The second scenario indeed reduces such overhead, at the price however of accommodating a trade-off on the satisfaction of the requirements of the different applications, which can be further complicated whenever some applications in A are considered more important than other ones. If this is the case, the requirements of more important applications should take precedence over those of less important applications in the computation of the ranking. For instance, the requirements of business/mission

	R_1	R_2	R_3	R_4
Availability	0.40	0.50	0.90	0.40
Performance	0.50	0.60	0.97	0.30
Security	0.60	0.70	0.80	0.20
Costs	0.20	0.40	0.50	0.60
...	...	...	...	...
Backup	0.40	0.60	0.30	0.30
StorageSpace	0.50	0.30	0.40	0.30
MobileSupport	0.60	0.80	0.30	0.40

FIGURE 2 An example of rating vectors $R_1, \ldots, R_4$ for four plans $p_1, \ldots, p_4$

critical applications should be reflected more in the plan ranking than those of less important applications. Taking application importance into consideration is a difficult task in itself since, being typically evaluated by human beings, the definition of an importance level is intrinsically imprecise (as a matter of fact, human beings tend to reason with natural language, with estimations such as 'important', 'not very important', and so on) and, whenever there are multiple stakeholders involved in the decision process, can be subjective (meaning that different stakeholders might have different perceptions on the importance of an application).

As overviewed in the discussion above, our problem entails different aspects and challenges that should be carefully taken into account to ensure a fair selection process. In the following sections, we will present the working of our approach. For the sake of readability, the remainder of this paper is organized according to the characterization of the scenarios illustrated above. After illustrating our model for representing plans and requirements in Section 3, in Section 4 we will address the 'one plan for each application' scenario, discussing how it is possible to rank a set of plans for a single application. We will then show, in Sections 5 and 6, how multiple rankings for a set of applications can be reconciled to a single ranking, possibly taking application importance into consideration. This organization nicely reflects the modular nature of our solutions, where the approach for a simpler problem is used as a building block for a more complex one.

3 | MODELING CLOUD PLANS AND REQUIREMENTS

In this paper, we are concerned with the problem of outsourcing a set $A = \{a_1, \ldots, a_n\}$ of applications and, to this end, of ranking a set $P = \{p_1, \ldots, p_m\}$ of candidate cloud plans, based on how these plans satisfy the specific applications' requirements (Section 2). Since plans differ from one another in terms of their characteristics, which make one plan more suitable than another for an application (e.g., a plan that does not offer strong security mechanism would not likely be a good fit for an application managing sensitive data), we represent plans and requirements building on such characteristics, as illustrated in this section.

We consider the characteristics of candidate plans to be defined with respect to a set $C = \{c_1, \ldots, c_l\}$ of criteria that are considered of interest for the set A of applications to be outsourced. For the sake of generality, we do not restrict the set of criteria to assume pre-defined values, and we assume that C is derived from a common ontology shared with the cloud providers [1]. For instance, C can include criteria such as the guaranteed availability, the charged costs, and the security guaranteed by the provider. Since plans in P differ in the characteristics of the offered services, each criterion $c_i \in C$ is rated according to the degree to which it is "satisfied" by a plan $p_j \in P$. Intuitively, this degree expresses how much the services offered by p_j are close to an ideal scenario that maximizes the satisfaction of c_i (e.g., a cloud provider offering its services for free would have the maximum rating for the *Cost* criterion). Without loss of generality, we express such rating with a value between 0 and 1, where higher ratings represent better satisfaction of the criterion. Each plan p_j is then associated with a *rating vector*, formally defined as follows.

Definition 1 (Rating vector)**.** Given a set P of candidate plans, a plan $p_j \in P$, and a set C of criteria, the *rating vector* of p_j is a vector $R_j[1, \ldots, |C|]$, where $R_j[i]$ represents the rating of p_j for criterion c_i, and $0 \leq R_j[i] \leq 1$.

For instance, a plan p_j providing more than 10 synchronized replicas, sophisticated authentication mechanisms and encryption algorithms, high CPU rates and network bandwidth, but applying expensive price lists, will have a high rating for security, availability, and performance, and a low rating for cost. Figure 2 illustrates an example of four rating vectors $R_1, \ldots, R_4$ for plans $p_1, \ldots, p_4$ respectively, over different criteria. For instance, p_1 is rated lower for criterion *Costs* ($R_1[\mathit{Costs}] = 0.20$) than for criterion *Security* ($R_1[\mathit{Security}] = 0.60$). We do not impose restrictions on the entity defining the ratings for plans. Such

	W_1
Availability	0.04
Performance	0.02
Security	0.04
Costs	0.90

	W_2
Backup	0.30
StorageSpace	0.50
MobileSupport	0.20

FIGURE 3 An example of weight vectors W_1, W_2 for two applications a_1 and a_2 over different criteria

ratings could be defined, for example, by the broker (if assumed trusted, or verifiable for correctness of behaviour [8]), by a trusted party, or by the user herself. Note that the user can also check, either herself or delegating the task to an external auditor, the correctness of the ratings with respect to the plans characteristics, which are measurable or declared in SLAs.

The set C of criteria is defined considering all the requirements of all the applications in A. Indeed, not *all* criteria in C may be relevant to *all* applications in A. We denote with $C_k \subseteq C$ the set of criteria relevant to application $a_k \in A$. For instance, with reference to our running example, the set C_1 of criteria relevant to a_1 is $C_1 = \{Availability, Performance, Security, Cost\}$, and the set C_2 of criteria relevant to a_2 is $C_2 = \{Backup, StorageSpace, MobileSupport\}$. Also, considering an application a_k with its set C_k of relevant criteria, not all criteria $c_i \in C_k$ can be assumed to be equally relevant to a_k. For instance, a_1 might value *Security* more than *Availability*.

