Design, implementation and evaluation of virtual resource description and clustering framework

Houssem Medhioub, Ines Houidi, Wajdi Louati, Djamal Zeghlache

Design, Implementation and Evaluation of Virtual Resource Description and Clustering Framework

Houssem Medhioub, Ines Houidi, Wajdi Louati and Djamal Zeghlache
Institut Télécom – Télécom SudParis
{firstname.lastname}@it-sudparis.eu

***Abstract*—This paper presents an approach to speed up and enhance matching of virtual network requests to available resources in virtual network provisioning frameworks. The method consists of introducing a weight or score expressing the importance of the resources, their attributes and the values taken by these attributes. The scores are obtained through statistical analysis of the requests for virtual network resources. Previous methods treat resources equally and experience longer delays and lower efficiency during the matching process. They were originally not designed for virtual network provisioning and the proposed method in this work aims at addressing their weaknesses in scalability, in resource representation and incremental updates. To highlight this new method, a functional comparison with these previous techniques is proposed. Evaluation results of the score based clustering and matching algorithm show improved scalability and performance in clustering efficiency and matching delays for virtual network provisioning.**

I. Introduction

The future Internet encompasses the emergence of large heterogeneous network infrastructures composed of an increasing number of fixed and mobile devices. Network Virtualisation has recently been considered as an important enabler to diversify the Internet by creating separate virtual networks running simultaneously on top of shared physical infrastructures [1], [2]. Virtual network provisioning consists in describing, matching, embedding and allocating virtual resources (e.g. virtual wireless access/edge routers, virtual wired/wireless links) offered as a service by Infrastructure providers. This paper focuses on enhancing resource description and clustering to improve the provisioning of virtual networks reported in previous work in [3] to enable infrastructure providers to describe and advertise their virtual resources and offer them as a service to virtual network providers and users. A UML resource description and *Conceptual Clustering* facilitate resource matching by grouping virtual resources with similar concepts, descriptions and properties into clusters. The clusters are organized in a tree structure called a dendrogram (or tree) with a *conceptual descriptions* of each cluster. A similarity based matching algorithm [3] using the dendrogram associates requested virtual nodes with the most similar cluster in the dendrogram to find the best match. This dendrogram relies on concepts to organize virtual resources descriptions. A concept is for instance a *router*, a switch or an optical link. A software *router* is also a concept but a child of the parent concept *router*. Moving down deeper into the tree or dendrogram leads the search closer to resources' attributes and values and incurs the highest search and match delay. The objective of this paper is to reduce this delay by speeding up the matching process by ranking attributes and values of virtual resources according to their importance and frequency of occurrence among all known resources and attributes as advertised by the infrastructure providers. This organization of the resource descriptions according to the virtual resource attributes and values score reflecting their importance and the number of times they appear in the data set can lead to significant improvement in resource matching delays. The most popular resource with the most frequently occurring attributes and attribute values will end up at the top of the tree and minimize matching delay. Another drawback is the cost in building these trees and the scalability of such constructions when the number of virtual resources to classify grows. The alternative proposed in this paper is to classify and build the trees using the attributes and the values of the attributes of these resources. Instead of comparing the resource descriptions to assess closeness or similarity, the classification will be based only on the resource attributes since there will be far less attributes than network resources.

The objective of this paper is hence to extend our previous work to reduce matching cost and select methods that can scale with increasing number of network resources libraries. To achieve these enhancements a number of extensions of the previous work were necessary: 1) specifying and developing the UML based resource description using an XML schema so the solutions can be implemented, 2) defining the entire advertisement process relying on abstraction and filtering process of some attributes so only appropriate information is disclosed to virtual network providers and users and 3) more important of all propose a new clustering algorithm that is more efficient in matching virtual resources and that can scale.

This paper is organized as follows. Section II describes the specification and implementation of a virtual resource description and advertisement framework. Section III proposes a new clustering algorithm to classify resources into a hierarchy of clusters. The implementation and evaluation of the VN resource description framework and clustering algorithms are reported in section IV. Note that the central contributions for this paper is the enhanced clustering that is attribute and attribute value centric as opposed to a conceptual clustering centric approach.

