CONTENTS

Knowledge Processing and Discovery

KNOWLEDGE IN

DISTRIBUTED COMPUTING

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 261–264

DISMY – A SEMANTIC GRID SYSTEM BASED ON LINDA, P2P, AND ALCHEMI

ANDREI-GHEORGHE IACOB AND SABIN C. BURAGA

Abstract. By using the Linda coordination model, the P2P paradigm, and existing semantic Web technologies, our proposal – DisMy – aims to be a semantic Grid system, employing the Alchemi Grid as a foundation for a knowledge-based Grid. The paper discusses aspects regarding the design, implementation, and practical deployment of the DisMy system.

1. Introduction

Grid computing [2] enables the sharing, selection, and aggregation of world-wide distributed heterogeneous resources for solving large-scale problems in different areas of interest or for proving access to massive repositories of data, information, or knowledge.

The paper presents DisMy, a proposal of a semantic Grid system based on the peer-to-peer (P2P) technologies and on the Linda coordination model that can be used to have access – in a structured and distributed manner – to resources described by metadata and ontological constructs.

2. The Linda Coordination Model and its Extensions

2.1. General Presentation. The Linda [5] language provides a communication model based on a bulletin board rather than direct messaging, using a shared memory called a *tuple space*. This approach is very useful in the context of Grid computing. As a coordination language, its sole responsibility is the communication and coordination applications developed in host languages (C, C#, or Java).

A Linda system is composed from a set of objects that can basically be of two kinds: *tuples* and *tuple spaces*. A tuple is a collection of fields, each with a certain type (usually borrowed from the host language). Linda specifications do not impose any restriction about the types of the fields inside the tuples, and the tuples can have any size (regarding the number of elements). This fact can extend the definition of a tuple until the point of these having as fields any type of the host language, other tuples, or tuple spaces. A tuple space represents a collection of tuples. Different

2000 *Mathematics Subject Classification.* 68U35, 68N99.
Key words and phrases. Grid computing, Semantic Web, Linda, tuples, Alchemi.

instances of the same tuple can reside in the same space. The communication takes place inside this tuple space, using specific operators [5].

2.2. **Extending Linda.**

For the tuple spaces, our proposed DisMy system uses a view-based approach, regarding the information in the shared memory space: the "data view" and the "fact view". The data are represented as all the tuples along with the classic Linda primitives presented earlier. The facts are the RDF (Resource Description Framework) [1] tuples along with the extended Linda primitives that will be presented further in this section. The fact view, using RDF tuples, maintains information about the tuple space itself. RDF makes possible semantic connections between documents, types of data, or other RDF tuples.

RDF triples can be easily adapted to model a Linda tuple [4]. All the RDF tuples contain fields to represent the basic RDF triple-based model: <subject, predicate, object>. In the DisMy tuple space, the tuples can have XML documents as fields (including RDF/XML), primitive data types or custom classes. The classic matching problem from the Linda model is extended in this implementation to accept these kinds of tuples.

Additionally, DisMy implements several operators that extend the Linda model with respect to the semantic Web [8].

3. DisMy as an Extension of the Alchemi system

Alchemi [6] is an open source Grid system, part of the Gridbus project. The goal of the project is to build an open Grid system using the .NET Framework, interoperable with existing systems. A P2P extension of a Grid has three requirements [7]: to be self-organized, without the help of a central server, to implement a Distributed Hash Table, and to offer support for the resource management based on the level of system usage.

The first requirement is implemented by the Alchemi management console and adapted to the P2P level. The second is represented by the Linda based distributed memory system and the third requirement will be addressed by our proposal.

The P2P model proposed by DisMy is developed on multiple Alchemi systems working in collaboration within a superior level Grid, which transforms the Alchemi Managers into peers.

For example, there are four Alchemi grid systems, each having a number of clients and a single manager. Each manager has a distributed memory space that is seen by its clients as a local memory space. This is exposed to the Grid by a service called P2PA (peer-to-peer agent) which has among its roles the management of this memory space (synchronization between the operations from inside or outside of the Alchemi Grid) and the communication with other P2PAs using Windows P2P technology.

3.1. **The P2PA Service.**

The P2PA is a composition of three services: Grid Resource Manager, Local Resource Manager, and Distributed Data Manager.