A natural way to express the requirements of an application $a_k \in A$ consists in associating a weight to each criterion in C_k. In fact, this permits to model applications with a different (and possibly disjoint) set of relevant criteria, with different relevance for different criteria. Considering an application a_k, the relevance of each criterion $c_i \in C_k$ is modeled by associating c_i with a *weight*, where higher weights model higher relevance of the criterion for a_k. Formally, the requirements of an application a_k are modeled as a *weight vector*, as follows (to enable comparison among the weights, we assume the weight vectors to be normalized).

Definition 2 (Weight vector)**.** Given a set A of applications, an application $a_k \in A$, and a set C_k of criteria relevant for a_k, the *weight vector* of a_k is a vector $W_k[1, \ldots, |C_k|]$, where $W_k[i]$ represents the weight (i.e., the relative relevance) of criterion c_i for a_k, and $\sum_{i=1}^{|C_k|} W_k[i]=1$, $k = 1, \ldots, n$.

Figure 3 illustrates two examples of weight vectors for two applications, a_1 and a_2. The weight vector $W_1 = [0.04, 0.02, 0.04, 0.90]$ of application a_1 states that *Availability* and *Security* have the same relative importance (0.04 each), which is slightly higher than the importance of *Performance* (0.02), but largely smaller than the importance of *Costs* (0.90). Like for rating vectors, we do not impose restrictions on the party in charge of defining weight vectors. Indeed, weights represent the requirements of the applications, and as such reflect the characteristics of the applications themselves. They can then be defined by the user, or by trusted stakeholders in general (i.e., entities having a share in the outsourcing process) who know the applications to be outsourced and are skilled enough to describe the requirements of applications in detail (e.g., their developers).

As illustrated in the remainder of this paper, weight vectors and rating vectors will be compared to rank applications. Their comparability is guaranteed by the fact that they are defined over criteria that are taken from a common ontology shared between users and cloud providers. More precisely, each criterion included in a weight vector for an application must also be included in the rating vectors of the cloud plans (which translates to the natural observation that plans must be rated for all criteria over which a requirement is defined).

4 | PLAN RANKING FOR AN APPLICATION

Given a set A of applications, a set P of plans, a weight vector W_k for each application $a_k \in A$ (Definition 2) modeling its requirements, and a rating vector R_j for each plan $p_j \in P$ (Definition 1) modeling its criteria ratings, we aim at producing, for each application $a_k \in A$, a ranking of the plans in P based on how the plans satisfy the requirements of a_k (i.e., on how they compare with vector W_k). To this purpose, a natural solution is to adopt traditional multi-criteria decision making (MCDM) techniques, as they effectively identify, in a pool of alternative solutions (plans in P in our scenario), the one that optimizes a set of objective functions (i.e., application requirements in our terminology). Among several existing MCDM techniques, a possible approach relies on adopting TOPSIS [9] as it showed to provide good results when applied to cloud scenarios, traditionally characterized by many alternatives compared to the number of criteria [10]. For simplicity, in the following, we

	p_1	p_2	p_3	p_4
Availability	0.40	0.50	0.90	0.40
Performance	0.50	0.60	0.97	0.30
Security	0.60	0.70	0.80	0.20
Costs	0.20	0.40	0.50	0.60

FIGURE 4 Decision matrix $\mathbb{R}_1$ for application a_1 given the rating vectors in Figure 2 and the weight vectors in Figure 3

refer our discussion to one application only (a_k), with the note that the process described is executed for each application in A. In this section, we illustrate how TOPSIS can be effectively used for computing a ranking for a single application a_k.

The key insight of TOPSIS relies on evaluating how far the possible solutions are from the so-called *ideal* and *anti-ideal* solutions: intuitively, the ideal (anti-ideal, resp.) solution is a (possibly non-existing) solution that satisfies all the criteria in the best (worst, resp.) possible way. TOPSIS then uses such distances to rank the solutions, ranking higher those that are closer to the ideal solution, and farthest from the anti-ideal one. In our framing of the problem, applying TOPSIS for computing a ranking for application a_k requires to evaluate how much each plan in P is far from an ideal plan p_k^+ and an anti-ideal plan p_k^- (note that, indeed, these plans might not belong to P), defined in terms of the satisfaction of the requirements of a_k. For each application in A, TOPSIS operates in three steps: *i)* it first computes a weighted decision matrix, based on weights and ratings (Defintions 1 and 2); *ii)* it identifies the ideal and anti-ideal plans; and *iii)* finally, it ranks the plans based on their distance from the ideal and anti-ideal ones. In the remainder of this section, we illustrate more in details the working of TOPSIS in our scenario.