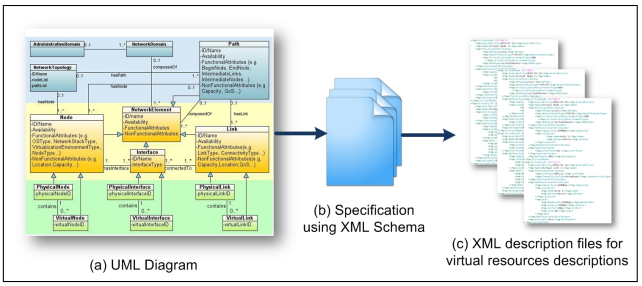

Fig. 1. Specification

II. Virtual Resource Description and Advertisement Framework

In [3], a UML diagram has been provided to express the relationships between virtual resources. This section extends the specification and implementation of this virtual resource description framework to XML schema in order to move from the design phase for which UML is appropriate to XML that enables actual implementation.

The UML diagram in Figure 1(a) gives an overview of objects (Node, Interface, Link...) of the Resource Description Specification and relations between these objects. Details on this diagram can be found in [3]). UML is a modeling language that separates the conception phase from the implementation phase and provides a generic description that is implementation independent. Languages like XML-Schema [4], RDF [5] or OWL [6] can be used to implement the same UML model. In this work, the XML-Schema was selected to implement the specification designed in the UML diagram. With XML-Schema files (Figure 1(b)), it is easy to define precisely the structure and content of XML documents. Another benefit of XML-Schema is to be able to validate any XML file description (Figure 1(c)) and to be sure that any XML file which describes Physical/Virtual network resources respects the specification.

Using well established and normalized XML files ensures standard information exchange between the different actors involved in virtual network provisioning on demand, i.e. infrastructure providers, virtual network providers and users. The XML based resource descriptions can be used by the infrastructure providers to advertise virtual resources and offer them as service to virtual network providers. The infrastructure providers need however to control the amount of information revealed or disclosed to third party. A filtering mechanism is required to limit the amount of advertised information or description. XSLT (eXtensible Stylesheet Language Transformations) is a suitable language for this purpose as it transforms XML documents into other XML documents by filtering part of the information in the original XML file.

These steps are depicted in the generation of virtual resource description files (Fig. 2(A)) that produces the original virtual resource description XML files, generation of mask file (Fig. 2(B)) that determines which attributes and portions of the original description file will be stripped before advertisement and the outcome of filtering (Fig. 2(C)) representing the filtered XML file that will be revealed to the virtual network providers or users.