The Grid Resource Manager (GRM) has the role of exporting the Alchemi system in the P2P network and its responsible of the global network level coordination of the applications running on the Grid and of the allocated computing resources.

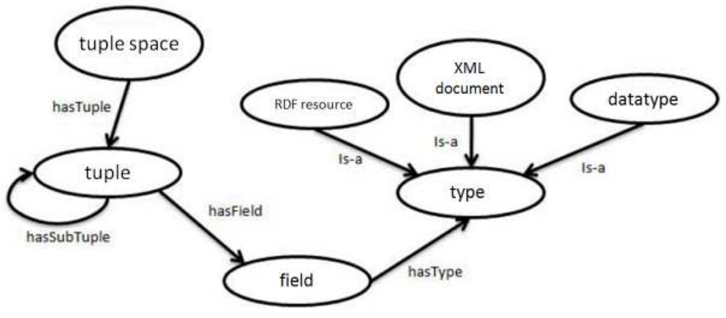


Figure 1. DisMy upper-level ontology

The Local Resource Manager (LRM) extends the base model of the Alchemi Manager using pure inheritance from object-oriented paradigm. The LRM inherits the capability of feeding threads to the Alchemi executors from the parent class.

The Distributed Data Manager (DDM) executes distributed shared memory processes. Its responsibilities are to serve the resource look-up requests and to synchronize the memory locations.

3.2. **Distributed Shared Memory Model.** DisMy defines the following types of fields for tuples: .NET objects, XML content, RDF content, Word, Excel, and PDF files – of course, these types could be further extended.

The difference between these types are at the implementation level, through their corresponding classes which have specific properties. For example, for PDF or Word documents there can be an author, title and any other metadata – denoted in terms of XFiles [3] or other vocabularies [1] – that can help manage them easier in a large scale system.

All the information is shared among the active peers in the network at a specific time using an absolute URI (Uniform Resource Identifier) addressing model. The URI of a resource in DisMy has the following form: `dismy://[host]/[tuple]/.../[tuple]/identification`.

The DisMy name solver helps the user with human friendly name support. For example, if the tuple is named at creation time "reviews", and the field with "paper", the address is: `dismy://PC/reviews/paper`.

Because Linda supports duplicate tuples, two elements with the same name are uniquely denoted by extending the address with another level in nesting, like this: `dismy://PC/reviews/paper/1`. This model can easily be adapted to support the development of a document version control system.

Additionally, we develop an ontology used to properly denote, at the semantic level, the main DisMy entities – see Figure 1.

3.3. **Case Study: Document Management.** To store metadata (in terms of RDF assertions), Linda tuples are used. Each field of a tuple is uniquely identified, thus the indexation could be easily performed. The distributed memory model permits the document storage on any Grid host, with the (automatic or manual) possibility of migration to other computer. Using SPARQL [1] query language, the documents could be retrieved based on metadata. To insert, delete, and edit documents, standard Linda operators are used. Metadata management is performed by using Linda operators, on the basis on existing vocabularies.

Using a desktop/Web interface, the users can apply filters (regarding author, subject, type, etc.) to perform queries within DisMy, in a transparent manner.

4. Conclusion and Further Work

The DisMy project binds four technologies/paradigms (Grid, P2P, semantic Web, Linda) to help solve certain issues regarding resource management on a large scale.

DisMy implements a decentralized P2P topology, using the Alchemi managers to manage the peer connections between executors. DisMy will support a dynamic connection mechanism in which peers will be linked based on parameters like network performance and peer load. The addressing schema would be extended to support meta-tuples.

References

[1] D. Allemang, J. Hendler, *Semantic Web for the Working Ontologist*, Morgan Kaufmann, 2008
[2] F. Berman, G. Fox, T. Hey (Eds.), *Grid Computing*, Wiley, 2003
[3] S. Buraga, *A Model for Accessing Resources of the Distributed File Systems*, LNCS 2326, Springer, 2002
[4] S. Buraga, L. Alboaie, *A Metadata Level for the tuBiG Grid-aware Infrastructure*, SYNASC 2004 Proceedings, Mirton, Timisoara, 2004
[5] D. Gelender, *Multiple Tuple Spaces in Linda*, LNCS 365, Springer, 1989
[6] R. Ranjan et al., *Alchemi: A .NET-based Grid Computing Framework and its Integration into Global Grids*, Technical Report, University of Melbourne, Australia, 2003
[7] R. Ranjan et al., *A Case for Cooperative and Incentive-based Coupling of Distributed Clusters*, Technical Report, University of Melbourne, Australia, 2008
[8] R. Tolksdorf, E. Paslaru Bontas, L. Nixon, *A Conceptual Model for Semantic Web Spaces*, Technical Report D-14195, Freie Universitat Berlin, Germany, 2004