- **Weighted decision matrix.** To determine the weighted decision matrix for each application a_k, TOPSIS uses a *decision matrix* $\mathbb{R}_k$, with a row for each criterion $c \in C_k$ and a column for each plan $p \in P$. Basically, the decision matrix for application a_k is composed of the rating vectors R (restricted to the criteria C_k relevant to a_k): each cell $\mathbb{R}_k[i][j]$ in the decision matrix represents the rating $R_j[i]$ assigned to plan $p_j \in P$, for criteria $c_i \in C_k$. Figure 4 illustrates the decision matrix for a_1, obtained from the rating vectors in Figure 2 restricted to the first four criteria (i.e., those relevant for a_1, Figure 3). The original TOPSIS proposal normalizes the decision matrix, to guarantee that values in different cells can be compared. Since the rating values assigned to plans are already a-dimensional values between 0 and 1, in our scenario, it is not necessary to normalize the decision matrix $\mathbb{R}_k$.
  To properly take into consideration the importance of the different criteria in C_k for the considered application a_k (i.e., its requirements), the decision matrix $\mathbb{R}_k$ is combined with vector W_k (i.e., with the weights assigned to each criterion). Each cell in the *weighted decision matrix* $\mathbb{D}_k$ for application a_k is then computed as the product $\mathbb{D}_k[i][j] = \mathbb{R}_k[i][j] \cdot W_k[i]$ of the rating obtained by plan p_j for criterion c_i, and the weight of criterion c_i for application a_k. Figure 5(a) illustrates the weighted decision matrix for application a_1 of our running example. For instance, $\mathbb{D}_1[Availability][p_1]$ is obtained as $\mathbb{R}_1[Availability][p_1]\cdot W_1[Availability]$=0.4 · 0.04=0.016. Note that the weighted decision matrix permits to identify, for each criterion c_i singularly taken, the best and the worst plan, which correspond to the highest and lowest values in the row representing c_i. As an example, the best plan w.r.t. the *Security* criterion for application a_1 is p_3, while the worst is p_4.

- **Ideal and anti-ideal solutions.** Based on the weighted decision matrix $\mathbb{D}_k$, TOPSIS is able to identify both the ideal and the anti-ideal solutions p_k^+ and p_k^- for application a_k. For the ideal solution p_k^+, the weighted rating for criterion c_i (denoted $D_k^+[i]$) is the maximum weighted rating obtained by a plan in P for c_i (i.e., $D_k^+[i]=max\{\mathbb{D}_k[i][j] : p_j \in P\}$). For instance, the ideal solution for application a_1, considering the weighted decision matrix in Figure 5(a), has weighted ratings $D_1^+=[0.036, 0.019, 0.032, 0.540]$. Similarly, for the anti-ideal solution p_k^-, the weighted rating for criterion c_i (denoted $D_k^-[i]$) is the minimum weighted rating obtained by a plan in P for c_i (i.e., $D_k^+[i]=min\{\mathbb{D}_k[i][j] : p_j \in P\}$). For instance, the anti-ideal solution for application a_1, considering the weighted decision matrix in Figure 5(a), has weighted ratings $D_1^-=[0.016, 0.006, 0.008, 0.180]$.

- **Ranking.** To produce a ranking, TOPSIS computes the Euclidean distance of each plan $p_j \in P$ from the ideal p_k^+ and anti-ideal p_k^- solution in a $|C_k|$-dimensional space. Then, it computes the relative closeness of each plan p_j to the ideal solutions as $\frac{dist_j^-}{dist_j^+ + dist_j^-}$, where $dist_j^+$ and $dist_j^-$ are the distance of p_j from p_k^+ and p_k^-, respectively (where such closeness values are maintained in a score vector S_k) as in Figure 5(b). Intuitively, the higher the closeness of a plan, the better the plan satisfies the requirements. For instance, the relative closeness values of the plans in P to p_1^+ for application a_1 considering the weighted decision matrix in Figure 5(a), is $S_1=[0.044, 0.500, 0.751, 0.914]$, as illustrated in Figure 5(b). Then, TOPSIS produces a ranking of the plans for application a_k by ordering them in decreasing order of S_k.

	Weighted decision matrix				Ideal solutions	
	p_1	p_2	p_3	p_4	p_1^+	p_1^-
Availability	0.016	0.020	0.036	0.016	0.036	0.016
Performance	0.010	0.012	0.019	0.006	0.019	0.006
Security	0.024	0.028	0.032	0.008	0.032	0.008
Costs	0.180	0.360	0.450	0.540	0.540	0.180

(a)

Distance from ideal solutions				
$dist_j^+$	0.361	0.181	0.090	0.034
$dist_j^-$	0.016	0.181	0.272	0.360
S_1	0.044	0.500	0.751	0.914

(b)

FIGURE 5 Weighted decision matrix $\mathbb{D}_1$ and ideal solution p_1^+ and anti-ideal solution p_1^- (a), and distances $dist_j^+$ and $dist_j^-$ of each plan p_j from p_1^+ and p_1^- and relative closeness S_1 of each plan to the ideal solutions for application a_1 (b)

	a_1	a_2	a_3	a_4
1°	p_4	p_1	p_4	p_1
2°	p_3	p_2	p_3	p_2
3°	p_2	p_3	p_2	p_4
4°	p_1	p_4	p_1	p_3

FIGURE 6 An example of ranking of a set $\{p_1, \dots, p_4\}$ of plans by a set $A = \{a_1, \dots, a_4\}$ of applications (best plan ranked first)

Through the consideration of ideal solutions, TOPSIS is then able to rank the candidate plans in P considering the requirements of an application a_k to be outsourced. For instance, with reference to our running example, the ratings in Figure 2, and the weights in Figure 3, the ranking of plans produced by TOPSIS for application a_1 is $\langle p_4, p_3, p_2, p_1 \rangle$.

The process we have illustrated in this section is executed for each application in A by the broker, which will then return to the user a set of $|A|$ rankings. The user can then evaluate, for each application a_k, where to rely for outsourcing, taking into consideration how much the requirements of a are satisfied by the plans. Figure 6 illustrates the rankings computed by TOPSIS for applications a_1 and a_2 in Figure 3, together with an example of the rankings for two additional applications a_3 and a_4.