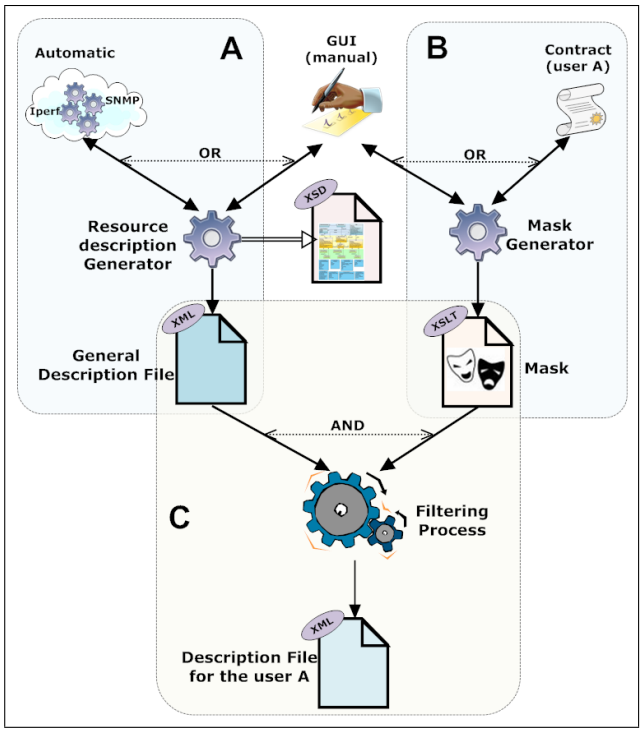

Fig. 2. Network Description Framework

1) Virtual Resource Description (Figure 2(A))
The Resource Description Generator produces XML resource description files by parsing network resource descriptions provided automatically by monitoring systems (e.g. SNMP and Iperf) or manually by end users via a graphical interface. From these inputs the generator outputs a compliant XML files containing the virtual resource description with attributes and values reflecting the state of the network resource. The output file is validated to make sure it respects the specification.
2) Mask (Figure 2(B))
The Mask Generator is used to supply the XSL file with a list of rules either manually through a user graphical interface or automatically from established contracts or service level agreements between actors. Each rule corresponds to a new command that must be put into the XSL file. The set of extracted or defined rules determines the data to hide.
3) Filtered Resource Description (Figure 2(C))
The filtered resource description generator produces the filtered files by taking as inputs the general description file and the mask of user A and applies the rules defined in the mask (XSL file) onto the original description file (XML file). The outcome is a new filtered description file (XML file) that can be sent to user A according to profile and rights.

III. Virtual Resource Clustering

The objective of virtual resource clustering is to facilitate and speed up resources search and match. As mentioned earlier and investigated in previous work [3], virtual network resource need to be classified into clusters grouping resources similar in characteristics, properties and offered services. Clustering of virtual resources need to meet some basic requirements to become flexible, scalable and efficient in allocating, maintaining and adapting virtual networks:

- the number of groups or clusters to create is not known in advance and should be created on need basis and can grow as needed. The classification has to be unsupervised and the clustering algorithm should give rise to as many clusters as needed without any restriction;
- the classification of incomplete resource descriptions into clusters must be possible;
- the clustering algorithm should scale with increasing number of nodes
- the number of network resources should be dynamic with resources that can appear, leave the system and change state anytime. The clustering algorithm should be incremental and dynamic enough to handle this requirement.
- Attributes used for the clustering need not have the same importance and in fact should be annotated with a score reflecting the importance of the information;
- resource data and metadata need to be classified into a conceptual cluster definition for each cluster. This conceptual definition is in conjunctive form, for reasons provided later in the document, that is for example: (Cluster 1 = [attribute 1 = value1.b] & [attribute 3 = value3.h]) because the search is also based on attributes in a conjunctive form.

Clustering has been the subject of extensive research but the available algorithms and techniques are suitable and efficient only in specific cases and scenarios and would not apply in general [7]–[10]. Existing clustering algorithm like COBWEB [11] or CLUSTER/3 [12] fulfil partially the requirements for virtual resource clustering at the level needed for efficient virtual network provisioning. A new clustering algorithm that can scale with network size and that is more suitable for virtual network embedding is proposed and evaluated.

Hierarchical clustering is either agglomerative (bottom-up) or divisive (top-down) [13]. Most hierarchical clustering algorithms use the agglomerative technique because their complexity is $O(n^2)$ versus $O(2^n)$ for the divisive techniques where n is the number of objects to classify. Despite the higher complexity compared to agglomerative techniques, a divisive algorithm is preferred since it leads to faster, simpler and more accurate dendrogram updates. To reduce complexity and make the algorithm more scalable when dealing with a large number of objects, a new measure of similarity indexed by the popularity of attributes is introduced.

For n objects, divisive hierarchical clustering algorithms face $(2^{n-1}-1)$ possible combinations when splitting the set of objects into two subgroups to build a tree representation of the data set. These algorithms assume that all attributes have equal weight or importance. In virtual networks provisioning, requests will typically contain key attributes with some more popular than others such as link and node type. A specific technology type will be less frequently requested for instance in virtualised environments and thus should have less weight. Clustering should take advantage of these differences to answer quickly and efficiently resource requests.

Our solution distinguishes attributes by assigning a coefficient of importance or "weight" to each attribute and their values and add this factor to the similarity function.

In hierarchical clustering, multiple approaches can be used for the similarity criterion like single linkage, average linkage, complete linkage, heuristic measure [11], LEF [12] (Lexicographical Evaluation Functional) or FIHC [14] (Frequent Itemset-based Hierarchical Clustering). For our algorithm and to satisfy the constraints related to the network resource description context, a new similarity criterion combining and extending concepts from all three is proposed. This similarity criterion called **AW-FA** "Attribute Weight and Frequency of Appearance" uses attributes and frequency of occurrence of these attributes and their values.

