

Evaluating Dynamic Services in Bioinformatics

Maíra R. Rodrigues^{1*} and Michael Luck¹

School of Electronics and Computer Science, University of Southampton,
Southampton SO17 1BJ, UK

Abstract. In dynamic applications characterised by a variety of alternative services with the same functionality but heterogeneous results, agents requesting services must find an efficient way to select a service provider from alternatives. In this context, this paper proposes an evaluation method to analyse the outcome of dynamic service, in order to provide a guide for agents in future decision-making over alternative interaction partners. We consider the application of the evaluation method to the bioinformatics domain and present empirical results that support the need for dynamic evaluation of services in that domain.

1 Introduction

Bioinformatics is a new field of research characterised by the application of computer technology to the management and analysis of biological data (i.e., to gather, store, analyse and merge genome and protein related information) [1]. Because of the vast quantities of data being generated by several genome and protein sequencing efforts, a large variety of services have been developed to analyse such data. These services are not only heterogeneous in terms of functionalities and results, they are also distributed over the Internet, and in continuous update. Such a dynamic, distributed and heterogeneous environment imposes restrictions on the task of managing and analysing biological data and services, and points to the suitability of an agent-based approach. When an agent is engaged in this kind of environment and needs to delegate to, or request a bioinformatics service from, another agent, it is likely that it will find many alternative agents providing similar services.

In essence, there are three ways of selecting a service provider from alternatives: by random selection; by identifying the best provider based on the service or provider description given by the providers themselves; or by identifying the provider with the best outcomes over previous interactions. Random selection may allow providers offering poor services to be selected over those offering better services. Selection based on descriptions given by providers does not guarantee that providers are giving correct information or that the described properties are valid when services are performed in different contexts. The more efficient way to select is the third option, by identifying partners with good outcomes

* The first author is supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) of the Brazilian Ministry of Education.

over previous interactions. This method is based on the assumption that, when services manifest a certain regularity of behaviour, a provider that performed well in the past is likely to perform well again in a similar situation. If the aim of evaluation is to provide some *criterion for future decision-making* over alternative interaction partners, then this offers a reasonable way to proceed.

In order to determine the best outcome in this way, however, agents requesting services must perform an *evaluation* of services after they are executed and the results are received. This evaluation should reflect the satisfaction of the user with the service outcome, which can be in relation to the quality of the interface, the provider's availability to perform the service when requested, the time taken to execute, the quality of the content returned, and so on. For example, the evaluation allows a requester to identify, among the possible providers, the one with the best attributes in a similar situation, like highest quality of results and lowest time to complete the request.

Although the application of multi-agent systems to the management and analysis of bioinformatics data has already been proposed [2, 3], previous work is concerned with high-level management and integration of bioinformatics tools and data, and does not address the evaluation of individual bioinformatics services.

In response, this paper proposes an evaluation method to analyse service outcomes, in order to provide a guide for agents in future decision-making over alternative interaction partners. This evaluation is needed to determine how *satisfied* an agent is with a service it has requested and received.

The paper starts with an analysis of the issues concerning evaluation of dynamic services, followed by a description of the proposed evaluation method to be embedded in the agents' internal architecture. We introduce a scenario in the bioinformatics domain in which we evaluate services for protein identification, and end by presenting empirical results that support the need for dynamic evaluation of those services.

2 Evaluation of Dynamic Services

When evaluating a service, independent of the context or domain in which the evaluation is taking place, we must consider which characteristics may be important to analyse during evaluation, since evaluators are usually interested in several aspects of the service. For example, when evaluating the food in a restaurant, customers might take into account the quality of the ingredients, the way the food was presented, and the price. Similarly, to evaluate computing services like search engines, users must consider, for example, the time taken to complete the query, the relevance of the content returned to the user in relation to the query, and the way results were presented. The number of characteristics to be evaluated in a service varies according to the evaluator and the type of the service; that is, the more complex the service, the more aspects might be relevant to observe.