5 | RECONCILING MULTIPLE RANKINGS

In the previous section, we have illustrated how it is possible to rank the candidate plans in P by considering each application in A independently from the other ones. Intuitively, due to the heterogeneity of the requirements of the applications in A on one hand, and of the cloud plans on the other, the rankings computed for the different applications in A can be quite different (see, for instance, the rankings in Figure 6). The solution needs then to be adjusted whenever all applications need to be outsourced with a single plan (second scenario in Section 2), to the aim of reconciling the different rankings produced for the different applications. Given a set of rankings, a simple solution would choose the plan that better satisfies the *majority* of the applications in A. However, such a trivial approach would cause several drawbacks and complications: for instance, a clear 'winner' might not exist (see, for instance, the rankings in Figure 6, where plan p_1 is ranked first by two applications out of four, and the same occurs for plan p_4). Also, it might leave some applications completely unsatisfied: as an extreme example, consider a situation in which all applications but one (say, a) select the same plan p_x as the first, and the same p_y as the last in the ranking, while a evaluates p_y first and p_x last. Such an approach would clearly select p_x as the best plan, thus leaving the requirement of a totally unsatisfied. To prevent similar situations, we propose to adopt a *consensus-based* approach, aiming at choosing the plan that best balances the requirements of all the applications in A, allowing thus for determining a solution providing a good trade-off in requirement satisfaction.

To this end, while noting that there are different approaches that can be applied (e.g., [11], [12]), we consider - as an example - the Borda count method [13], which operates as follows [4]:

i) each application a_k associates a score $b_k(p_j)$ with each plan p_j, computed as $|P| - x$, with x the number of plans that are ranked higher than p_j (then, the first ranked plan will be assigned $|P|$ points, and the last ranked plan will be assigned 1 point);

	$b(p)$					
	a_1	a_2	a_3	a_4	$B(p)$	
p_1	1	4	1	4	10	
p_2	2	3	2	3	10	
p_3	3	2	3	1	9	
p_4	**4**	**1**	**4**	**2**	**11**	✓

FIGURE 7 Borda scores for the plans in Figure 2 given the rankings in Figure 6

(a)

	$\lambda(a)$
a_1	0.24
a_2	0.87
a_3	0.62
a_4	0.72

(b)

	$b^\lambda(p)$					
	a_1	a_2	a_3	a_4	$B^\lambda(p)$	
p_1	**0.24**	**3.48**	**0.62**	**2.88**	**7.22**	✓
p_2	0.48	2.61	1.24	2.16	6.49	
p_3	0.72	1.74	1.86	0.72	5.04	
p_4	0.96	0.87	2.48	1.44	5.75	

FIGURE 8 An example of importance values for the applications in Figure 2 (a) and weighted Borda scores for the plans in Figure 2 (b) given the rankings in Figure 6

ii) the overall Borda score $B(p_j)$ of plan p_j is computed summing all the scores assigned to it by all the applications (i.e., $B(p_j) = \sum_{k=1}^{|A|} b_k(p_j)$).

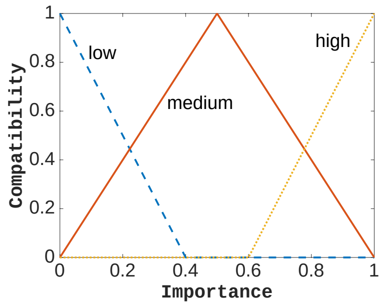
Cloud plans with higher Borda scores have more consensus among all applications and, as a result, better balance the satisfaction of their requirements. For this reason, the broker can return to the user a single ranking, computed by encompassing all the requirements of the applications, built on the overall Borda scores of the plans. As an example, Figure 7 illustrates the Borda scores assigned by each application to each plan, and the overall score of each plan: it is easy to see that the final ranking to be returned to the user orders plans as $\langle p_4, \{p_1, p_2\}, p_3\rangle$, where p_1 and p_2 are both in the second position sharing the same Borda score (and hence can be regarded as equivalent from the global point of view of requirement satisfaction).

The application of the traditional Borda count method to our scenario implicitly considers all applications as 'equivalent' (i.e., all applications have the same share in giving points to the plans). However, as discussed in Section 2, in many real-world scenarios, some applications might be considered 'more important' than other ones, with the clear consequence that the satisfaction of their requirements should be considered more than that of 'less important' applications. We address these scenarios by allowing the applications' owners to specify an *importance level* for each application. Without loss of generality, we assume such importance levels to be defined through a function $\lambda : A \to [0, 1]$ assigning to each application $a \in A$ a value $\lambda(a)$ in the interval $[0, 1]$, where $\lambda(a_x) > \lambda(a_y)$ iff a_x is more important than a_y (i.e., higher values model higher importance). Figure 8(a) illustrates an example of importance levels for the four applications in Figure 2 (we will discuss an approach for defining such values in Section 6). In this example, a_2 is more important than a_4, which is more important than a_3, which in turn is more important than a_1.

The Borda count approach illustrated at the beginning of this section can be nicely extended to take application importance into consideration, with the note that any weighted rank aggregation technique would also do. The weighted version of Borda count technique operates in a similar way as its traditional (unweighted) version, but it considers the importance $\lambda(a)$ of each application $a \in A$ in the computation of the Borda scores of the considered plans, which are then computed as $b_k^\lambda(p_j) = \lambda(a_k) \cdot b(p_j)$. The overall weighted Borda score $B^\lambda(p_j)$ of a plan p_j is then the sum of such weighted scores, that is, $B^\lambda(p_j) = \sum_{k=1}^{|A|} b_k^\lambda(p_j)$. For instance, $b_1(p_3) = 3$, since application a_1 ranks plan p_3 second over a total of 4 plans. The corresponding weighted score $b_1^\lambda(p_3)$ is equal to $\lambda(a_1) \cdot b_1(p_3) = 0.24 \cdot 3 = 0.72$. Figure 8(b) illustrates the overall weighted Borda scores for the plans in Figure 2, given the plan rankings in Figure 6 and the application importance in Figure 8(a). It is interesting to note that, while according to unweighted Borda count in Figure 7, the best plan is p_4 (and this comes with no much surprise, as it is ranked first by two applications and, in the overall computation, it is the most broadly acceptable), taking application importance into consideration modifies the result and p_1 is selected as the best plan. While considering all applications equally important the overall ranking was $\langle p_4, \{p_1, p_2\}, p_3\rangle$, the ranking computed considering different application importance levels is $\langle p_1, p_2, p_4, p_3\rangle$.