The similarity criterion relies on the notion of scores on resources attributes and their values as opposed to one to one comparisons between virtual resources descriptions as in COBWEB and CLUSTER/3:

Score of the attribute A_i:

$$S_i = \text{score}(A_i) = [c_i \times H(A_i)] \times \sum_{j=0}^{k_i} (c_{i,j} \times H(a_{i,j}))$$

where :

A_i : i^{th} attribute to use in the clustering process; $i = 1..n$

with:

n : number of objects to classify

m : number of attributes used in the clustering process

O_i : i^{th} object to classify, $i = 1..n$

$O = \{O_1, O_2, ..., O_n\}$: all objects to classify

$A = \{A_1, A_2, ..., A_m\}$: list of attributes to use in classification.

$D(A_i) = \{a_{i,1}, a_{i,2}, ..., a_{i,k_i}\}$: domain of attribute A_i with

$a_{i,j}$: j^{th} value of attribute A_i

k_i : number of attribute values that A_i may have.

$C = \{C_1, C_2, ..., C_m\}$: all weights of attributes used in the process of clustering with C_i represinting the weight of attribute A_i given by $C_i = \{c_i, c_{i,1}, c_{i,2}, ..., c_{i,j}\}$ with:

c_i : the overall coefficient of the attribute A_i

$c_{i,j}$: the coefficient of the value $a_{i,j}$ of the attribute A_i.

The function $H(a)$ calculates the number of occurrences of an entity a in a set of objects to classify.

Among all scores S_i, the algorithm selects the attribute with the best score:

$$S = \max_{i \in 1..m} (S_i)$$

from which to conduct the subdivision of virtual resources into subgroups or clusters to form the subtrees. The outcome of this clustering algorithm results in a tree with more popular

virtual resources attributes located at the top.

The pseudocode of the clustering algorithm is given below:
Clustering_Algorithm() {
If they are available resources to classify then
{

a. for all available virtual resources description, find the attribute with the best score (S) using the AW-FA criteria
b. for this attribute with the best score extract the values $a_{i,j}$ taken by this attribute
c. for each value $a_{i,j}$, run a new thread which execute the function Clustering_Algorithm() for the sub graphs

}
else Stop.
}

The algorithm leads to two main advantages and one disadvantage:

(+) At each tree subdivision stage, there are m combinations, with m is the number of attributes, instead of the $(2^{n-1}-1)$ subdivision combinations required by the other methods. This leads to a significant complexity reduction from $O(2^n)$ to $O(m)$ where n is the number of all virtual resources to classify and m is only the number of virtual resources attributes (typically $m << n$). A large number of virtual network descriptions can be more easily managed and classified.

(+) The output tree of the clustering algorithm is request-oriented. For example, if the majority of queries contains the resource type, this attribute will be at the top of the tree and will be found immediately.

(-) $AW-FA$ results in a less efficient conceptual clustering of the dendrogram because the creation of groups is no longer based on a one to one comparison between objects. Objects with attribute values with higher scores will be close to each other irrespective of the number of attributes they have in common.

We compare our proposed algorithm, named HCC4ND (Hierarchical Conceptual Clustering for Network Description), with two well-known algorithms: COBWEB [11] and CLUSTER/3 [12]. Since the source code of COBWEB and CLUSTER/3 algorithms are not available on line, we were not able to compare the HCC4ND performance results to existing results in the literature. To highlight the advantage of our proposed algorithms, we rather provide functional comparison of our algorithm to existing ones (see table I). COBWEB which is incremental and scalable can add/remove dynamically descriptions of objects but does not provide a conjunctive description for the created clusters because it uses a probabilistic similarity criterion. CLUSTER/3 instead provides conjunctive description for each created cluster but is neither scalable nor sufficiently dynamic. HCC4ND proposed in this work combines the advantages of these algorithms and adds the notion of score to the similarity function to reflect the popularity of virtual network resources, of their attributes and the values taken by these attributes.

IV. Implementation and Performance Evaluation

For performance evaluation both the network description, the advertisement and the clustering frameworks were implemented (about 4500 SLOC - Source lines of code).

A. Network Description and advertisement Framework

The implemented network description generators can be used via APIs (based on JDOM and XQuery) or a GUI (Graphical User Interface). They can (1) Generate a new resource description file (2) Edit a resource description file (3) Generate a new "Mask file" (4) Apply a mask onto a resource description file to create a new filtered resource description file.

