Majority-Rule-Based Web Service Selection

Karim Benouaret[1], Dimitris Sacharidis[2], Djamal Benslimane[1], and Allel Hadjali[3]

[1] Claude Bernard Lyon1 University, LIRIS, 69622 Villeurbanne, France
{karim.benouaret, djamal.benslimane}@liris.cnrs.fr
[2] IMIS, Athena Research Center, Marousi 15125, Greece
dsachar@imis.athena-innovation.gr
[3] Enssat, University of Rennes 1, IRISA, 22305 Lannion, France
allel.hadjali@enssat.fr

Abstract. In many Web service selection scenarios, the responsibility to decide which is the appropriate service is shared among multiple parties, e.g., among the department heads of a university. The standard approach is to discard services which are unanimously inappropriate, and return the rest. However, as the involved parties may have conflicting interests, it is possible that only few services are eliminated, and thus almost all discovered services need to be considered. This work addresses this shortcoming, by enforcing the *majority rule*: a service is discarded if the majority of the parties find it inappropriate. We formulate the majority-rule-based service selection problem based on the notions of dominance and skyline. Furthermore, we propose an algorithm that returns a more manageable set of services, eliminating many inappropriate ones, and is more efficient that standard skyline techniques.

1 Introduction

Several techniques for discovering Web services have been recently proposed. As the number of services and service providers proliferate, there is a large number of candidate, most likely competing, services for fulfilling a desired task. Thus, *service selection* is becoming important for helping users to identify desirable services. User preferences play a key role during the selection process [3,14,4]. However, in many practical situations, the responsibility to decide which is the appropriate service is shared among multiple parties, e.g., among the department heads of a university.

The service selection process follows two phases. In the first, given the user's preferences on service description attributes, the degrees of match between a requested and an available service (see e.g., [12,11,8]) are computed. In this work, we assume the Jaccard coefficient for matching service descriptions. If I_1, I_2 are two intervals, their Jaccard coefficient is $J(I_1, I_2) = \frac{|I_1 \cap I_2|}{|I_1 \cup I_2|}$, where $|I|$ measures the length of the interval [9].

The second phase of service selection is to identify the most interesting services w.r.t. users preferences. Most of service selection approaches focus on computing a score for each service as an aggregate of its individual matching degrees.

Various approaches for aggregating the matching degrees exist. A common direction is to assign weights over different preference attributes; e.g., [10]. However, when multiple users are involved, it would be difficult to make tradeoffs between different weights. The natural option is to use the skyline operator [5,7,13] to determine an objectively good set of services [15,1,16,2,18,17]. We refer to this set as the *unanimous service skyline*, and it contains all services which are not unanimously dominated. A service *unanimously dominates* another, if the former has higher matching degrees than the latter in all users' preferences.

Computing the unanimous service skyline frees users from assigning relative importance over different preference attributes. However, a major drawback is that, when multiple parties are involved, the number of services in the skyline becomes very large and no longer offers any interesting insights. The reason is that as the number of users and preferences increase, for any services s_i, s_j, it is more likely that s_i and s_j are incomparable, i.e., better than each other in different matching degree. It is thus crucial to further reduce the size of the service skyline.

The core of the above drawback is in the definition of dominance, which requires a unanimous verdict. To mitigate this, we choose to follow the majority rule. Informally, a service *majority-dominates* another, if the former has higher matching degrees than the latter in the *majority* of users' preferences. Then, we naturally define the *majority service skyline* as the services which are not majority-dominated.

To compute the majority service skyline, we make the observation that conventional skyline computation algorithms, with the exception of [6], cannot be adapted, due to the intransitivity of the majority-dominance relationship. Therefore, an extension of the algorithms in [6] can be used to compute the majority service skyline. However, we propose a novel algorithm for the service selection problem and show that it most cases it outperforms the extended algorithms.

The rest of the paper is structured as follows. Section 2 introduces the problem of majority service skyline and describes the majority service skyline computation algorithm. Section 3 presents our experimental study and Section 4 concludes the paper.