Faculty of Computer Science, "A.I.Cuza" University of Iaşi, Romania
E-mail address: {gheorghe.iacob, busaco}@info.uaic.ro

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 265–267

DISTORTION-BASED MEDIA-FRIENDLY CONGESTION CONTROL

ADRIAN STERCA (1), ZSUZSANNA MARIAN (2), AND ALEXANDRU VANCEA (2)

Abstract. This extended abstract describes a media-friendly congestion control algorithm suited for multimedia streaming in best-effort networks. The transmission rate computed by this algorithm follows the shape of the transmission rate of a TCP-friendly congestion control, but it considers also the distortion it would create in the stream perceived by the receiver. Based on this predicted distortion, the media-friendly congestion control algorithm alters the TCP-friendly transmission rate so that it minimizes this distortion.

1. Introduction

The multimedia traffic in the Internet has increased in the latest years. This increase was boosted by the rise of bittorent and peer-to-peer file sharing applications. Multimedia streaming applications, as opposed to file transfer applications, have great bandwidth demands and strict real-time requirements, demands which do not coexist well with the best-effort nature of the Internet that does not offer any QoS guarantees. Due to these expectations of multimedia streaming, it is of paramount importance for the stability of the network that this type of applications perform congestion control. However, traditional congestion control performed by TCP's AIMD (i.e. Additive Increase Multiplicative Decrease) is not suitable for multimedia streaming because of transmission rate fluctuations incurred and because of delays incurred by retransmissions. TFRC (i.e. TCP-Friendly Rate Control) [1, 2] alleviates to some extent the problems of TCP's AIMD by smoothing out the transmission rate, so that in the long term it has a throughput approximately equal to the throughput of a TCP flow in the same network conditions. The TCP-Friendly Rate Control is a rate-based congestion control that has two main components: the throughput function and the WALI (i.e., Weighted Average Loss Intervals) mechanism for computing the loss rate. The throughput function is the throughput equation of a TCP-Reno source [3]:

$$X(p) = \frac{s}{R\sqrt{\frac{2p}{3}} + t_{RTO}(3\sqrt{\frac{3p}{8}})p(1+32p^2)}, \quad (1)$$

2000 *Mathematics Subject Classification.* 90B18, 68M20.
Key words and phrases. media-friendly congestion control, TCP-friendly congestion control, multimedia streaming.

where X is the sending rate in bytes/sec, s is the packet size, R is the round-trip time (RTT), p is the steady-state loss event rate and $t_{RTO} = 4 * R$ is the TCP retransmit timeout value. This throughput function is behind TCP-friendliness of TFRC. WALI, the mechanism for computing the loss rate as a weighted average of the last 8 loss intervals, is responsible for the smoothness of throughput. However, although smooth congestion controls improve the delivery of multimedia streams, they are not the optimal solution, because they don't take into consideration media characteristics of the stream (i.e. they are not media-friendly).

2. Media-friendly and TCP-friendly Congestion Control

The work presented in this extended abstract builds upon our previous work in media-friendly and TCP-friendly congestion controls for multimedia streaming [4, 5]. More specifically, we are considering the UTFRC (*Utility-driven TCP-Friendly Rate Control*) media-friendly congestion control. By the name UTFRC (*Utility-driven TCP-Friendly Rate Control*) we refer to a congestion control which computes the transmission rate in the following way:

$$X_{UTFRC}(t) = U(q(t)) * X_{TFRC}(t) \tag{2}$$

where t is time, $X_{TFRC}(t)$ is the transmission rate computed by TFRC at time t using Eq. 1, $U(q(t))$ is a utility function (i.e. a media-friendly function) which is increasing with respect to $q(t)$ and $q(t)$ is an n-dimensional function giving the values of various media characteristics over time:

$$q(t) = (m_1, m_2, .., m_n)(t)$$

where t is time and each of $m_1(t), m_2(t), .., m_n(t)$ is a function that measures one media characteristic like bitrate, PSNR value, client buffer fill level etc. The function $U(q)$ embodies the usefulness of increasing TFRC's throughput above the rate computed with Eq. 1 to the streaming application.

