# A Generic SLA Negotiation Protocol For Virtualized Environments

Rafael L. Gomes, Luiz F. Bittencourt, Edmundo R. M. Madeira
University of Campinas (UNICAMP), Brazil
email: {rafaellgom, bit, edmundo}@ic.unicamp.br

***Abstract*—Many companies use the Internet as a basis for their services, defining Service Level Agreements (SLA) with their respective Internet Service Providers. However, the current Internet works in a Best Effort manner, what points toward the concept of network virtualization to support the Future Internet. Within this context, this work proposes a generic SLA negotiation protocol for Virtualized Environments (VEs). The proposed protocol allows the client to negotiate resources (for example, the bandwidth for the virtual network or the storage capacity of the cloud) and features (for example, protocol stack of the virtual network or operating system in the cloud) used in the VE. Experiments showed the effectiveness of the proposed protocol in fulfilling the requirements defined by the client.**

***Index Terms*—Service Level Agreement, Virtualization, Negotiation, Future Internet, Dynamic Resource Provision.**

## I. Introduction

Internet has been growing and many companies use it as a basis for their services. To utilize a service available through the Internet, clients usually establish with the desired provider a service level agreement (SLA) that specifies properties which must be maintained during service provisioning.

Currently, there is no guarantee of service level on the Internet. Hence, there is a consensus that it needs to be updated, creating the "Future Internet". Along with this, the Network Virtualization (NV) arises as one of the most important technologies for the Future Internet. In a nutshell, the NV is a technology that enables the deployment of multiple environments over the same physical infrastructure [1].

The flexibility of virtualization allows users and providers to negotiate several virtual services, with aspects related to both resources and features utilized. Within this context, this work proposes a generic SLA negotiation protocol for Virtual Environments (VEs). The proposed protocol allows clients to negotiate not only resources, but also to negotiate the features of the VE with the service provider. This characteristic is fundamental for the protocol to act in a variety of virtualized environments, such as virtual networks and clouds. The proposed negotiation process is based on similarity and multicriteria decision making (MCDM) methods.

We consider as *resources* the parameters that are measurable, such as bandwidth for virtual networks and clock speed for virtual machines in clouds. On the other hand, we consider as *features* the parameters that describe characteristics or behaviors of the VE. For example, the routing protocol in a virtual networks, and operating systems in the case of clouds.

The proposed protocol aims to model the negotiated VE according to the client needs. For example, if a client needs two virtual networks, one for multimedia traffic (high traffic requirements) and another one for traditional data traffic (low traffic requirements), two networks could be negotiated: a virtual network with MPLS (MultiProtocol Label Switching) for multimedia traffic, and another simpler network with the Open Shortest Path First (OSPF) for traditional data traffic. Likewise, for a cloud environment, it can negotiate parameters such as Operating System (OS), storage capacity, and so on.

This paper is organized as follows. Section II presents related work regarding SLA negotiation, while Section III introduces the main issues involving this problem. Section IV describes the proposed generic SLA negotiation protocol. Experiments are shown in Section V, and Section VI summarizes the paper and presents future work.

## II. Related work

In this section we describe some important works related to SLA negotiation.

Modica et al. [2] aim to enhance the flexibility of the WS-Negotiation, a negotiation protocol that is based on WS-Agreement. It enables the renegotiation and modification of QoS guarantees while the service is being provided.

Al-aaidroos et al. [3] propose an agent-based conceptual framework for web service SLA negotiation which enables a single service provider to negotiate with multiple web service consumers at the same time. The main goal of the proposal is to accelerate the negotiation process through software agents representing consumers and providers at the back-end system.

Zaheer et al. [5] show V-Mart, an open market model for automated service negotiation in Network Virtual Environments for Virtual Network Providers (VNP) and Infrastructure Providers (InP). For InPs, V-Mart fosters an open competition environment through auctioning, and the VNP can disseminate a request for quotation when it desires to set up a VN.

Gomes et al. [6] propose a SLA negotiation protocol for virtual networks, negotiating the network resources and the protocol stack of VNs. However, the proposal is limited for fixed virtual network parameters, preventing the generic negotiation of different parameters.

None of the papers found in the literature focus on the generic negotiation of features and resources of virtualized environments, which is the proposal of this work.