2 Computing the Majority Service Skyline

Section 2.1 introduces the problem, while Section 2.2 describes our algorithm.

2.1 Problem Definition

We assume a set of users $\mathcal{U} = \{u_1, u_2, \ldots, u_m\}$, and a set of discovered services $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$. We use $s_i.u_k$ to denote the matching degrees of service s_i w.r.t. user u_k. Given a user u_k, we say that service s_i *weakly dominates* s_j w.r.t. u_k, denoted as $s_i.u_k \succeq s_j.u_k$, iff s_i has better matching degrees than s_j on all specified preference attributes. A service s_i *dominates* s_j w.r.t. u_k, denoted

as $s_i.u_k \succ s_j.u_k$, iff s_i has better matching degrees than s_j on all specified preference attributes, and strictly better matching degree on at least one.

Given a set of users $\mathcal{U}$, we say that service s_i *unanimous-dominates* s_j, denoted as $s_i \succ_U s_j$, iff s_i weakly dominates s_j w.r.t. all users, i.e., $\forall u_k \in \mathcal{U}$ $s_i.u_k \succeq s_j.u_k$, and there exists one user, say u'_k, for which s_i dominates s_j, i.e., $\exists u'_k \in \mathcal{U}$ $s_i.u'_k \succ s_j.u'_k$. Given a set of discovered services $\mathcal{S}$ and a set of users $\mathcal{U}$, the *unanimous service skyline* $USS(\mathcal{S},\mathcal{U})$ comprises the set of services that are not dominated by any other.

Given a set of users $\mathcal{U}$, we say that service s_i *majority-dominates* s_j, denoted as $s_i \succ_M s_j$, iff (1) there exists a subset $\mathcal{U}' \subseteq \mathcal{U}$ containing more than half of the users such that s_i weakly dominates s_j w.r.t. all users in this subset, i.e., $|\mathcal{U}'| > \lfloor |\mathcal{U}|/2 \rfloor$ and $\forall u_k \in \mathcal{U}'$ $s_i.u_k \succeq s_j.u_k$, and (2) there exists one user, say u'_k, for which s_i dominates s_j, i.e., $\exists u'_k \in \mathcal{U}$ $s_i.u'_k \succ s_j.u'_k$. Given a set of discovered services $\mathcal{S}$ and a set of users $\mathcal{U}$, the *majority service skyline* $MSS(\mathcal{S},\mathcal{U})$ comprises the set of services that are not majority- dominated by any other.

Problem statement: Given a set of users $\mathcal{U}$ and a set of discovered services $\mathcal{S}$, compute the *majority service skyline*.

2.2 Majority Service Skyline Algorithm

In this section, we introduce the *Majority Service Skyline Algorithm* (MSA), which is based on the following properties. Note that the proofs of all lemmas and theorems appear in the full version of this paper[4].

Theorem 1. *It is possible to have a set of users $\mathcal{U}$ and a set of discovered services $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$ such that s_1 majority-dominates s_2, s_2 majority-dominates s_3, ..., s_{n-1} majority-dominates s_n and s_n majority-dominates s_1, i.e., forming a cyclic majority dominance relationship.*

The previous theorem shows that the majority dominance relationship shares the cyclic property of the k-dominance relationship introduced in [6]. Therefore, a service cannot be discarded even if it is majority-dominated because it might be needed for excluding other services. This justifies why the existing algorithms for computing the skyline are not applicable for computing the majority service skyline. However, the one scan algorithm (OSA) and two scan algorithm (TSA) of [6], can be adapted to compute the majority service skyline, by exchanging k-dominance checks for majority dominance checks. In the following, we denote as OSA and TSA the adaptations of the algorithms in [6] to computing the majority service skyline.

The MSA algorithm also takes advantage of the following observations.