3. Distortion-based Media-friendly Congestion Control

The main contribution of this extended abstract is to define a media-friendly function, $U(q(t))$, which includes the signal power of each frame from the video stream. In order to obtain the signal energy (i.e. power) contained in each video frame, we compute off-line for each frame the distortion induced in the perceived stream by not delivering that specific frame. In order to quantify the distortion we use a simple Mean Squared Error (MSE) metric. After we have computed the signal energy contained in each video frame, we compute using these values an average signal energy across the whole video stream. All these computations are done off-line. Then, during streaming, whenever UTFRC updates its transmission rate (i.e. once per RTT or when a loss event is detected, whichever comes first), it uses for the media-friendly function a value greater than 1 if the signal energy of the current streaming second is above average (i.e. the distortion is above average) or a value smaller than 1 if the signal energy of the current streaming second is below average (i.e. the distortion is below average).

This way, the transmission rate of UTFRC will also track the signal energy distribution of the video stream (i.e. UTFRC is media-friendly).

References

[1] S. Floyd, M. Handley, J. Padhye, J. Widmer, *Equation-Based Congestion Control for Unicast Applications*, ACM SIGCOMM 2000.

[2] S. Floyd, M. Handley, J. Padhye, J. Widmer, *TCP Friendly Rate Control*, RFC 3448, January 2003.

[3] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, *Modeling TCP Throughput: A Simple Model and its Empirical Validation*, SIGCOMM Symposium on Communications Architectures and Protocols, Aug. 1998.

[4] A. Sterca, *Congestion Control in Streaming Protocols*, PhD thesis, Babes-Bolyai University, Cluj-Napoca, 2008.

[5] A. Sterca, *UTFRC - Utility-driven TCP-Friendly Rate Control for Multimedia Streams*, in Proc. of the 17th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, IEEE Computer Society, pp. 167-172, Germany, February 2009.

(1) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania
E-mail address: `forest@cs.ubbcluj.ro`

(2) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania
E-mail address: `mzie0590@scs.ubbcluj.ro`

(2) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania
E-mail address: `vancea@cs.ubbcluj.ro`

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 268–271

WEB ANALYTICS FOR EDUCATIONAL CONTENT

SANDA DRAGOS[1] AND RADU DRAGOS [2]

Abstract. Web Analytics helps you to evaluate the performance of your Website. It is a series of techniques used to assess online the behavior of visitors in order to understand and optimize Web usage.

Although there is an abundance of tools performing Web analytics they are usually oriented on business and e-commerce traffic analysis. This paper proposes a Web analytics instrument that assesses traffic on sites with educational content.

The educational content sites (i.e., Google Scholar [2] and CiteSeer [1]) demonstrated their utility through their widespread use. However, like most E-learning systems today, they focus on the technology aspect with apparently lesser efforts spent on developing a system that can be tailored and adapted to individual learners [4].

Paper [7] proposes a theoretical framework of Web analytics that offers a better understanding of online teaching and learning.

1. Web Analytics

Web analytics is the measurement, collection, analysis and reporting of Internet data for purposes of understanding and optimizing Web usage [3].

There are two main technological approaches to collect data for Web analytics instruments. The first method, logfile analysis, reads the logfiles in which the Web server records all its transactions. The second method, page tagging, uses JavaScript on each page to notify a third-party server when a page is rendered by a Web browser. Both collect data that can be processed to produce Web traffic reports.

The Web server log maintains a history of page requests. The World Wide Web Consortium (W3C) maintains a standard format [8] for Web server log files. They contain information about the request, including client IP address, request date/time, page requested, HTTP code, bytes served, user agent and referrer. These files are usually not accessible to general Internet users, only to the Webmaster or other administrative person.

2. Our Proposal

The idea of this new line of research started from two existing instruments: an E-learning tool, called PULSE [5, 6] and a Web analytics instrument. The latter is a

2000 *Mathematics Subject Classification.* 90B18, 91E45.
Key words and phrases. Web Analytics, Web Metrics.