## III. SLA negotiation and language specification

A SLA is an agreement between parties, for example a provider and its client. Using an automatic approach, the

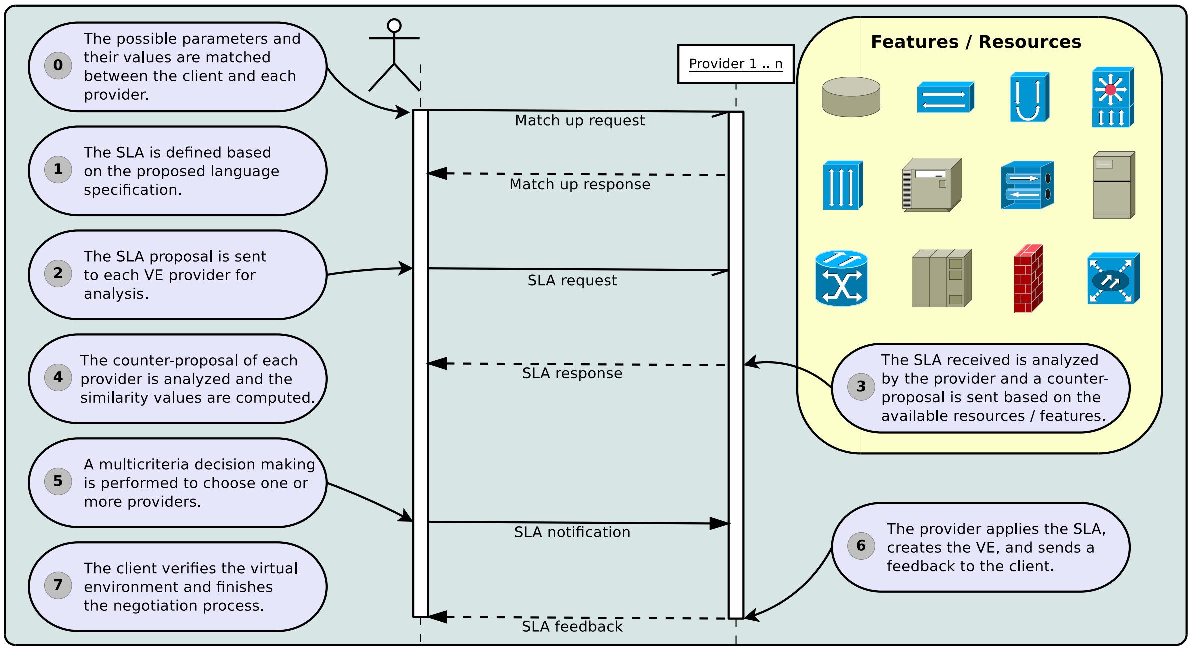

0 The possible parameters and their values are matched between the client and each provider.
1 The SLA is defined based on the proposed language specification.
2 The SLA proposal is sent to each VE provider for analysis.
4 The counter-proposal of each provider is analyzed and the similarity values are computed.
5 A multicriteria decision making is performed to choose one or more providers.
7 The client verifies the virtual environment and finishes the negotiation process.
Provider 1 .. n
Features / Resources
Match up request
Match up response
SLA request
SLA response
3 The SLA received is analyzed by the provider and a counter-proposal is sent based on the available resources / features.
SLA notification
6 The provider applies the SLA, creates the VE, and sends a feedback to the client.
SLA feedback

Fig. 1. Steps involved in the automatic negotiation mechanism.

negotiation can achieve the objective that has been requested by the client. Automated negotiation is important when the consumer of a service is a software that has to negotiate the SLA on the fly. The steps involved in the proposed mechanism for automatic negotiation are shown in Figure 1. This figure presents only the entity *Provider i*, but the negotiation steps occur for each provider defined by the client.

In step “0”, before the negotiation starts, the client matches the possible parameters and their values with all providers, allowing a fair evaluation of the parameters. After the client has defined the SLA (step “1”), a SLA proposal is sent to each provider (step “2”). Each provider analyzes the received SLA and sends a counter-proposal based on its available parameters (step “3”). The counter-proposal is analyzed by the client using the process described in the Section IV, and the similarity values for the response are generated (step “4”). With all counter-proposals analyzed, an MCDM is performed to choose the best option to deploy the SLA, notifying the chosen provider(s) (step ”5“). The chosen provider applies the SLA, creating the VE and sending a feedback to the client (step “6”), which verifies the VE (step ”7“), completing the negotiation process. Steps “6” and “7” are performed only by providers chosen in step “5”.