linguistic label	triangular number
low	$(0, 0, 0.4)$
medium	$(0, 0.5, 1)$
high	$(0.6, 1, 1)$

(a)



(b)

FIGURE 9 An example of linguistic labels and their triangular numbers (a), and of their graphical representation with triangular functions (b)

6 | DERIVING APPLICATION IMPORTANCE

The approach for reconciling multiple rankings considering applications' importance (Section 5) assumes that the importance of the different applications in A be expressed as a single numerical (crisp) value, in our examples taken from the range $[0, 1]$. However, in real-world scenarios there might be multiple subjects that can have a share in the definition of the importance levels (which are intrinsically subjective), and/or the assessment can be subject to a certain degree of uncertainty/fuzziness. The assumption above is therefore too simplistic. In this section, we illustrate how the importance levels adopted in Section 5 can be *derived* from subjective and imprecise opinions expressed by a set of individuals. For simplicity, we refer to such individuals as a set $S = \{s_1, \dots, s_z\}$ of *stakeholders*, that can include application developers, customers, investors, final stakeholders, and more in general any subject who can have a share in the process and wishes to contribute to it.

6.1 | Expressing subjective importance

The consideration of multiple stakeholders clearly complicates our scenario, since the importance of an application is often subjective, and each stakeholder in S may associate a different importance with the same application, depending on her point of view. For instance, the legal representative of the company will probably consider applications running financial analysis over the company's income to be more important than applications used to prepare advertisement flyers, which might be instead considered vital for the marketing office.

Selecting a single crisp value to model the importance of an application implicitly assumes that stakeholders are absolutely certain of their assessments (as crisp values leave no space for imprecision/flexibility) and agree on a common value. This, in turn, implicitly requires that all stakeholders have deep (technological) knowledge and understanding of all applications. Since we consider a possibly heterogeneous set of stakeholders, and human judgments intrinsically entail a certain degree of imprecision, we allow stakeholders to specify their opinions on the importance of applications with a certain degree of imprecision through *fuzzy numbers* [14]. A fuzzy number is a generalization of a (traditional) crisp real value, and includes a set of admissible values each with a degree of compatibility with the number itself. Among the several existing fuzzy numbers, in this paper we consider *triangular numbers* due to their intuitiveness. We note, however, that our approach can be easily adapted to use other fuzzy numbers. A triangular number is a triplet (L, M, U) with L the lowest value, M the most promising value, and U the upper value ($L \leq M \leq U$). In our scenario, assigning a triangular number (L, M, U), with $L, M, U \in \{0, \dots, 1\}$, as the importance of an application a_k models the fact that the stakeholder considers a_k important with a value in the interval $[L, U]$, with M the most promising value (i.e., it is the stakeholders' best guess). A triangular number (L, M, U) can be graphically represented as a triangular function, delimited by the values L, M, U. To help stakeholders in using them, we associate triangular numbers with *linguistic labels*, taken from an ontology including labels such as low, medium, and high and defined by experienced stakeholders or by trusted subjects. Figure 9(a) illustrates three triangular numbers and their linguistic labels, graphically illustrated as triangular functions in Figure 9(b).

For each application $a_k \in A$, each stakeholder $s_h \in S$ expresses how much she feels that application a_k is important by associating with it a linguistic label. Each application a_k is then associated with an *importance vector* $\Lambda_k[1, \dots, z]$, where $\Lambda_k[h]$ is the importance assigned by stakeholder s_h to application a_k. Figure 10(a) illustrates an example of the importance vectors for the

	s_1	s_2	s_3
a_1	low	medium	low
a_2	high	high	high
a_3	high	medium	low
a_4	high	high	low

(a)

$r(s_1) = 0.5$
$r(s_2) = 0.3$
$r(s_3) = 0.2$
(b)

FIGURE 10 An example of importance vectors for the applications in Figure 2, given by a set $S = \{s_1, s_2, s_3\}$ of stakeholders (a), and relevance of the stakeholders in S (b)

	s_1	s_2	s_3
a_1	(0, 0, 0.4)	(0, 0.5, 1)	(0, 0, 0.4)
a_2	(0.6, 1, 1)	(0.6, 1, 1)	(0.6, 1, 1)
a_3	(0.6, 1, 1)	(0, 0.5, 1)	(0, 0, 0.4)
a_4	(0.6, 1, 1)	(0.6, 1, 1)	(0, 0, 0.4)

FIGURE 11 Fuzzy importance vectors representing the importance vectors in Figure 10(a) according to the triangular numbers in Figure 9

applications in Figure 2, given by a set $S = \{s_1, s_2, s_3\}$ of three stakeholders. For instance, a_1 is considered of low importance by stakeholders s_1 and s_3, and of medium importance by stakeholder s_2.