Dynamic services are considered here both as the services changing constantly through new versions and updates, and the services that, despite manifesting regular behaviour when working under similar conditions, can vary their performance depending on the parameter configurations used. An example of such dynamic services are those in the bioinformatics domain, in which the dynamism is a consequence of the great amount of information resulting from continuing genomic and proteomics research. Also, some bioinformatics services, like those related to comparison and search against gene or protein databases, vary their performance according to *both* the quality of the input data used for comparison and search, and the configuration parameters used for execution.

However, because the performance of dynamic services can change from one execution to another, evaluations have to be undertaken every time a service is executed. In addition, the evaluation should be associated with context information, to allow future analysis of service results under different contexts. More specifically, we identify four distinct issues to be addressed by an evaluation method in this context, as follows.

1. *Consistency*: the evaluation method must deliver an evaluation measure that allows comparison between evaluations, even if they were generated at different points in time. This is important when interactions between agents are repeated over time under different conditions, and when new services appear.
2. *Generality*: the evaluation method must be general enough to be applied to different types of service. Having one general method that can be applied in all situations is more advantageous than having to define specific methods for different types of services. A consequence of this generality is the need to support evaluation according to multiple attributes, since agents with different goals may be interested in evaluating different attributes of the service.
3. *Continuity*: the evaluation process must occur every time agents interact, instead of only once, since the performance of services may change from one interaction to another if different input configurations are used, if services are updated, or if new data has been published.
4. *Discriminated information*: the evaluations provided by the evaluation method must allow flexible decision-making in the future service selection process, so services can be selected either by comparing evaluations of particular attributes, or by comparing a single evaluation that combines the measures of all attributes.

2.1 Alternative Approaches

Traditional evaluation approaches calculate the evaluation of a service using scoring or utility functions, which return a quantitative evaluation for the service [4–6]. Utility functions can be calculated based on observed values only, or on the comparison of observed and expected values. In the first case, which we refer to as the *absolute evaluation* approach, the utility is derived from values

that are observed directly from service outcomes, and the evaluation of a service depends only on its performance and is not influenced by expected performance [4, 6]. In the second case, which we refer to as *relative evaluation* approach, the utility of a service or attribute is derived from the comparison of values from the outcome of the service at hand with those of a similar service, or with expected values [7].

The main difference between these two approaches is that absolute evaluation yields independent measures, while relative evaluation renders comparative measures which require either information about a similar service, or the identification of ideal or expected performance.

As an alternative to quantitative measures, service evaluation can also use qualitative measures by using classification functions (or rules), which appear in several approaches in web services literature [8–10]. These classify services or service attributes according to pre-defined quality categories, such as *terrible*, *poor*, *acceptable*, *good*, or *excellent* for services, and *low*, *medium*, or *high* for attributes response time and cost. The classification function applies pre-defined thresholds to determine the evaluation of a service or attribute so that, for example, the attribute response time is evaluated as *high* when it is less than 10 seconds, as *low* when it is more than 100 seconds, and so on. In addition to evaluation categories, similar approaches consider probabilistic values to represent the degree of pertinence to a specific category, as in evaluating service s_1 to be *good* with a probability of 0.8 [8, 10].

If we consider the periodicity of the evaluation process, we can observe that relative evaluation methods, which are based on the comparison of evaluations of similar services, are more suitable for low frequency and bootstrapping evaluations. Since more than one service must be evaluated during the evaluation process so that comparison is possible, when evaluations occur more frequently, it is more costly for agents to evaluate two or more services at once than to evaluate only one through an absolute evaluation method.

Regarding the characteristics of the evaluated service, it has to be considered that for services which have several possible variations in parameter configuration and each of these can yield a distinct result, it is difficult to identify expected measures of performance. Thus, in this case, absolute evaluation methods are more suitable than relative evaluation methods which are based on expected values.

Finally, to guarantee consistent comparisons between evaluations generated at different points in time, it is more appropriate to use absolute measures than relative ones, because in relative evaluation methods measures are dependent on another service's performance or to an expected value. If the services used as a comparative basis, or the expected values, change from one evaluation to the other, the comparison of evaluations can lose consistency. For absolute evaluation methods, however, the evaluation process is independent of other services and expected performance, so evaluation measures are not biased.