For the SLA definition in step “1”, the service description as well as its guarantees must be described, where a common SLA specification language for negotiation is necessary. In this work we propose a class-based language specification of SLA for VEs, allowing the negotiation of resources and also features for the VE. The proposed language specification is based on the XML, thus with intrinsic expressive capacity to describe the service and general definitions.

The SLA must have some elements in its description: the parties, agreement parameters, description of the services (usually measurable parameters), obligations, and the cost of the services. Beyond these traditional SLA elements related to the VEs, this work defines as additional elements: the description of the classes and features to be applied in the desired class. Figure 2 shows the language specification.

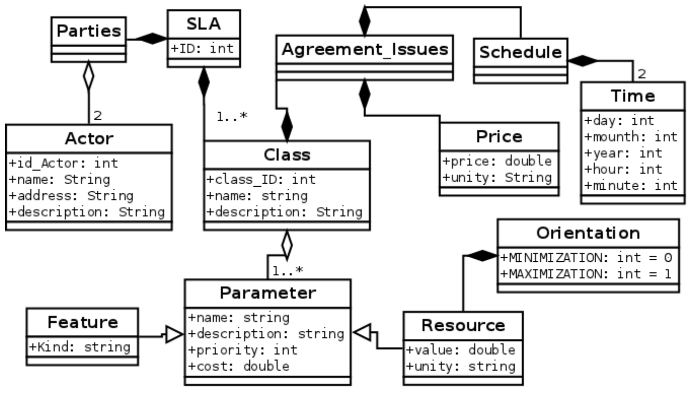


Fig. 2. Diagram of the proposed language specification.

First, we have the component “SLA”, which is the root of the SLA, containing an identification (ID) for it. The component “Parties” represents two “Actor” components, with information of both client and provider (one “Actor” component for each party). The component “SLA” can define one or more “Class” components, which describe the classes of service related to the VE. For example, we can have one class defining a cloud and another one defining a virtual network to access the cloud. Or, we can have the definition of two virtual networks with different configurations. The component “Agreement_Issues” represents the information about the duration of the SLA, violation, and the price for the negotiated class.

Inside each class we can define the component “Parameter”, which can be of two types: “Feature” or “Resource”. The component “Parameter” has some attributes to identify the parameters and to describe information regarding cost and priority (used in the negotiation process), where these attributes are inherited by the components "Feature" and "Resource"

The component "Feature" describes the non-measurable parameters related to the configuration of the VE. As an example, in a virtual network we can have some protocols, where each protocol would be a "Feature". The attribute *kind* defines the type of the feature, and it is used to allow the client/provider to identify protocols that have the same kind in the negotiation process. For example, we cannot compare the OSPF with the IPv4 in the negotiation, so using the *kind* we identify IPv4 as an addressing protocol and OSPF as a routing protocol. With this, each protocol will be compared only with protocols of the same kind. We can have the same situation when comparing OSs in a cloud environment, and in many other situations. This approach is based on the idea that the information about *kind* is matched up between the parties.

The component "Resource" represents measurable parameters of the VE, so we can define resources such as bandwidth, storage amount, and so on. This component has two attributes to define the value of the parameter and has the "Orientation" component. The "Orientation" describes whether, in the negotiation process, the parameter is intended to be increased or decreased. For example, resources as bandwidth or clock speed are intended to be increased (the higher the value, the better). Likewise, loss or delay are intended to be decreased (the lower the value, the better).

With the developed language the client can define SLAs using some classes, each class with its particular parameters (resources, features, contract duration, price and others). Then, the SLA is used for automatic negotiation, where the decision model of the negotiation party implies to: define attribute constraints; identify the desired objectives; and prioritize these objectives. Any MCDM technique can be used in such decision model. In this work we used the well-known Analytic Hierarchy Process (AHP) method [7].

## IV. Proposed Generic SLA Negotiation Protocol

Virtualization brings the possibility of adapting the infrastructure for diverse client needs. Thus, VEs can be defined to represent client requirements. This paper proposes a generic SLA negotiation protocol for VEs which has the ability to negotiate both features and resources.

The proposed protocol is based on similarity and MCDM methods. The similarity approach is used to compare the parameters in the SLA defined by the client with the response sent by the provider, while the MCDM is used to decide which provider should deploy the SLA based on three criteria: price, feature similarity, and resource similarity. These three criteria are associated with priorities defined by the client, indicating which one is preferable, and then the MCDM is applied.