To obtain the overall importance $\lambda(a_k)$ of application a_k, it is then necessary to properly combine the different opinions expressed by the stakeholders in Λ_k. However, not all stakeholders might be equally important in the outsourcing process. For instance, we can expect the opinion of application owners/developers, who have a clear technical understanding of the applications, to be considered more relevant than the opinion of application customers who, while having indeed interest, might not have a precise knowledge of the applications and cloud market. We account for this possibility by modeling the relevance of the opinion of each stakeholder s_h in S with a *stakeholder relevance* $r(s_h)$, which is a numerical value between 0 and 1. Such a relevance could be defined by a trusted third party overseeing the entire plan selection and outsourcing process, or collaboratively by the stakeholders themselves, if considered trustworthy. Note that we do not use fuzzy numbers or linguistic labels for relevance, because they should precisely reflect the interests at stake of each stakeholder, and hence a precise assessment (which avoids imprecision and vagueness) is needed. Figure 10(b) illustrates the relevance of the three stakeholders in our running example.

Given a set A of applications (each with its importance vector Λ) managed by a set S of stakeholders (each with her relevance $r(s)$), our problem requires then to compute an estimation of the *real* importance $\lambda(a_1), \ldots, \lambda(a_n)$ of the applications in A, considering the opinions of all stakeholders in S (i.e., the importance vectors of the applications) as well as stakeholder relevance.

6.2 | Crisp application importance

Our approach for estimating a crisp importance value for each application operates in two steps: *i) aggregate* the opinions of the different stakeholders, weighted by their relevance; and *ii) translate* such aggregated value into a crisp number in the interval [0, 1], as follows.

- **Aggregated importance.** Since aggregation of values usually operates on numerical values, to combine the importance assigned by stakeholders to applications we first translate linguistic labels into the corresponding triangular numbers. Figure 11 illustrates the importance vectors in Figure 10 where labels have been substituted with the corresponding triangular numbers according to Figure 9(a). We refer to an importance vector with labels translated into triangular numbers as a *fuzzy importance vector*, denoted $\tilde{\Lambda}$.

  Given a set of triangular numbers, there are several techniques for aggregating them to obtain a single value that is representative of the input set. Among these approaches, we adopt the *Weighted Triangular Average* (WTA), which computes the weighted average among a set of weighted triangular numbers, due to its clear and natural semantics. We note that other solutions could also be used (e.g., consider the highest value assigned to each application, or the value assigned by the most important stakeholder). Given an application a_k with fuzzy importance vector $\tilde{\Lambda}_k$, we use the WTA to compute a *weighted average* among the importance values assigned by the stakeholders to a_k, taking their relevance into account. The aggregated importance $\tilde{\lambda}(a_k)$ of a_k is then computed as:

	$\tilde{\lambda}(a)$
a_1	(0, 0.15, 0.58)
a_2	(0.6, 1, 1)
a_3	(0.3, 0.65, 0.88)
a_4	(0.48, 0.8, 0.88)

(a)

	$\lambda(a)$
a_1	0.24
a_2	0.87
a_3	0.62
a_4	0.72

(b)

FIGURE 12 Aggregated importance $\tilde{\lambda}$ (a) and real importance λ (b) of applications $a_1, \ldots, a_4$ derived from the fuzzy importance vectors in Figure 11

$$\tilde{\lambda}(a_k) = \frac{\sum_{h=1}^{|S|} \tilde{\Lambda}_k[h] \cdot r(s_h)}{\sum_{h=1}^{|S|} r(s_h)}$$

that is, weighting the triangular numbers expressing the importance given by each stakeholder (i.e., $\tilde{\Lambda}_k[h]$) by her relevance (i.e., $r(s_h)$), and averaging the result. The weighted importance $\tilde{\lambda}(a_k)$ is still a triangular number since: multiplying a triangular number (L, M, U) by a real number x produces a triangular number $(x \cdot L, x \cdot M, x \cdot U)$, and summing two triangular numbers (L_1, M_1, U_1) and (L_2, M_2, U_2) still produces a triangular number $(L_1 + L_2, M_1 + M_2, U_1 + U_2)$. Also, dividing a triangular number (L, M, U) by a real number y still produces a triangular number $(\frac{L}{y}, \frac{M}{y}, \frac{U}{y})$. Figure 12(a) illustrates the aggregated importances $\tilde{\lambda}(a)$ for the applications in our running example, given the fuzzy importance vectors in Figure 11 and the stakeholder relevance in Figure 10(b). For instance, $\tilde{\lambda}(a_1)$ is computed as:

$$\begin{aligned}\tilde{\lambda}(a_1) &= \frac{(r(s_1) \cdot \tilde{\Lambda}_1[1]) + (r(s_2) \cdot \tilde{\Lambda}_2[2]) + (r(s_3) \cdot \tilde{\Lambda}_3[3])}{r(s_1) + r(s_2) + r(s_3)} = \\ &= \frac{(0.5 \cdot (0, 0, 0.4)) + (0.3 \cdot (0, 0.5, 1)) + (0.2 \cdot (0, 0, 0.4))}{0.5 + 0.3 + 0.2} = \\ &= (0, 0.15, 0.58)\end{aligned}$$

- **Real importance.** Given a single triangular number $\tilde{\lambda}(a_k)$ expressing, in an aggregated way, the opinion of all the stakeholders on the perceived importance of an application $a_k \in A$, we transform $\tilde{\lambda}(a_k)$ into a crisp number. The consideration of fuzzy numbers nicely helps us in this task. Indeed, any defuzzification method transforms a fuzzy number into a single crisp value that is a good crisp representation of the fuzzy number. Figure 12(b) illustrates the real importances $\lambda(a_k)$ for the applications in our running example, obtained by defuzzifying the aggregated importances $\tilde{\lambda}(a_k)$ in Figure 12(a) with the well-known Center-of-Area defuzzification method (which outputs the x-coordinate of the centroid of the triangular area identified by the fuzzy number to be defuzzified). For instance, the real importance of application a_1 is $\lambda(a_1) = 0.24$, which is the centroid of the triangular area delimited by triangular number $\tilde{\lambda}(a_1) = (0, 0.15, 0.58)$.