3 General Evaluation Scheme

Considering the issues concerning the evaluation of dynamic services identified previously and the analysis of alternative evaluation approaches, we propose a general evaluation method based on absolute evaluation measures. This approach is more suitable for generating evaluations that can be consistently compared, is less costly over repeated interactions, and does not require the identification of expected performance values (which is not trivial for services influenced by different input configurations like bioinformatics services). The approaches to address the dynamic services' evaluation issues are described in the following.

First, to provide generality, the evaluation must take place over multiple attributes of a service. Thus, agents requesting services with different functionalities may have individual sets of evaluation attributes, but follow the same evaluation scheme.

Second, to ensure continuity and consistency, all attributes must be evaluated according to an independent utility function, which receives as input values that are directly observed from the service outcome. The definition of utility functions based on measures directly observed from results, instead of based on similarity or expected measures, allows the evaluation to be performed after each interaction, since it is independent of other services or specific input configurations.

The choice for *observable* measures as input for utility functions instead of expected or ideal values, also avoids relative evaluations that are not desirable for two main reasons. If expectation values change, evaluations based on old expectations will not be correctly compared with those based on a different, new expectation value. Also, if the service being evaluated is very dynamic and sensitive to different input conditions, it becomes difficult to determine what value to expect.

Finally, to have discriminated information, the evaluation method must generate individual evaluations for each attribute instead of a single evaluation measure. If necessary, a single evaluation measure should be calculated during the selection process. This is because, to combine all attributes in one evaluation, the evaluator would have to consider the relevance of each attribute by assigning different weights according to its preference. However, if the evaluator's preferences change over time and, as a consequence, the weights assigned to each attribute, consistent comparison between evaluations is not possible. The components and processes that implement the solutions described above are presented in the next sections.

3.1 Evaluation Attributes

All services need to be evaluated in relation to certain attributes. For example, to evaluate a database, an evaluator agent usually distinguishes between the aspects it wants to assess, which could be the quality of the data entries, and data access performance. We call these aspects *evaluation attributes*, which indicate what is relevant to evaluate in a service or resource.

Bioinformatics services deal mainly with the analysis of biological data related to DNA and protein sequences aiming at identifying their function in living organisms as well as their structure. From a user's perspective, these services must be capable of identifying all information that is related to the biological data being analysed, so they can have hints about its function and structure. Also, it is important that any similarities that are identified by bioinformatics services are correct by a high degree of confidence. Such data-related aspects are also found in traditional web search engine evaluation metrics [12], with the difference that in traditional web search the query data is usually known. In addition, services in bioinformatics usually handle great amounts of data, and thus service results can take hours or even days to complete. In this case, performance-related aspects also have to be evaluated. Although evaluation criteria regarding performance are generally applied in the evaluation of web services [13], they are not considered in many benchmarks for bioinformatics services [18, 11], which are mainly concerned with data-driven aspects. Based on this analysis, we have identified four evaluation attributes for bioinformatics services which consider both data and performance-related aspects of these services, as described below.

- *Sensitivity* refers to the ability to identify all significant information that is related to the input data independent of its quality.
- *Accuracy* indicates whether result errors were generated from service execution. An algorithm for comparing the similarity of a protein sequences with a sequence database, for example, must be accurate enough to return only matches that are related to the input and to avoid random matches.
- *Reliability* relates to the capacity of the service to deliver results as expected, and to recover results when there is a failure during execution. A reliable service must have a very low frequency of failure.
- *Performance* indicates the time needed to complete a task.

Depending on the service being evaluated, other attributes, in addition to those described above, can be considered for evaluation and added to the list of attributes (see [12] and [13] for an extensive list of possible attributes). We consider similar services as types, and the only constraint on the choice of evaluation attributes is that services of the same type must be evaluated using the same set of evaluation attributes, to guarantee a consistent comparison between them. The choice of a set of evaluation attributes for an agent is related to the objective of the evaluation. For an agent receiving a service, the evaluation is intended to measure its satisfaction with the service results.