Now the negotiation process will be detailed. First, the client and the provider match up their parameters, ensuring a fair evaluation of the SLA proposal. After that, the client sends a SLA proposal to the provider, according to the SLA language described in Section III.

When the provider receives the SLA proposal, it analyzes, for each defined class, the following issues:

- **Agreement issues**: the provider verifies if the deployment time can be obeyed.
- **Resource issues**: the provider analyzes if the resource amounts can be guaranteed. If the amount can be fully guaranteed, the provider replies with the original value received. But, if the provider can only partially meet the required resource amount, it sends the amount that can be guaranteed. If the provider does not support the resource, it is deleted from the SLA response.
- **Feature issues**: if the provider can deploy the VE with the features defined by the client, the same configuration is replied. However, if any feature cannot be deployed, it sends another one according to the provider settings. For example, if the client requests an OSPF protocol and the provider does not support it, the provider can send the RIP as response, or even disconsider that protocol in the SLA response.

After receiving the SLA response from the provider, the client analyzes each class in the SLA. First, the client verifies if the agreement issues are compatible with the required ones. After that, the client analyzes the resource parameters to generate the similarity rate. The same is done for the feature parameters. This process, which is performed by the client for each provider, is described in the following subsections.

### A. Resource Analysis

For *resource* parameters, the similarity is calculated according to the following: if the resource amount should be maximized (bandwidth, for instance), the similarity is calculated as shown in Equation 1; if the resource amount should be minimized (delay, for instance), the similarity is calculated as shown in Equation 2. $Value_{requested}$ is the value requested by the client in the proposal, while $Value_{received}$ is the value in the response given by the provider.

$$Sim_{max}(resource) = \frac{Value_{received}}{Value_{requested}} \tag{1}$$

$$Sim_{min}(resource) = \frac{Value_{requested}}{Value_{received}} \tag{2}$$

The two defined equations aim to represent the required amount that can be met by the provider. Therefore, $0 \leq Sim_{max} \leq 1$ and $0 \leq Sim_{min} \leq 1$. These definitions are under the assumption that the provider will send replies with values equal or less than the requested value for resources that requires maximization, and equal or greater otherwise.

Let $\mathcal{R}$ be the set of resource parameters requested by the client, $Sim_r$ be the computed similarity using Equation 1 or Equation 2) for resource $r$, and let $\omega_r$ be the priority associated with resource $r$ ($\omega_r \in \mathbb{N}$). After calculating the similarity for each requested resource parameter, the Equation 3 is used to generate the final resource similarity ($Sim_{resource}$) for the provider being considered.

$$Sim_{resource} = \frac{\left(\sum_{r \in \mathcal{R}} \omega_r * Sim_r\right)}{\left(\sum_{r \in \mathcal{R}} \omega_r\right)} \tag{3}$$

## B. Feature Analysis

For the feature similarity calculation, techniques of distance for nominal variables are used. A nominal variable is used when a number is semantically only a reference, instead of representing quantity. A desired characteristic for nominal variables is the consistent labeling.

The similarity technique used in the feature analysis is based on [8], and in our context all variables are *kinds*, i.e., attributes that define the type of the feature (for example routing protocols or addressing protocols).

To calculate the distance between two objects represented by nominal variables, we need to consider the number of categories (possibilities) on each variable. If the number of categories is only two, we can use distance for binary variables. On the other hand, if the number of categories is more than two, we need to transform these categories into a set of dummy variables that assume binary values.

To deal with non-binary variable distances, let $C$ be the number of categories. Then we can assign each value of the category into $n_{dv}$ dummy variables with binary values, which must satisfy the condition $C \leq 2^{n_{dv}}$, thus it can be computed as in Equation 4.

$$n_{dv} = \left\lceil \frac{log^C}{log^2} \right\rceil \tag{4}$$

For example, we can consider two kinds of protocol: routing protocols and addressing protocols. Addressing protocols may assume two reference numbers IPv4 = 0 and IPv6 = 1, and routing protocols may have three options RIP= 0, OSPF = 1 and EIGRP = 2.