Figure 12(b) illustrates the real importance for the four applications $a_1, \ldots, a_4$ of our running example, estimated over the linguistic importance levels and stakeholders' relevance in Figure 10. Note that these values are the values that have been used for computing the weighted ranking illustrated in Section 5.

7 | RELATED WORK

The scientific community has recently investigated different issues related to the problem of selecting and/or ranking cloud providers/services (e.g., [15]). The line of work closest to ours provides support for expressing and considering arbitrary user requirements in cloud plan selection. The work we presented in this paper builds on the preliminary findings in [4], which started investigating the problem of accommodating contrasting requirements coming from multiple applications. However, the proposal in [4] explicitly assumes that all applications to be outsourced are equally important, while here we manage application importance and present an approach for deriving such importance given a set of linguistic/fuzzy expressions. The use of fuzzy requirements and/or fuzzy logic has been investigated in cloud plan selection (e.g., [16, 17, 18, 19]). Our work differs from the approach in [16] since this latter explicitly focuses on storage plan selection. In [17], the authors propose a fuzzy-based approach for cloud service composition, while we aim at ranking a set of plans and do not compose resources from different providers. Our work differs also form the approaches in [18, 19], which also adopt MCDM and fuzzy techniques, but do not address the problem of reconciling multiple rankings taking into consideration application importance. MCDM techniques have also been investigated in [10], which however pursues the goal of comparing possible techniques to select the best cloud service based on performance measurements. While related, all these solutions differ from our work, which aims at providing a single brokerage-based solution, exploitable in different outsourcing scenarios, that incrementally addresses the different challenges that the considered scenarios can entail.

Another related line of works deals with the problem of supporting users in specifying requirements and preferences for cloud plan selection. The approach in [1] proposes a language for expressing user requirements and preferences, a formal model for reasoning on them, and different strategies for ranking acceptable services. The solution in [20] proposes a fuzzy-based brokerage system for cloud plan selection able to reason on linguistic user requirements, and to assess a candidate plan through fuzzy inferences. These works differ from ours as they do not consider the problem of reconciling contrasting requirements coming from multiple applications. The solution in [3, 21] focuses instead on the definition of an SLA between providers and customers based on customers' needs and satisfying possible dependencies enabled by them. The authors of [22] propose a solution for allocating virtual machines to cloud providers satisfying, at the same time, users' and providers' requirements, while we aim at ranking a set of candidate cloud plans.

Our work can also resemble approaches that deal with the problem of cloud provider/service assessment. The proposal in [23], while sharing with us the goal of ranking a set of cloud plans, focuses on pre-defined KPIs and does not address the issues entailed by the different outsourcing scenarios that we consider. Similarly, the approach in [24] operates on a set of pre-defined metrics. The solution in [25] deals with the problem of estimating the risk of interactions with a set of providers based on their trustworthiness and SLA transparency. In [26], the authors present a framework consisting of a broker in charge of selecting resources from different providers. All these proposals address a problem different from ours.

8 | CONCLUSIONS

We addressed the problem of selecting the most suitable cloud plan when different applications need to be outsourced. We have first characterized two outsourcing scenarios and the challenges they entail. We have then proposed a brokerage-based approach that, based on the specific outsourcing scenario and its challenges, produces a ranking (or a set thereof) computed over a set of candidate cloud plans, based on how they satisfy the requirements of different applications. The proposed approach is able to consider the fact that different applications may have different importance to the eyes of the stakeholders overseeing the outsourcing process, hence differentiating the impact that their preferences should have in the selection process. To capture the natural imprecision implied by human reasoning, our approach allows stakeholders to specify applications importance using linguistic labels. Our solution also permits to take into consideration the stakeholders' relevance, when combining their opinions on applications importance.

ACKNOWLEDGMENTS

This work was supported in part by the EC within the H2020 under grant agreement 644579 (ESCUDO-CLOUD), and within the FP7 under grant agreement 312797 (ABC4EU).

References

[1] S. De Capitani di Vimercati, S. Foresti, G. Livraga, V. Piuri, and P. Samarati, "Supporting user requirements and preferences in cloud plan selection," *IEEE Transactions on Services Computing (TSC)*, 2017, pre-print.

[2] C. Qu and R. Buyya, "A cloud trust evaluation system using hierarchical fuzzy inference system for service selection," in *Proc. of AINA 2014*, Victoria, Canada, May 2014.

[3] S. De Capitani di Vimercati, G. Livraga, and V. Piuri, "Application requirements with preferences in cloud-based information processing," in *Proc. of IEEE RTSI 2016*, Bologna, Italy, September 2016.

[4] A. Arman, S. Foresti, G. Livraga, and P. Samarati, “A consensus-based approach for selecting cloud plans,” in *Proc. of IEEE RTSI 2016*, Bologna, Italy, September 2016.

[5] S. De Capitani di Vimercati, S. Foresti, G. Livraga, and P. Samarati, “Supporting users in data outsourcing and protection in the cloud,” in *Proc. of CLOSER 2016*, Rome, Italy, April 2016.

[6] C. Alcaraz and J. Lopez, “Analysis of requirements for critical control systems,” *International Journal of Critical Infrastructure Protection*, vol. 5, no. 3, pp. 137–145, 2012.

[7] N. Tsalis, M. Theoharidou, and D. Gritzalis, “Return on security investment for cloud platforms,” in *Proc. of IEEE CloudCom 2013*, Bristol, UK, December 2013.