B. Clustering Framework

Reading and accessing large XML repositories can be achieved through mainly three well known approaches:

- **Parsing each XML file** : This is the classical simple and intuitive XML access where each XML file is parsed. The entire XML file resides in memory to conduct hierarchical data traversal but this requires unreasonable amounts of memory for large XML files and becomes unpractical when many XML files need to be parsed. Scalability becomes a key weakness.
- **Relying on relational databases** : In this case, XML data is mapped (manually or automatically) to traditional relational database systems and this solves the scalability problem encountered in classical XML file parsing. Nevertheless the inherent flexibility of XML still results, even in this case, in very inefficient mappings with very large numbers of created tables with often many columns with null values, leading to waste of storage space and database global structures that are not normalized. In addition, semantic information is lost during the XML to relational databases mappings due to the XML file structure.
- **Using Native XML databases** : To avoid the interactions between XML and relational databases, it is easier and much more efficient to store the data in native XML; more precisely by using a native XML database that relies on indexation systems. The system and design become less complex and the database administration is significantly simplified. A comparison between native XML and relational databases is provided in [15] and [16]. Among the best-known native XML systems the following appeared to us as the most appropriate: Apache Xindice [17], eXist-db [18], [19] and Berkley DB XML [20]. A performance analysis was conducted and reported for these systems in [21].

To manipulate large XML repositories and to implement the clustering framework, an open source database management system built on XML technology, eXist-db [18], [19], was selected. Figures 3 and 4 highlight the choice of eXist-db (using the Native XML database approach) to manage the XML description files.

Algorithm name	Dynamic add & delete of objects	Conjunctive representation	Similarity criterium	Incremental	Scalability
COBWEB	+	-	probabilistic	+	+
Cluster/3	-	+	LEF	-	-
HCC4ND	+	+	AW-FA	+	+

TABLE I
COMPARISON BETWEEN COBWEB, CLUSTER/3 AND HCC4ND

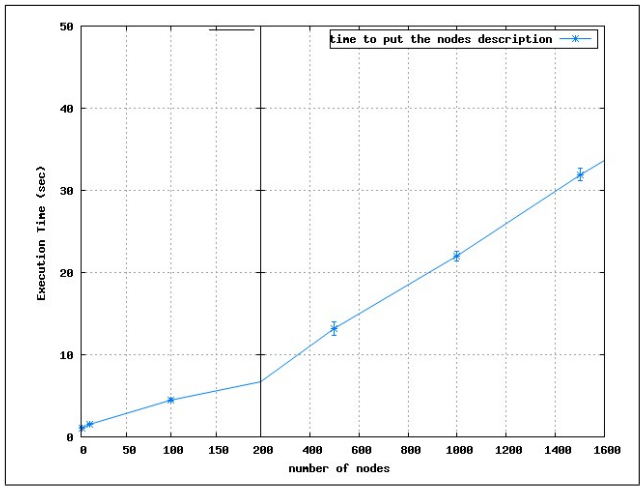

Fig. 3. Average time delay to insert a new description file into eXist-db

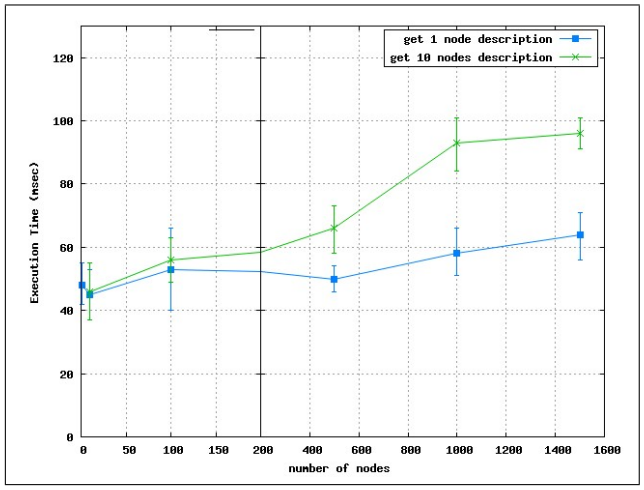

Fig. 4. Average time delay to extract 1 and 10 description files from eXist-db

An overview of the clustering framework steps is described in figure 5: *1)* The first step is to publish the description files. This publication is achieved by adding description files into the eXist-db database. *2)* The second step is running the clustering algorithm. This algorithm requires two entries. The first entry is the description files saved in the Native XML database. The second entry is an XML file that contains the list of attributes to use in the clustering with their weights (coefficients) needed to calculate the AW-FA similarity criterion. The output of the