According to Equation 4, for the addressing protocols we set only one dummy variable ($dv_{a_1}$), and for the routing protocols we have to use two dummy variables ($dv_{r_1}$ and $dv_{r_2}$). Using the schema described before, we have the variable definitions shown in (5). To convert the values representing the option to the dummy variables schema, we transform the values into corresponding binary numbers. In our example the EIGRP protocol is represented by (1,0).

$$\begin{aligned} &Addressing \{ dv_{a_1} = \{0,1\} \\ &Routing \begin{cases} dv_{r_1} = \{0,1\} \\ dv_{r_2} = \{0,1\} \end{cases} \end{aligned} \tag{5}$$

The client requirements are represented by a vector $\mathcal{V}$ of dummy variables. Thus, a client request in our example is represented by the vector $\mathcal{V} = (dv_{a_1}, (dv_{r_1}, dv_{r_2}))$. For example, if the client requests IPv6 and OSPF, we have $\mathcal{V} = (1,(0,1))$ as the feature vector. Likewise, if a provider has the protocols IPv4 and RIP, the client receives the response $\mathcal{V} = (0,(0,0))$.

After that, we need to calculate the distance for each variable, where any distance method can be used. In this paper the Unmatched distance ($\mathcal{U}_d$) is used [8]. $\mathcal{U}_d$ is the number of positions at which corresponding symbols are different divided by the number of positions. So, $\mathcal{U}_d$ always has values between 0 and 1. Using the $\mathcal{U}_d$ in our example, we obtain the distance $\mathcal{U}_d = 0.5$ between the routing client request $(0,1)$ and the provider response $(0,0)$.

To generate the similarity $Sim_f$ for each feature $f$, the correspondent unmatched distance ($\mathcal{U}_{d_f}$) is subtracted from 1, as shown in Equation 6. Since $\mathcal{U}_d$ is a value between 0 and 1, the similarity will be in this range too.

$$Sim_f = 1 - \mathcal{U}_{d_f} \tag{6}$$

To calculate the final feature similarity ($Sim_{feature}$) for the provider being considered, the Equation 7 is used, where $\mathcal{F}$ is the set of feature parameters requested by the client and $\omega_f$ is the priority configuration for the feature $f$ ($\omega_f \in \mathbb{N}$).

$$Sim_{feature} = \frac{\left(\sum_{f \in \mathcal{F}} \omega_f * Sim_f\right)}{\left(\sum_{f \in \mathcal{F}} \omega_f\right)} \tag{7}$$

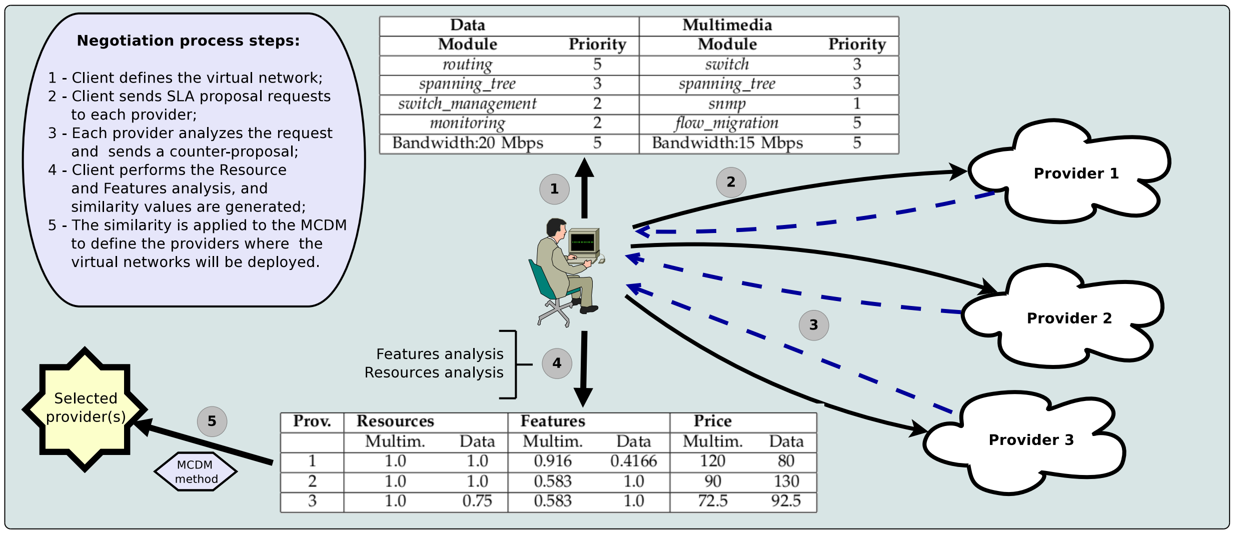
Besides defining priorities for resources and features to compute their similarity, the customer also defines priorities for each criteria (price, features similarity, and resources similarity). Therefore, with the similarities computed, the MCDM input comprises the $Sim_{feature}$, the $Sim_{resource}$, and the price for each provider, as well as three priorities defined by the client for these three criteria. With this information, the MCDM method chooses the best provider to deploy the SLA.