3.2 Result measures

Evaluation attributes are measured in terms of elements that can be directly observed from the service results. For example, the sensitivity of a search engine may be measured in relation to the number of matched hits that are related to the input query, and its performance may be measured in terms of the time taken to complete the search. We call these computable elements *result measures*.

Using a more specific view, we define result measures as pieces of information derived from service results that can be used to determine the service’s utility. Result measures are service-dependent, since they relate to the function and purpose of a service.

We distinguish between two types of measures: *static measures* and *dynamic measures*. Static measures are those whose values do not change, or rarely change, from one execution to another, and dynamic measures are those whose values tend to change when the inputs or external conditions vary from one execution to the other. The distinction between these two types of measures is important when considering the continuity of the evaluation process and the characteristics of the services being evaluated.

The accuracy attribute, for example, can be defined in terms of both static and dynamic measures. In case of database search engines, a static measure for the search engine’s accuracy would be, for example, whether new entries submitted to the database are verified by human-experts (a curated database is generally considered more accurate than a non-curated one). This measure does not change over time and, thus, can be evaluated only once instead of every time the service is used. A dynamic measure for the search engine’s accuracy would be the number of matched items returned by the search algorithm that are not relevant to the input query (the fewer the number of irrelevant matches, the more accurate the search engine). Different from the static measure, the dynamic measure can vary from one execution to another when different input data or parameter configurations are used. Thus, the evaluation of dynamic measures must be performed every time the service is used.

3.3 Evaluation Functions

An agent that is involved in an interaction and wants to evaluate a service of type t , uses a set of evaluation attributes represented by $A_t = \{a_i, \dots, a_n\}$. Each evaluation attribute a_i in a set A_t has its own evaluation function, which is expressed in terms of utility. To measure an attribute’s utility, we take into account the result measures associated with that attribute.

When more than one attribute is evaluated, all utility measures must be on the same scale to allow their future combination in a single measure by the selection mechanism. To guarantee that utility measures (U_i) for attributes a_i and $i = \{1, \dots, n\}$ are on the same scale, we define a basic, normalized function which is:

- $U_i = b^c$, for decreasing utility attributes; and
- $U_i = b^{\frac{1}{c}}$, for increasing utility attributes.

Where $b \in (0, 1)$ defines how strict the range of acceptable values is (it alters the shape of the curve in Figure 1) and c represents the result measure associated with attribute a_i . The range of acceptable values is higher for $b > 0.5$ and smaller for $b < 0.5$. According to the function, for decreasing utility attributes the utility will be higher for smaller values of their results measures (c), as shown

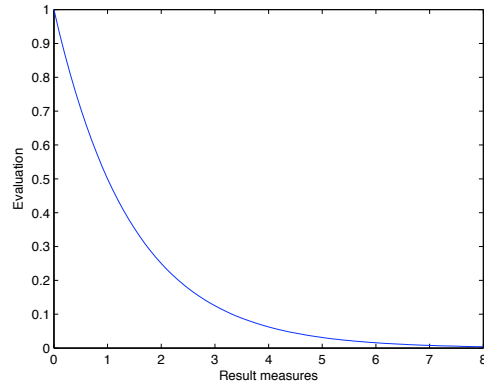


Fig. 1. General utility function for evaluation attributes with $b = 0.5$.

in Figure 1. For increasing utility attributes the utility is higher for higher values of their result measures (c).

3.4 Storing Evaluations

After the evaluation for an attribute is calculated, it is stored by the evaluator agent as an *evaluation tuple*:

$$(r, p, s, C_j, a_i, U_i)$$

where r is the agent requesting the service, p is the agent providing the service, s is the service being evaluated, C_j is the input configuration for service s in the current interaction, a_i is the attribute being evaluated, and U_i is the evaluation of this attribute.

3.5 Evaluation process

For each type of needed service, the evaluator must identify a set of attributes which it considers relevant to evaluate. Also, for each identified evaluation attribute, the result measures that best define that attribute need to be determined and applied to the utility function for that attribute. For dynamic measures, repeated evaluations are more relevant since they allow the analysis of service behaviour under different input configurations.