Lemma 1. *If s_i unanimous-dominates s_j, then s_i majority-dominates s_j. i.e., $s_i \succ_U s_j \Rightarrow s_i \succ_M s_j$.*

Lemma 2. *If s_i unanimous-dominates s_j and s_j majority-dominates s_k, then s_i majority-dominates s_k. i.e., $s_i \succ_U s_j \wedge s_j \succ_M s_k \Rightarrow s_i \succ_M s_k$.*

[4] `http://liris.cnrs.fr/Documents/Liris-5691.pdf`

Lemma 3. *Let $f : \mathcal{S} \rightarrow \mathbb{R}^+$ be a monotone function aggregating the matching degrees of s_i for all users. If s_i unanimous-dominates s_j, then $f(s_i) > f(s_j)$. i.e., $s_i \succ s_j \Rightarrow f(s_i) > f(s_j)$.*

From Lemma 1 and Lemma 2, we can see that it is sufficient to compare each service against the unanimous skyline services to detect if it is part (or not) of the majority service skyline. This essentially reduces the number of comparisons. Specifically, *if a service s_i is unanimous-dominated, then discard it as (1) it is not part of the majority service skyline (Lemma 1), and (2) it is unnecessary for eliminating other services (Lemma 2).*

Lemma 3 also helps reduce unnecessary comparisons. In fact, to exploit this property, we sort the services in non-ascending order of the sum of their matching degrees. Then, given a service s_i, searching for services by which s_i is unanimous-dominated can be limited to the part of the service before s_i. This is the idea behind the SFS algorithm [7], which in this context we apply it for cyclic dominance relationships.

The MSA algorithm leverages the observations made above to compute efficiently the majority service skyline. Based on Lemma 1 and Lemma 2, MSA maintains two sets $\mathcal{R}$ and $\mathcal{T}$, containing respectively the set of intermediate majority skyline services and the set of intermediate unanimous skyline services that are not in $\mathcal{R}$. Thus, $\mathcal{R} \cup \mathcal{T}$ constitutes the intermediate unanimous skyline.

The MSA algorithm operates as follows. First, services in $\mathcal{S}$ are sorted in a non-ascending order of the sum of their matching degrees, and both sets $\mathcal{R}$ and $\mathcal{T}$ are initialized to empty sets. Then, the top service (i.e., the service with the maximum sum of matching degrees), say s_i, is extracted from $\mathcal{S}$. Service s_i is compared against services in $\mathcal{R} \cup \mathcal{T}$, i.e., the set of services that may unanimous-dominate s_i (as the other services cannot dominate s_i from Lemma 3). If s_i is unanimous-dominated, then it is removed from $\mathcal{S}$ as it is not part of the majority service skyline (Lemma 1) and it is unnecessary for eliminating other services (Lemma 2). Otherwise, i.e., when s_i is not unanimous- dominated by any service in $\mathcal{R} \cup \mathcal{T}$, if s_i majority-dominates any service s_j in $\mathcal{R}$ (i.e., s_j is not a service in MSS), then s_j is removed from $\mathcal{R}$ to $\mathcal{T}$, as it is a unanimous skyline service, thus useful for eliminating other services. For the same reason, if s_i is *majority-dominated* by any service in $\mathcal{R} \cup \mathcal{T}$, it is inserted into $\mathcal{T}$ as it is not part of the majority service skyline. Else, s_i is an intermediate MSS service and is thus inserted into $\mathcal{R}$. Once all services in $\mathcal{S}$ have been examined, i.e., $\mathcal{S}$ is empty, services in $\mathcal{R}$ form the majority service skyline, and $\mathcal{R}$ is returned.