## V. Experiments

This section aims to show the decision making capacity of the negotiation process. We want to evaluate the effectiveness of the proposed protocol in modeling the parameters of the VE and fulfilling the requirements defined by the client.

The experiments consist of an Openflow network negotiation [9], where the features of negotiation are modules of Nox Controllers [10]. The choice of the Openflow is because many Future Internet projects around the world are using it for their experiments to support new protocols and virtualization-based architectures. Moreover, the decoupling of network control plane and the physical topology, performed by Openflow, allows the negotiation of specific kinds of modules.

The experiment comprises one client and three providers, where two virtual networks are negotiated: one for multimedia traffic and another one for data traffic. Table I shows the modules used in the experiments, their respective kinds and reference numbers. The reference numbers are used to generate the similarity between the negotiated modules, as described in Section IV-B. The modules *flow_migration* and *statapp* can be found in the Omni tool [11], whereas the other ones



**Negotiation process steps:**

1 - Client defines the virtual network;
2 - Client sends SLA proposal requests to each provider;
3 - Each provider analyzes the request and sends a counter-proposal;
4 - Client performs the Resource and Features analysis, and similarity values are generated;
5 - The similarity is applied to the MCDM to define the providers where the virtual networks will be deployed.

| Data | | Multimedia | |
|---|---|---|---|
| Module | Priority | Module | Priority |
| *routing* | 5 | *switch* | 3 |
| *spanning_tree* | 3 | *spanning_tree* | 3 |
| *switch_management* | 2 | *snmp* | 1 |
| *monitoring* | 2 | *flow_migration* | 5 |
| Bandwidth:20 Mbps | 5 | Bandwidth:15 Mbps | 5 |

| Prov. | Resources | | Features | | Price | |
|---|---|---|---|---|---|---|
| | Multim. | Data | Multim. | Data | Multim. | Data |
| 1 | 1.0 | 1.0 | 0.916 | 0.4166 | 120 | 80 |
| 2 | 1.0 | 1.0 | 0.583 | 1.0 | 90 | 130 |
| 3 | 1.0 | 0.75 | 0.583 | 1.0 | 72.5 | 92.5 |

Fig. 3. Steps in the negotiation and experimental configuration, with client's virtual network requests considering three providers.

can be found in the default Nox controller [10]. Moreover, the last three columns of Table I are the prices for modules (features) available in each provider (P1, P2, and P3), whereas the character "X" indicates that the provider does not support the module.

TABLE I
MODULES USED IN THE EVALUATION SCENARIO.

| Module | Kind | Ref. | Desc. | P1 | P2 | P3 |
|---|---|---|---|---|---|---|
| *hub* | Forwarding | 1 | Layer 2 packet forwarding | 0 | 0 | 0 |
| *switch* | | 2 | | 10 | 20 | 10 |
| *pyswitch* | | 3 | | 10 | 20 | 10 |
| *routing* | | 4 | | X | 30 | 10 |
| *flow migration* | Flow Migration | 1 | Migrates flows between switches | 50 | X | X |
| *discovery* | General | 1 | Allows Link discovery | 10 | 10 | 10 |
| *topology* | | 2 | Topology maintenance | 10 | 10 | 10 |
| *authenticator* | | 3 | Active host storage | 10 | 10 | 10 |
| *switch management* | Switches Management | 1 | Creation, modification and exclusion of flows | 10 | 10 | 10 |
| *snmp* | | 2 | | X | 10 | 10 |
| *statapp* | Statics | 3 | Collects statistics | 20 | X | 20 |
| *switchstats* | | 2 | | 20 | 20 | 20 |
| *monitoring* | | 1 | | X | 20 | 20 |
| *spanning tree* | Loop Management | 1 | Avoids loops | 30 | 30 | 30 |

Besides offering the features listed in Table I, each provider offers the resource *Bandwidth*: P1 has 40 Mbps costing 2 per Mbps; P2 has 40 Mbps costing 2 per Mbps; P3 has 30 Mbps costing 1.5 per Mbps.

Figure 3 illustrates the steps performed in the negotiation process, showing the virtual networks requested by the client in the experiments, as well as the similarity values generated from the response received from each provider using the data presented in Table I. The steps presented in the Figure 3 follow the negotiation process described in Section III (Figure 1).