Every time an agent receives a service, it initiates an evaluation process for that service, which is carried out by the evaluation method. The input for the evaluation method is the service result itself, or information about the service execution process. The output is a set of evaluations, one for each evaluation attribute of the service. The evaluation process for an agent r when interacting with a partner p , and evaluating a service s which was performed using an initial configuration C_j , follows the steps below.

1. For each evaluation attribute a_i in A_i do
 - (a) Compute the *result measure* associated with evaluation attribute a_i based on service result information.
 - (b) Calculate the *evaluation* U_i for a_i using its associated evaluation function, which takes as input the result measures.
 - (c) Store the evaluation for the individual attribute evaluations together with related information in the *evaluation tuple* (r, p, s, C_j, a_i, U_i) .

4 Applying the Evaluation Method

In this section we consider the application of the proposed evaluation method to the bioinformatics domain. In particular, we evaluate services that are used in *proteomics* research [1].

Identification of the proteins which are present in particular cells or tissues of living organisms is one of the main goals of proteomics research. This task is carried out with the help of what is known as a *MS/MS search engine*¹, which receives as input a data file containing a list of peptides from an unknown protein called the *spectra*, and uses *matching algorithms* to search the unknown peptides against a database of known protein sequences. The output is the list of peptides which matched the database, and the protein sequences to which they are associated. The premise behind MS/MS search engines is that if two proteins have similar peptide sequences, they probably have similar function and structure, even if they come from different organisms or different cells.

Alternative MS/MS search engines are publicly available, and differ from each other mainly in the implementation of the matching algorithm. Examples of such search engines are Mascot[14], Tandem[15], and OMSSA[16], some of which run on remote servers, while others run as local services.

Although there are alternative MS/MS search engines with the same functionality, they can yield heterogeneous results for the same input data. Thus, from a user perspective, the evaluation of these search engines can provide a criterion for future selection. In addition, MS/MS search engines are very sensitive to initial configurations and to the quality of the input data, as shown in empirical studies described in [17, 18, 11]. As a consequence, some services may be more suitable for data with a certain quality or for particular configuration setting than others. This means that, even if one search engine performs better when using a particular configuration setting, it may vary its performance when working with a different configuration setting. Thus, MS/MS search engines need a repeated evaluation process, and in terms of their dynamic attributes.

In this context, in the following sections we apply the proposed evaluation method to evaluate MS/MS search engines and show empirical results that support the need for a repeated evaluation in case of dynamic services.

¹ MS/MS is a specific type of input received by the search engines, which comes from mass spectrometry machines.

4.1 Identifying Evaluation Attributes

Experiments for protein identification usually have two focuses: identification of a single protein from unknown input data, or identification of more than one protein from multiple, unknown inputs. In both cases, relevant attributes to be evaluated by the requester are the service *sensitivity*, *accuracy*, and *performance*. The first attribute evaluates the capacity of the search engine to match relevant proteins, which are those associated with a higher number of peptides. The second attribute, accuracy, evaluates the capacity of the search engine to identify true matches and to avoid false positives. Finally, service performance measures the time taken from the input submission until the results are received.

4.2 Identifying Result Measures

Each evaluation attribute is measured according to concrete values that can be observed from parsed service results or from the execution process. For MS/MS search engines, the result measures used to evaluate each attribute are as follows.

Sensitivity The sensitivity of a MS/MS search engine is related to its ability to identify all matches that are related to the input spectra. All search engines have a *significance value* associated to each protein match, which represents the chance of that match being a random match. If this value is above a certain threshold, the protein match is considered a *true positive*. Thus, for this scenario, we measure the sensitivity of a MS/MS search engine in terms of the *number of proteins* identified above the significance threshold, and the *number of peptides* matching the proteins (the *peptide ratio*). The higher the number of significant protein matches and the number of peptides per identified protein, the higher the sensitivity of the search engine.