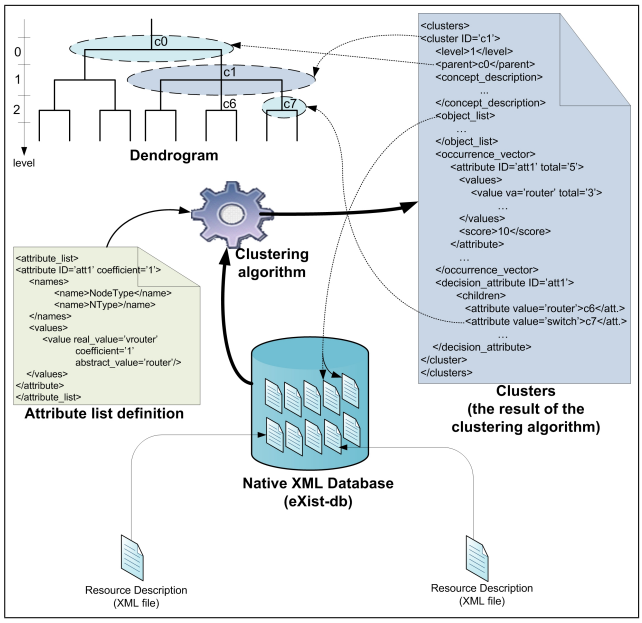

Fig. 5. An overview of the implemented clustering framework

clustering algorithm are XML files representing the dendrogram.

For this proposed clustering algorithm, the main objective is to evaluate the time delay required to find virtual resources that match VN request. An experiment using a dendrogram built with 7 attributes (*VNType*, *NetworkStack*, *Virtualization Environment*, *InterfaceType*, *LinkType*, *ConnectivityType* and *OSType*) was used to evaluates the average time delay required to match five requests with typical complexity. A dual-processor PC (Core2 Duo, 2.16 GHz with 2 Go of RAM) and a Linux based OS (Ubuntu with Java 1.6, eXist-db 1.2 and JDOM 1.1) was used for this experiment. The typical queries are chosen so that the candidate resources correspond to various levels in the dendrogram ranging from higher to deeper levels in the hierarchy (figure 6):

- **Request 1**: [search clusters IDs where *VNType* = *Switch*]. This query contains a single attribute that matches a virtual resource with the highest score and finds a fast match at the top of the dendrogram.
- **Request 2**: [search clusters IDs where *VNType* = *Switch*, *NetworkStack* = *TCP/IP*, *VirtualizationEnv* = *XEN*, *InterfaceType* = *ethernet*, *LinkType* = *VLAN*, *Connectiv-*

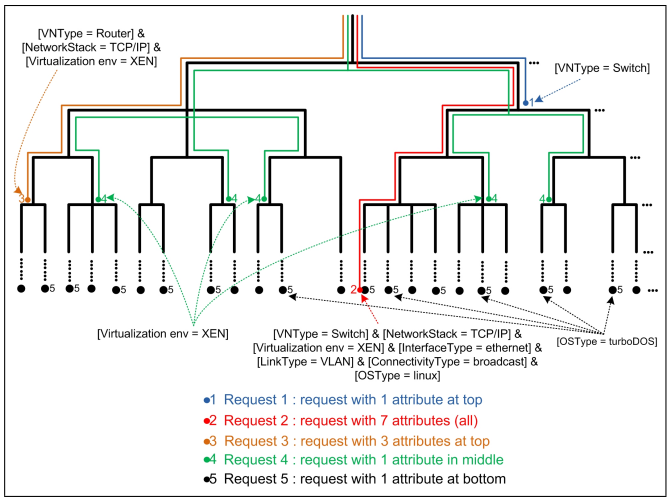

Fig. 6. Different types of requests to run on the dendrogram.