PHP statistics tool that gathers site-usage information into a MySQL database and creates analysis such as yearly, monthly, weekly, daily and hourly statistics on the number of hits, the number of visits, the number of pages and the number of sites. Such overall statistics over three years are depicted in Figure 1. All these results are also given in table format.

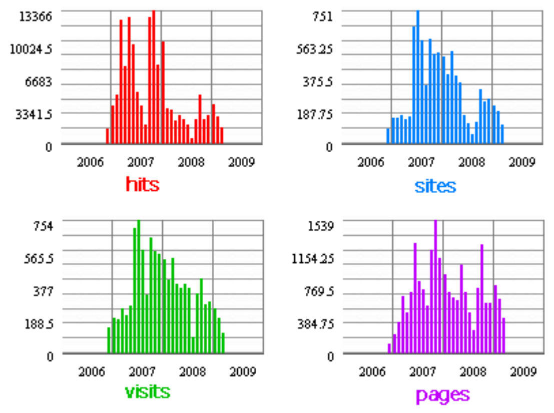
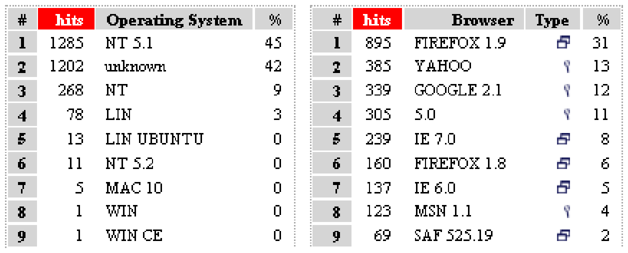


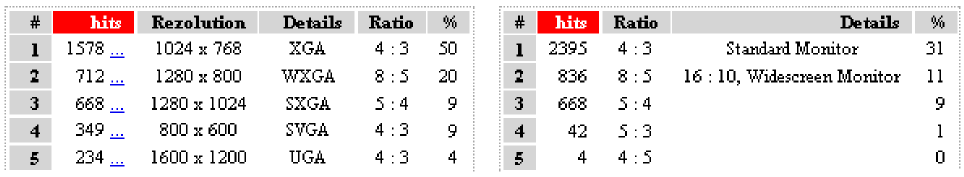
FIGURE 1. Generated Traffic



#	hits	Operating System	%
1	1285	NT 5.1	45
2	1202	unknown	42
3	268	NT	9
4	78	LIN	3
5	13	LIN UBUNTU	0
6	11	NT 5.2	0
7	5	MAC 10	0
8	1	WIN	0
9	1	WIN CE	0

#	hits	Browser	Type	%
1	895	FIREFOX 1.9	🗗	31
2	385	YAHOO	📍	13
3	339	GOOGLE 2.1	📍	12
4	305	5.0	📍	11
5	239	IE 7.0	🗗	8
6	160	FIREFOX 1.8	🗗	6
7	137	IE 6.0	🗗	5
8	123	MSN 1.1	📍	4
9	69	SAF 525.19	🗗	2

(a) The top of Operating Systems and Browsers from the collected User Agents



#	hits	Rezolution	Details	Ratio	%
1	1578 ...	1024 x 768	XGA	4 : 3	50
2	712 ...	1280 x 800	WXGA	8 : 5	20
3	668 ...	1280 x 1024	SXGA	5 : 4	9
4	349 ...	800 x 600	SVGA	4 : 3	9
5	234 ...	1600 x 1200	UGA	4 : 3	4

#	hits	Ratio	Details	%
1	2395	4 : 3	Standard Monitor	31
2	836	8 : 5	16 : 10, Widescreen Monitor	11
3	668	5 : 4		9
4	42	5 : 3		1
5	4	4 : 5		0

(b) The top of resolutions and resolution ratios

FIGURE 2. Sample of Web analytics results

There are also generated lists of:

hostnames that accessed the site: A list is provided with all hostnames corresponding to the host IPs that accessed the site. The list is ordered in a decreasing order by the number of hits from each hostname. There are also provided three more lists with Top Internet Domains (i.e., edu, net, com), second level domains (i.e., ubbcluj.ro, googlebot.com, rdsnet.ro), and third level domains (i.e., search.msn.com, crawl.yahoo.net, staff.ubbcluj.ro). These statistics offer us a view on who is visiting the site in terms of geographic location (e.g., ro, ie, de, it) or the searching engines they used (e.g., googlebot.com, search.msn.com, crawl.yahoo.net).