3 Experimental Evaluation

In this section, we present an experimental evaluation of our approach. Our objective is to prove the *effectiveness* of the majority service skyline and the *efficiency* of the proposed algorithm. Specifically, we focus on two issues. (1) The size of the majority service skyline (denoted as MSS). To demonstrate that the majority service skyline further reduces the size of the (traditional) skyline,

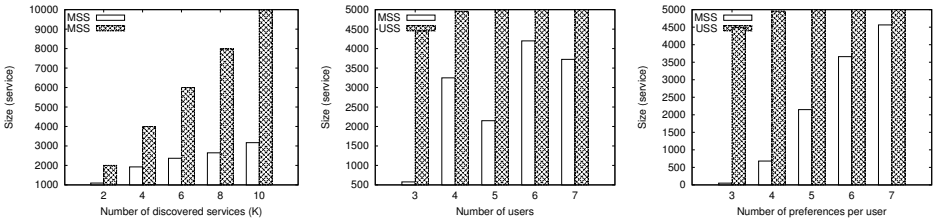
we also compute the size of the unanimous service skyline (denoted as USS) (2) The performance of our algorithm in terms of elapsed time for computing the majority service skyline. For comparison purposes, we also implemented the adaptations of OSA and TSA [6] for computing the majority service skyline.

Table 1: Parameters and Examined Values

Parameter	Symbol	Range	Default
Number of discovered services	n	$[2, 10]$K	5K
Number of users	m	$[3, 7]$	5
Number of preferences per user	d	$[3, 7]$	5

Due to the limited availability of real-world service data, we implemented a service generator that takes as input a (real-world) model service and its associated constraints, representing the requested service and the multiple users preferences, and produce a set of synthetic services, as well as their associated constraints, representing the set of discovered services. The Jaccard coefficient is used for computing the matching degrees between discovered service' constraints and users preferences. The generation of the sets of synthetic services is controlled by the parameters in Table 1, which displays the parameters under investigation, their corresponding ranges and their default values. In each experimental setup, we investigate the effect of one parameter, while setting the remaining ones to its default value.

The service generator and the algorithms, i.e., MSA, OSA and TSA were implemented in Java, and all experiments were conducted on a 2.3 GHz Intel Core i5 processor.

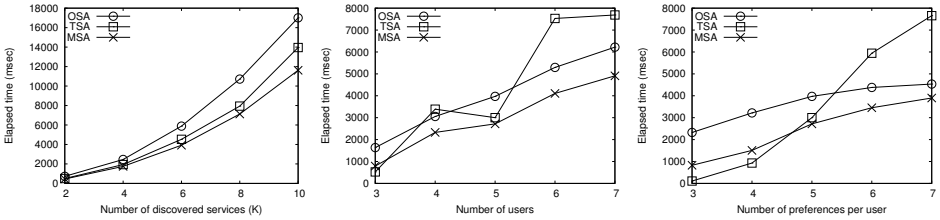


(a) Effect of n (b) Effect of m (c) Effect of d

Fig. 1: Result cardinality.

Figure 1 shows the cardinality of MSS and USS w.r.t. n, m and d. Constantly, the size of MSS is less than USS, which is almost equal to the number of discovered services, as the unanimous service skyline cannot discard all inappropriate services, while the majority service skyline includes only the most interesting ones. As shown in Figure 1a, the size of the majority service skyline increases slightly with n. This is because as n varies, it is becoming more difficult to find services which are majority-dominated. Figure 1b shows a fluctuation in the size of the majority service skyline. The fluctuation is related to the definition of the

majority dominance relationship. Indeed, we can distinguish two trends. One for the even values of m, and the second for the odd values of m; each trend increases as m increases. This is because, if we have an odd value of m, say m_o, and an even value of m, say m_e, such that $m_o = m_e + 1$, then the percentage of most of users for m_e is greater than that of m_o. For example, for $m = 4$, the percentage is $\frac{3}{4} = 0.75\%$, and for $m = 5$ the percentage is $\frac{3}{5} = 0.60\%$. When this percentage is large, a small number of services is discarded, and vice versa. As depicted in Figure 1c, the size of the majority service skyline increases significantly with d. As d increases, a service has greater probability to not be dominated in all preference attributes w.r.t. a given user.



(a) Effect of n (b) Effect of m (c) Effect of d

Fig. 2: Elapsed time.