In the case of the Provider 3, it fully complies with the Bandwidth for the Multimedia network. For the Data network, it has only part of the required Bandwidth. Likewise, regarding the features for the Multimedia network, only Provider 1 has the high priority *flow_migration* module. As a consequence, Provider 1 gets a higher similarity.

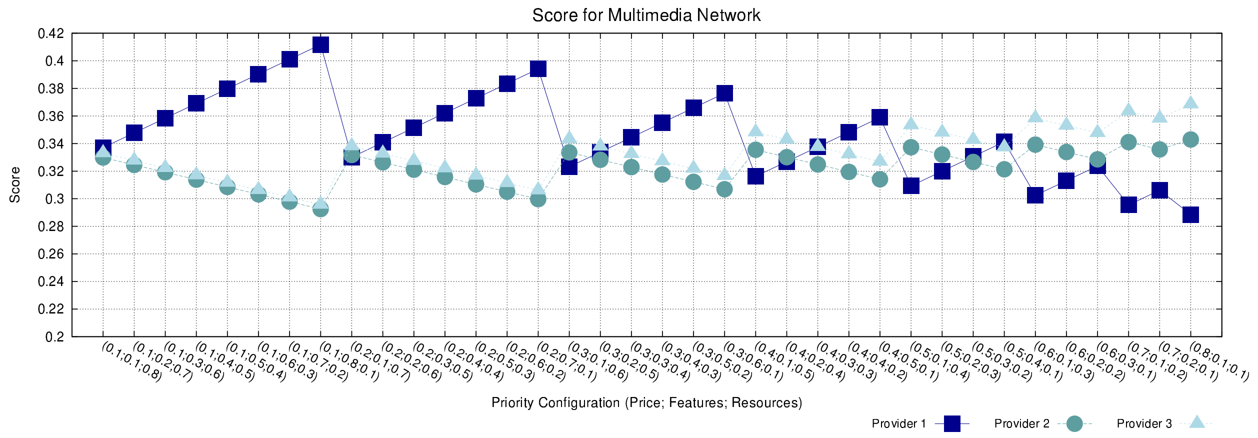
However, regarding the Data network features, Provider 1 does not have the *routing* and *monitoring* modules, where the first one has a high priority. Thus, Provider 1 has a lower similarity compared with the other providers. Analyzing the Price defined by the providers, we note a trade-off between the Price and the negotiated parameters. The more the negotiated network approaches the client requirements, the more expensive it is.

Figure 4 shows the experimental results. The "Y" axis represents the Score of the providers generated by the MCDM method, and the "X" axis shows 36 possibilities of the tuple $(price; features; resources)$, representing the priority configuration for each criterion. Due to the use of the AHP method, the provider with higher Score is chosen to deploy the virtual network.

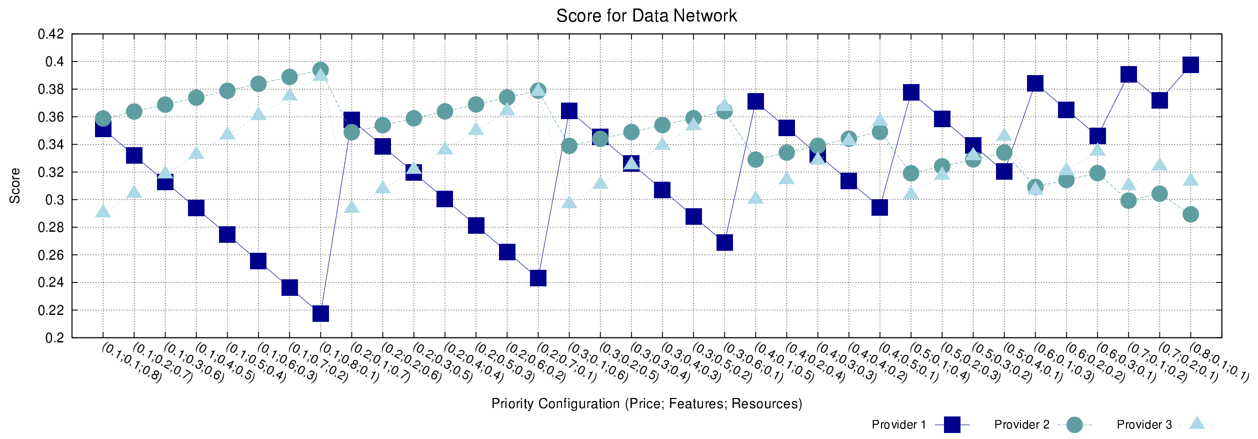
Regarding the results for the multimedia network, shown in Figure 4(a), Provider 1 presents higher scores in situations where the Price priority is lower, following the data shown in Figure 3. Provider 1 best complies with the requirements from the client, however with a higher cost. Likewise, the behavior for the Providers 2 and 3 follows the same reasoning when they have lower priority for the Feature criterion, which has smaller similarity value.