Both result measures can be directly identified by parsing the search result. The number of proteins is calculated by counting all protein matches returned that are above a significance threshold. A simple way of calculating the number of peptides per protein might be by the simple average of peptide concentrations per matching protein as follows:

$$peptide_ratio = \frac{1}{n} \times \sum_{i=1}^n pp(i)$$

where n is the total number of proteins, and $pp(i)$ is the number of peptides matching protein i .

Accuracy For general Internet search engines like Google or Altavista, a common approach to measure accuracy is to observe the number of occurrences of the query in the matching web site [12]. However, the main difference between MS/MS search engines and general web search engines is that users of the latter engines submit keywords or expressions with the goal of finding related information, while MS/MS search engine users usually submit data with an unknown

identity with the goal of finding similar data that could give a hint about the identity or origin of the submitted data.

The submission of unidentified data to search engines makes it difficult for the evaluator to determine the accuracy of the matched results. Unless users submit data which is already known, they cannot determine whether the match is a false positive without further investigation or relying on a detailed human expert analysis.

Although determining whether a result is accurate without further investigation is a very difficult task, some approaches have been proposed in [18] and [11] that can provide some kind of result validation, as described below.

- *Search with multiple input files*: this sends multiple input files from the same protein and analyses each result individually to identify those matches that were common for all input files and those that were not. The idea is that searching with multiple input files from the same original protein sample reduces the chance of *false positives* from being repeated in all individual results, so that they can be identified as ‘odd’ (false positive) matches.
- *Corroborate results of different services*: this repeats the request sent to one service in a different service and compares the matches given by both. The premise is that services give top hits similar enough to validate each other’s results and, thus, the comparison of different results can identify both *true positives* and *false positives* that are between the best hits.
- *Compare matched protein masses with expected mass*: usually, before starting a MS/MS search, requesters have some idea of the mass of the protein that is present in the input file. Thus, the result can be validated by comparing the mass of top match proteins with the expected mass. True matches must have similar mass to the expected one.

Apart from the fact that the corroboration approach depends on another service, any of these approaches could be adopted to determine the service accuracy, since all provide concrete, observable result measures to be used by the proposed evaluation method. For example, if we take the *number of false positives* as a result measure for service accuracy, all three validation approaches provide different ways of getting the information, and the choice of which to adopt is left to specialist users. For the scope of this paper, we do not implement the evaluation of this attribute in the empirical study.

Performance The performance of a MS/MS search engine can be measured by the *response time*, which represents the amount of time a requester has to wait for the service result. The response time differs from processing time because the former considers the influence of network traffic. From the point of view of a requester, it is more relevant to consider the performance of the MS/MS service in terms of response time.

As with the other result measures, the response time can be determined during the execution process. In the current approach, contextual information that might be useful in determining external factors like network traffic or provider

overhead are not considered. However, if services are repeatedly evaluated, it may be possible to identify such variations in performance.

4.3 Evaluation Functions

Since search engines match each individual entry in an input file with the database, the size of the input file (the number of entries) will influence some evaluation attributes like performance and sensitivity. In other words, the larger the input file, the bigger the expected response time for the service, and the higher the expected number of matching proteins and peptide ratio.

In this context, the evaluation of each attribute of the MS/MS search engine considers the size of the input data in addition to the respective result measures.

Sensitivity The utility function that evaluates the sensitivity attribute is given in terms of result measures associated with that attribute, in this case the number of proteins and peptide ratio. A desirable service is one given a result with a higher number of significant protein matches and a higher concentration of peptides per protein, indicating that there was a big coverage of the protein sequence. The utility function that reflects the desirable result is in the form:

$$U_s = 0.5^{sm}$$

where sm (sensitivity measure) is the relation between the number of proteins returned by the service, peptide ratio, and the input size (in Kbytes).

$$sm = \frac{input_size}{protein_number \times peptide_ratio}$$

Here, the service sensitivity increases as the number of identified proteins and associated peptides increase. Thus, the value of sm is calculated in an inverted form (with a decreasing value for higher result measures) to reflect this behaviour in the utility function. Since the number of proteins is usually much smaller than the peptide ratio, the latter has a bigger contribution to the utility.