Figure 2 investigates the runtime of OSA, TSA and MSA w.r.t. n, m and d. Overal, MSA outperforms OSA and TSA. Figure 2a shows that the execution time of the algorithms increases with n. However, MSA consistently outperforms OSA and TSA. As shown in Figure 2b, when m increases, the performance of TSA deteriorates due to the second scan performed. However, the execution time of OSA and MSA increases slightly with m. Still, MSA is better. As shown in Figure 2c, TSA is better than OSA and MSA for $d \leq 4$ since the size of the majority service skyline is small, thus a large number of services can be eliminated in the first scan. However, TSA does not scale with d as the size of the majority service skyline becomes large, thus the second scan is very time consuming. The execution time of OSA and MSA, on the other hand, increases slightly with d. Finally, observe that MSA consistently performs better than OSA.

4 Conclusion

We introduce a novel concept for the preference-based Web service selection under multiple users preferences problem based on the majority rule. This allows users to make a "democratic" decision on which services are the most appropriate. We develop a suitable algorithm for the majority-rule-based Web selection problem. Our experimental evaluation demonstrates the effectiveness of the concept and the efficiency of the algorithm.

References

1. Alrifai, M., Skoutas, D., Risse, T.: Selecting skyline services for qos-based web service composition. In: WWW. pp. 11–20 (2010)
2. Benouaret, K., Benslimane, D., Hadjali, A.: On the use of fuzzy dominance for computing service skyline based on qos. In: ICWS. pp. 540–547 (2011)
3. Benouaret, K., Benslimane, D., Hadjali, A., Barhamgi, M.: Fudocs: A web service composition system based on fuzzy dominance for preference query answering. PVLDB 4(12), 1430–1433 (2011)
4. Benouaret, K., Benslimane, D., Hadjali, A., Barhamgi, M.: Top-k web service compositions using fuzzy dominance relationship. In: IEEE SCC. pp. 144–151 (2011)
5. Börzsönyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: ICDE. pp. 421–430 (2001)
6. Chan, C.Y., Jagadish, H.V., Tan, K.L., Tung, A.K.H., Zhang, Z.: Finding k-dominant skylines in high dimensional space. In: SIGMOD Conference. pp. 503–514 (2006)
7. Chomicki, J., Godfrey, P., Gryz, J., Liang, D.: Skyline with presorting. In: ICDE. pp. 717–719 (2003)
8. Dong, X., Halevy, A.Y., Madhavan, J., Nemes, E., Zhang, J.: Simlarity search for web services. In: VLDB. pp. 372–383 (2004)
9. Duda, R.O., Hard, P.E.: Pattern Classifcation and Scene Analysis. A Wiley-Interscience Publication, New York (1973)
10. Lamparter, S., Ankolekar, A., Studer, R., Grimm, S.: Preference-based selection of highly configurable web services. In: WWW. pp. 1013–1022 (2007)
11. Li, L., Horrocks, I.: A software framework for matchmaking based on semantic web technology. In: WWW. pp. 331–339 (2003)
12. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.P.: Semantic matching of web services capabilities. In: International Semantic Web Conference. pp. 333–347 (2002)
13. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. In: SIGMOD Conference. pp. 467–478 (2003)
14. Wang, H., Xu, J., Li, P.: Incomplete preference-driven web service selection. In: IEEE SCC (1). pp. 75–82 (2008)
15. Yu, Q., Bouguettaya, A.: Computing service skyline from uncertain qows. IEEE T. Services Computing 3(1), 16–29 (2010)
16. Yu, Q., Bouguettaya, A.: Computing service skylines over sets of services. In: ICWS. pp. 481–488 (2010)
17. Yu, Q., Bouguettaya, A.: Efficient service skyline computation for composite service selection. IEEE Transactions on Knowledge and Data Engineering 99(PrePrints) (2011)
18. Yu, Q., Bouguettaya, A.: Multi-attribute optimization in service selection. World Wide Web 15(1), 1–31 (2012)