In the results for the Data Network, shown in Figure 4(b), we can see that Provider 1 is advantageous in situations where the Price has higher priority, due to its lower cost to deploy the network. On the other hand, Provider 2 is the chosen in most cases due to its high similarity for Resources and Features. Provider 3 is chosen only in situations where both Price and Feature priorities are higher, since it has a lower cost than Provider 2, as well as a high Features similarity.

The shown experiments suggest that the proposed negotiation protocol can meet the specifications set by the client. This effectiveness is due to the parameter modeling based on the similarity approach and priorities, suitable as input for a multicriteria decision making method. Therefore, the negotiation protocol enables the client to choose the focus of the negotiation process, adapting the choices made according to the settings defined by the client.



(a) Score of each provider for the multimedia network



(b) Score of each provider for the data network

Fig. 4. Score results according to the criterion priority configuration.

## VI. Conclusion

This paper presents a generic SLA negotiation protocol and a language specification for virtual environments. The proposed protocol allows clients to negotiate resources and features to be deployed in the virtual environment. The proposal is based on similarity and priorities applied to a multicriteria decision making (MCDM) method. The similarity is used to compare the SLA proposals, and an MCDM approach is defined to choose which provider is more suitable to deploy the SLA, according to the client priorities. Experiments showed the effectiveness of the proposed protocol, attesting that the combination of similarity, priorities, and MCDM techniques is a powerful approach to support SLA negotiation.

As future work, we will extend the protocol to support renegotiation of the SLA and end-to-end SLA negotiation.

## Acknowledgment

The authors would like to thank FAPESP (2012/04945-7), CAPES, CNPq and RNP for the financial support.

## References

[1] N. M. M. K. Chowdhury and R. Boutaba, "Network virtualization: state of the art and research challenges," *IEEE Communications Magazine*, vol. 47, no. 7, pp. 20–26, 2009.

[2] G. Di Modica, V. Regalbuto, O. Tomarchio, and L. Vita, "Enabling re-negotiations of SLA by extending the ws-agreement specification," in *IEEE Intl. Conference on Services Computing.*, july 2007, pp. 248 –251.

[3] M. Al-aaidroos, N. Jailani and M. Mukhtar, "Agent-based negotiation framework for web service's SLA," in *7th International Conference on Information Technology in Asia (CITA 11)*, 2011.

[4] E. Nitto, M. Penta, A. Gambi, G. Ripa, and M. L. Villani, "Negotiation of service level agreements: An architecture and a search-based approach," in *5th Intl. conference on Service-Oriented Computing*, 2007, pp. 295–306.

[5] F. Zaheer, J. Xiao, and R. Boutaba, "Multi-provider service negotiation and contracting in network virtualization," in *Network Operations and Management Symposium (NOMS), 2010 IEEE*, april 2010, pp. 471 –478.

[6] R. L. Gomes and E. Madeira, "An automatic SLA negotiation protocol for a future internet," in *Proceedings of IEEE 3rd Latin-American Conference on Communications*, 2011.

[7] A. M. S. Alkahtani, M. E. Woodward, and K. Al-Begain, "Prioritised best effort routing with four quality of service metrics applying the concept of the analytic hierarchy process," *Computers and Operations Research*, vol. 33, pp. 559–580, March 2006.

[8] K. Teknomo, "Similarity measurement," *Available at: http://people.revoledu.com/kardi/tutorial/Similarity/*

[9] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[10] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "Nox: towards an operating system for networks," *SIGCOMM Computer Communication Rev.*, vol. 38, pp. 105–110, July 2008.

[11] D. M. F. Mattos, N. C. Fernandes, V. T. da Costa, L. P. Cardoso, M. E. M. Campista, L. H. M. K. Costa, and O. C. M. B. Duarte, "Omni: Openflow management infrastructure," in *Intl. Conference on the Network of the Future*, 2011.