Performance A desirable result in terms of performance will have small values for response time. Thus, the evaluation of service performance is given by the normalized utility function:

$$U_p = 0.5^{rt}$$

where rt is the relation between the service response time and the size of the input file, and is given by the following:

$$rt = \frac{response_time}{input_size}$$

The response time is given in seconds, and the input size is given in Kbytes. According to the function, service performance will be higher for smaller values of rt and, consequently, for smaller response times.

<i>Parameter</i>	C_1	C_2
Database	NCBIInr	NCBIInr
Enzyme	Trypsin	Trypsin
<i>Taxonomy</i>	<i>Mammals</i>	<i>All entries</i>
<i>Fixed Modifications</i>	<i>Carbamidomethyl (C)</i>	<i>None</i>
Potential Modifications	None	None
Peptide Tolerance	2.0Da	2.0Da
Fragment Tolerance	0.8Da	0.8Da
Missed Cleavages	1	1

Table 1. Initial configurations for MS/MS search services.

4.4 Results

We apply the proposed evaluation method to evaluate four different search engines, two locally and two remotely accessed. These search engines are Mascot remote [14], Tandem local and remote [15], and OMSSA local [16].

For this empirical study, the requests were submitted to all services using the same input spectra (580.8Kb), and two different input configurations, as shown in Table 1. Configurations C_1 and C_2 have different settings for parameters *Taxonomy* and *Fixed Modifications*, while the others are kept the same. We repeated the evaluation process for each input configuration using the same spectra to observe the changes in evaluation results.

Results for the evaluation of the four different MS/MS search services according to the *sensitivity* attribute are shown in Table 2, and the services with best evaluation for this attribute using each configuration are highlighted in bold. Here we observe that, for configuration C_1 *Mascot* has a better sensitivity, but for configuration C_2 , *Tandem local* is better. Evaluation results for the performance attribute are shown in Table 3. Here, we observe that *Tandem local* has better performance when using configuration C_1 , but *Tandem remote* is better when configuration C_2 is used.

From this results we observe that, if the evaluation process had not been repeated after the first interaction with Tandem local, a requester interested in finding the best service in terms of sensitivity would have the wrong information that Mascot would perform better when using configuration C_2 as well.

Similarly, if the evaluation process had not been repeated after an interaction with Tandem remote, a requester interested in finding the best service in terms of performance would have the incorrect information that Tandem local would present better performance also for configuration C_2 .

Moreover, the results show that dynamic services, like those presented on this empirical study, despite having similar functionally, can yield heterogeneous results under different configurations. Thus, there is a need to dynamically evaluate services to improve the efficiency of future selection of alternative services.

Service	protein_number		peptide_ratio		Sensitivity	
	C_1	C_2	C_1	C_2	C_1	C_2
Mascot	13	40	10	9	0.045	0.327
Tandem Local	8	36	11	51	0.010	0.803
Tandem Remote	2	3	9	6	1.9E-10	1.9E-10
OMSSA Local	31	31	4	4	0.039	0.039

Table 2. Evaluating MS/MS search services according to the sensitivity attribute.

Service	response_time (sec)		Performance	
	C_1	C_2	C_1	C_2
Mascot	172	200	0.814	0.787
Tandem Local	11	41	0.986	0.952
Tandem Remote	26	37	0.969	0.956
OMSSA Local	2581	2489	0.045	0.051

Table 3. Evaluating MS/MS search services according to the performance attribute.

5 Conclusion and Future Work

We have presented a general evaluation method for dynamic services to be used by agents in bioinformatics applications. We discuss the issues for efficient evaluation of dynamic services, which include the adoption of a repeated evaluation process, the use of absolute evaluations, and the generation of comparable evaluations, and describe the components of the evaluation method.

We apply the evaluation method to evaluate services for protein identification, and show the importance of a dynamic evaluation process for those services through empirical results. Results show that there is a need to dynamically evaluate services to have more accurate information about their results, so that agents requesting services in dynamic environments can improve the selection of alternative services in the future. Future work aims to develop selection strategies for agents using service evaluation, considering both individual attribute evaluations and a combined evaluation of different attributes.