[8] J. Li, A. C. Squicciarini, D. Lin, S. Sundareswaran, and C. Jia, “MMB^{cloud}-tree: Authenticated index for verifiable cloud service selection,” *IEEE Transactions on Dependable and Secure Computing (TDSC)*, vol. 14, no. 2, pp. 185–198, 2017.

[9] H. Ching-Lai and K. Yoon, *Multiple attribute decision making: methods and applications*. Springer-Verlag, 1981.

[10] Z. Rehman, O. Hussain, and F. Hussain, “IaaS cloud selection using MCDM methods,” in *Proc. of IEEE ICEBE 2012*, Hangzhou, China, September 2012.

[11] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, “Rank aggregation methods for the Web,” in *Proc. of WWW 2001*, Hong Kong, China, May 2001.

[12] W. D. Cook and L. M. Seiford, “On the Borda-Kendall consensus method for priority ranking problems,” *Management Science*, vol. 28, no. 6, pp. 621–637, 1982.

[13] J. C. de Borda, *Mémoire sur les Élections au Scrutin*. Histoire de l’Academie Royale des Sciences de Paris, 1781.

[14] J. Dijkman, H. van Haeringen, and S. de Lange, “Fuzzy numbers,” *Journal of Mathematical Analysis and Applications*, vol. 92, no. 2, pp. 301–341, 1983.

[15] S. De Capitani di Vimercati, S. Foresti, G. Livraga, V. Piuri, and P. Samarati, “Supporting users in cloud plan selection,” in *From Database to Cyber Security: Essays Dedicated to Sushil Jajodia on the Occasion of his 70th Birthday*, I. Ray, I. Ray, and P. Samarati, Eds. Springer, 2018.

[16] C. Esposito, M. Ficco, F. Palmieri, and A. Castiglione, “Smart cloud storage service selection based on fuzzy logic, theory of evidence and game theory,” *IEEE Transactions on Computers (TCC)*, vol. 65, no. 8, pp. 2348–2362, 2016.

[17] A. V. Dastjerdi and R. Buyya, “Compatibility-aware cloud service composition under fuzzy preferences of users,” *IEEE Transactions on Cloud Computing (TCC)*, vol. 2, no. 1, pp. 1–13, 2014.

[18] I. Patiniotakis, S. Rizou, Y. Verginadis, and G. Mentzas, “Managing imprecise criteria in cloud service ranking with a fuzzy multi-criteria decision making method,” in *Proc. of ESOCC 2013*, Málaga, Spain, September 2013.

[19] L. Sun, J. Ma, Y. Zhang, H. Dong, and F. K. Hussain, “Cloud-FuSeR: Fuzzy ontology and MCDM based cloud service selection,” *Future Generation Computer Systems*, vol. 57, pp. 42–55, 2016.

[20] S. De Capitani di Vimercati, S. Foresti, G. Livraga, V. Piuri, and P. Samarati, “A fuzzy-based brokering service for cloud plan selection,” 2018.

[21] S. De Capitani di Vimercati, G. Livraga, V. Piuri, P. Samarati, and G. Soares, “Supporting application requirements in cloud-based IoT information processing,” in *Proc. of IoTBD 2016*, Rome, Italy, April 2016.

[22] R. Jhawar, V. Piuri, and P. Samarati, “Supporting security requirements for resource management in cloud computing,” in *Proc. of IEEE CSE 2012*, Paphos, Cyprus, December 2012.

[23] S. K. Garg, S. Versteeg, and R. Buyya, “A framework for ranking of cloud computing services,” *Future Generation Computer Systems*, vol. 29, no. 4, pp. 1012–1023, 2013.

[24] A. Li, X. Yang, S. Kandula, and M. Zhang, “CloudCmp: comparing public cloud providers,” in *Proc. of IMC 2010*, Melbourne, Australia, November 2010.

[25] N. Ghosh, S. K. Ghosh, and S. K. Das, “SelCSP: A framework to facilitate selection of cloud service providers,” *IEEE Transactions on Cloud Computing (TCC)*, vol. 3, no. 1, pp. 66–79, 2015.

[26] P. Pawluk, B. Simmons, M. Smit, M. Litoiu, and S. Mankovski, “Introducing STRATOS: A cloud broker service,” in *Proc. of IEEE CLOUD 2012*, Honolulu, HI, USA, June 2012.

AUTHOR BIOGRAPHY

Ala Arman received a M.Sc. in Software Engineering of Distributed Systems (2012) from the Royal Institute of Technology, Sweden, and a Ph.D. in Computer Science (2018) from the Università degli Studi di Milano, Italy. His main research interests are focused on large-scale distributed systems and cloud computing.

Sara Foresti is an associate professor at the Computer Science Department, Università degli Studi di Milano, Italy. Her research interests are in data security and privacy. She has published more than 90 papers in journals, conference proceedings, and books. She has been a visiting researcher at George Mason University, VA (USA). She has served as General chair and PC Chair of several conferences. She chairs the IEEE CS Italy Chapter.
http://www.di.unimi.it/foresti

Giovanni Livraga is an assistant professor at the Computer Science Department, Università degli Studi di Milano, Italy. His research interests are in data privacy and security in emerging scenarios. His PhD thesis received the ERCIM STM WG 2015 award. He has been a visiting researcher at SAP Labs, France and George Mason University, VA (USA). He has served as PC Chair and member of several conferences.
http://www.di.unimi.it/livraga

Pierangela Samarati is a professor at the Computer Science Department, Università degli Studi di Milano, Italy. Her main research interests are in data protection, security, and privacy. She has participated in/coordinated several projects. She has published more than 260 papers in journals, conference proceedings, and books. She has served as General Chair and PC Chair of several conferences. She has been named ACM Distinguished Scientist (2009) and IEEE Fellow (2012).
http://www.di.unimi.it/samarati

□