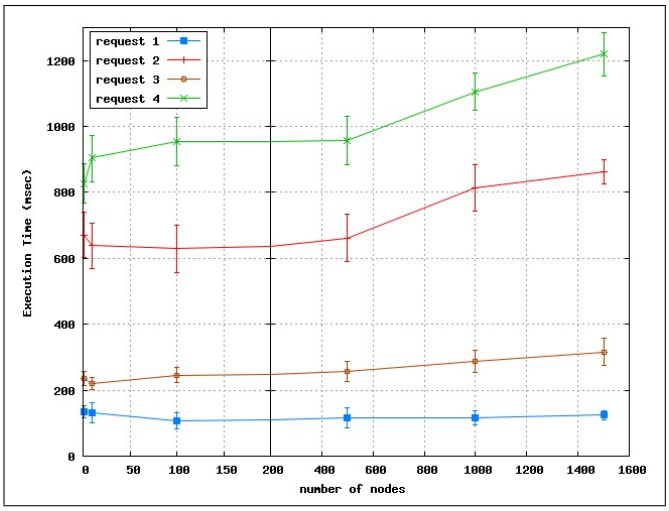

Fig. 7. Average time delay of request 1-4

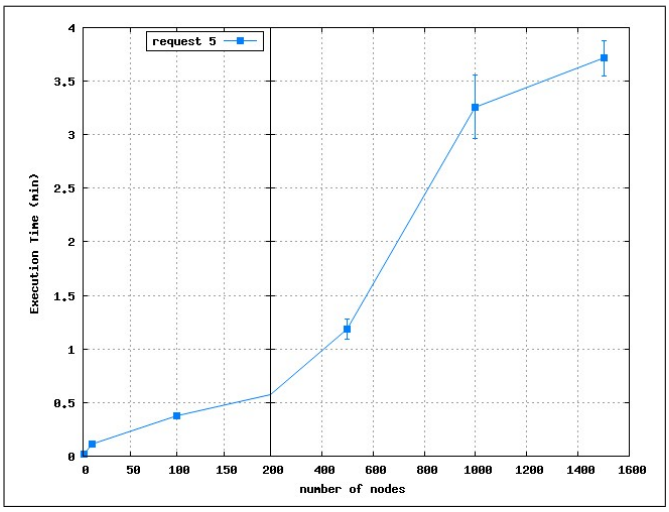

Fig. 8. Average time delay of request 5

ityType = *broadcast* and *OSType* = *linux*]. This query contains all possible attributes so it traverses the entire dendrogram in depth until reaching a leaf.

- **Request 3**: [search clusters IDs when *VNType* = *router*, *NetworkStack* = *TCP/IP* and *VirtualizationEnv* = *XEN*]. This query contains three attributes that correspond to the first top three attributes in the dendrogram. This request is selected to cover the case where the cluster matching the query is located near the middle of the tree.
- **Request 4**: [search clusters IDs when *VirtualizationEnv* = *XEN*]. This type of query contains one attribute located half way into the dendrogram and corresponds to cases where the search occurs in depth and in width in the entire upper half of the dendrogram. This specific query extracts IDs of all clusters that have the value "xen" for the "Virtualization env" attribute in their conceptual description.
- **Request 5**: [search clusters IDs when *OSType* = *TurboDOS*]. This query contains a single rarely used and requested attribute. This scenario corresponds to the antipodal version of request 1 where matching clusters are found at the very bottom of the dendrogram and incur highest search cost.

As depicted in figure 7, the average time delay required to match request 1 is stable at a 110 ms with increasing number of nodes from 1 to 1500. When the matching occurs deeper in the dendrogram (as for the case of requests 2, 3 and 4), the matching delay increases to 1.2 s for 1500 nodes with various penalties depending on scores (popularity and number of attributes). Figure 8 shows that the matching delay of requests of type 5 (where all the dendrogram must be parsed) increases from to 1.2 s to 3.55 min with the number of nodes ranging from 1 to 1500 nodes. The algorithm implementation is based on threads and all the evaluation was conducted using only one PC. If the execution of threads is distributed to multiple PCs, the algorithm performance will be improved especially for the fifth request.