Although the evaluation method targets services in the proteomics domain, it can also be applied to services in other domains that share the same characteristics of dynamism (where different evaluation criteria may be required, but using the same method).

References

1. Campbell, A.M., Heyer, L.J.: Discovering Genomics Proteomics and Bioinformatics. Benjamin Cummings, San Francisco CA (2002)
2. Bryson, K., Luck, M., Joy, M., Jones, D.T.: Applying agents to bioinformatics in geneveaver. In: Cooperative Information Agents IV. Volume 1860 of Lecture Notes in Artificial Intelligence. Springer-Verlag (2000) 60–71

3. Decker, K., Zheng, X., Schmidt, C.: A multi-agent system for automated genetic annotation. In: Fifth International Conference on Autonomous Agents, Montreal (2001) 433–440
4. Edwards, W., Newman, J.R.: Multiattribute evaluation. In Sullivan, J.L., Niemi, R.G., eds. Volume 26 of Quantitative Applications in the Social Sciences. Sage, CA (1982) 7–94
5. Russel, S., Norvig, P.: Intelligent Agents. In: Artificial Intelligence: A Modern Approach. Prentice Hall, New Jersey (1995) 31–50
6. Yoon, K.P., Hwang, C.: Multiple attribute decision making: An introduction. In Lewis-Beck, M.S., ed. Volume 104 of Quantitative Applications in the Social Sciences. Sage, CA (1995) 1–75
7. Caverlee, J., Liu, L., Rocco, D.: Discovering and ranking web services with BASIL: a personalized approach with biased focus. In: International Conference on Service-oriented Computing, New York, ACM Press (2004) 153–162
8. Casati, F., Castellanos, M., Dayal, U., Shan, M.: Probabilistic context-sensitive and goal-oriented service selection. In: Second International Conference On Service Oriented Computing, New York USA (2004) 316 – 321
9. Chakraborty, D., Jaiswal, S.K., Misra, A., Nanavati, A.A.: Middleware architecture for evaluation and selection of 3rd party web services for service providers. In: IEEE International Conference on Web Services, IEEE Press (2005)
10. Day, J., Deters, R.: Selecting the best web service. In: Conference of the Centre for Advanced Studies on Collaborative research, Ontario Canada, IBM Press (2004) 293–307
11. Kapp, E.A., Schtz, F., Connolly, L.M., *et al*: An evaluation, comparison, and accurate benchmarking of several publicly available ms/ms search algorithms: Sensitivity and specificity analysis. *Proteomics* **5**(13) (2005) 3475–3490
12. Dhyani, D., Wee, K., Showmick, S.S.: A survey of web metrics. *ACM Computing Surveys* **34**(4) (2002) 469–503
13. Lee, K., Jeon, J., Lee, W., Jeong, S., Park, S.: Qos for web services: Requirements and possible approaches. Working group note, W3C (2003)
14. Perkins, D.N., Pappin, D.J.C., Creasy, D.M., Cottrell, J.S.: Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis* **20**(18) (1999) 3551–3567
15. Craig, R., Beavis, R.C.: A method for reducing the time required to match protein sequences with tandem mass spectra. *Rapid communications in mass spectrometry* **17** (2003) 2310–2316
16. Geer, L.Y., Markey, S.P., Kowalak, J.A., *et al*: Open mass spectrometry search algorithm. *Proteomics Research* **3** (2004) 958–964
17. Boutilier, K., Ross, M., Podtelejnikov, A.V., Orsi, C., Taylor, R., Taylor, P., Figeys, D.: Comparison of different search engines using validated ms/ms test datasets. *Analytica Chimica Acta* **534** (2005) 11–20
18. Chamrad, D.C., Korting, G., Stuhler, K., Meyer, H.E., Klose, J., Bluggel, M.: Evaluation of algorithms for protein identification from sequence databases using mass spectrometry data. *Proteomics* **4**(3) (2004) 619–628