V. Conclusion

A new clustering algorithm is developed and evaluated to classify resources according to their importance and frequency of occurrence to support more efficiently virtual network provisioning for future networks. The clustering algorithms is shown to scale with increasing number of network resources libraries. The algorithm reduces the complexity from $O(2^n)$ to $O(m)$ by focusing on virtual resource attributes (m attributes) and their values rather than one to one similarity assessments between virtual nodes (n nodes with $m << n$).

Future work will consist of extending the virtual resource description to take into account the cloud networking resources and evaluating more thoroughly the clustering algorithm, especially for cloud networking environement.

Acknowledgment

This work has been supported by the IST 7th Framework Programme Integrated Project 4WARD [22], which is partially funded by the Commission of the European Union. The views

expressed in this paper are solely those of the authors and do not necessarily represent the views of Institut Télécom – Télécom SudParis or the respective projects and sponsors.

REFERENCES

[1] N. Niebert, I. Khayat, S. Baucke, R. Keller, R. Rembarz, and J. Sachs *Network Virtualization: A Viable Path Towards the Future Internet*. Wireless Personal Communications, Springer, volime 45, pages 511-520, 2008.

[2] N.M.M.K. Chowdhury and R. Boutaba *Network virtualization: state of the art and research challenges*. Communications Magazine, IEEE, volume 47, number 7, pages 20-26, 2009.

[3] I. Houidi, W. Louati, D. Zeghlache and S. Baucke *Virtual Resource Description and Clustering for Virtual Network Discovery*. Proc. IEEE International Conference on Communications Workshops ICC 2009.

[4] XML Schema. Online site: http://www.w3.org/XML/Schema

[5] RDF - Resource Description Framework.
Online site: http://www.w3.org/RDF/

[6] OWL - Web Ontology Language.
Online site: http://www.w3.org/TR/owl-features/

[7] P. Berkhin *Survey Of Clustering Data Mining Techniques*. Technical report, Accrue Software, San Jose, CA, 2002.

[8] V. Estivill-Castro *Why so many clustering algorithms: A position paper*. SIGKDD. Explorations 4, 6575, 2002.

[9] J. Kleinberg *An impossibility theorem for clustering*. MIT Press, pages 446453, 2002.

[10] R. Xu and D. Wunsch *Survey of clustering algorithms*. Neural Networks, IEEE Transactions on In Neural Networks, IEEE Transactions on, Vol. 16, No. 3. 2005.

[11] D. H. Fisher *Knowledge Acquisition Via Incremental Conceptual Clustering* Mach. Learn., Kluwer Academic Publishers, 1987, 2, 139-172.

[12] W. Seeman and R. S. Michalski *The CLUSTER3 System for Goal-oriented Conceptual Clustering: Method and Preliminary Results* Proceedings of The Data Mining and Information Engineering 2006 Conference, Prague, Czech Republic, July 11-13, 2006.

[13] C. Vrain *Hierarchical Conceptual Clustering in a First Order Representation* ISMIS '96: Proceedings of the 9th International Symposium on Foundations of Intelligent Systems, 643–652, 1996, Springer-Verlag.

[14] B. C.M.Fung, K. Wang, B. C.M, F. Ke and M. Ester *Hierarchical Document Clustering Using Frequent Itemsets* In Proc. SIAM International Conference on Data Mining 2003 (SDM 2003).

[15] G. Gou and R. Chirkova. *Efficiently Querying Large XML Data Repositories: A Survey*. IEEE_J_KDE, 2007.

[16] XML Database Products, Ronald Bourret. Online site:
http://www.rpbourret.com/xml/XMLDatabaseProds.htm

[17] Apache XML Project - Xindice.
Online site: http://xml.apache.org/xindice

[18] eXist-db an Open Source repository and retrieval engine for XML documents. Online site: http://exist.sourceforge.net/

[19] W. Meier. *eXist: An Open Source Native XML Database*. Darmstadt University of Technology.

[20] Oracle Berkeley DB XML - an open source, embeddable XML database. Online site: http://www.oracle.com/database/berkeley-db/xml/index.html

[21] N. Mabanza, J. Chadwick and G.S.V.R. Krishna Rao . *Performance evaluation of Open Source Native XML databases - A Case Study*. Proc. 8th International Conference Advanced Communication TechnologyICACT, 2006.

[22] The FP7 4WARD Project. Online site: http://www.4ward-project.eu