pages that were accessed: The list of pages offers us a view on the most "interesting" pages on the site.

user agents used: The most important statistics collected from user agents are the operating systems used by the visitors and their browser type and version. A sample of such lists is presented in Figure 2(a). The operating systems can indicate the device used to access the site. For instance "Windows CE" (the last line on Figure 2(a)) is used by minimalistic computers and embedded systems such as personal digital assistants (PDAs) or mobile phones. User agents can also indicate if a visitor arrived at the site through search engines. As presented in Figure 2(a) the largest amount of traffic is coming from search engines such as Google, Yahoo, and MSN.

referrers: Is another way of determining where people are visiting from, as a referrer is the URL of a previous item which led to this request. Referrers can be local pages (a page within the site) or an external page (which can again be a search engine). Statistics on local referrers can help content site optimization by determining which content areas have the most affinity.

resolutions: As the operating system, resolution can indicate the device used.

Out of the three fundamental questions (Who?, What? and Why?) our current Web analytics instrument answers the first two. It determines who is visiting the site and what they are looking for. The most important question remains 'Why?'. We are working to extend this Web analytics tool to answer this last question, more specifically: Why doesn't one student ever visit the site? Why does another student visit it twice a day? Is the material helpful? What material is the most helpful?

From current statistics results that PULSE is helpful as it recorded around 3000 hits within last month. However, our goal is to obtain more meaningful statistics by implementing the two following strategies:

Visitor segmentation: Segmentation isolates the behavior of certain types of online visitors. By using PULSE's log-in phase, individual student's site accesses can be located within collected data. Thus, segmentation can be performed based on demographics such as gender, year of study, line of study, marks.

Despite the fact that each person's learning requirements may be different, there are often wide areas of overlap between individuals that can be mutually beneficial. Similarity in learning needs defining functional communities of learners.

Testing and experimentation: By using slight variations, it is possible to determine which minor differences make the biggest difference. The same content presented in different format (e.g., text versus graphical/multimedia, pdf versus presentation) can have a different impact on students/visitors.

3. Conclusion

There are many ways to monitor user activity beyond the capabilities of a generic statistics package. Most Web analytics instruments are driven by commercial interests, namely, the tracking of online customers behavior. There are some similarities between online customers and online students as they both search for information. However, purchasing is a much simpler act than learning. This paper proposes a framework that extends the existing tool to assess per student behavior on our E-Learning instrument called PULSE.

References

[1] *Citeseer.* http://citeseer.ist.psu.edu/cs.

[2] *Google scholar.* http://scholar.google.com.

[3] J. Burby, A. Brown, and W. S. Committee, *Web Analytics Definitions*, Web Analytics Association, 2300 M Street, Suite 800, Washington DC 20037, August 2007.

[4] P. Desikan, C. DeLong, K. .Beemanapalli, A. .Bose, and J. Srivastava, *Data Mining for E-Learning*, WIT Press, Ashurst Lodge, Ashurst, Southampton, SO40 7AA, UK, 2006, ch. Web Mining For Self Directed E-Learning, pp. 21–40. ISBN: 1-84564-152-3, ISSN: 1742-0172.

[5] S. Dragos, *PULSE - a PHP Utility used in Laboratories for Student Evaluation*, in International Conference on Informatics Education Europe II (IEEII), Thessaloniki, Greece, November 2007, pp. 306–314.

[6] ———, *PULSE Extended*, in The Fourth International Conference on Internet and Web Applications and Services (ICIW), Venice/Mestre, Italy, May 2009.

[7] K. Fansler and R. Riegle, *A model of online instructional design analytics*, in 20th Annual Conference on Distance Learning and Learning, 2004.

[8] P. M. Hallam-Baker and B. Behlendorf, *Extended log file format*, Working Draft WD-logfile-960323, World Wide Web Consortium (W3C).

(1) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania
E-mail address: `sanda@cs.ubbcluj.ro`

(2) Communication Center, Babes-Bolyai University, Cluj-Napoca, Romania
E-mail address: `radu.dragos@ubbcluj.ro`

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 272–275

NODE RANKING IN A DYNAMIC DISTRIBUTED VIDEO PROXY-CACHING SYSTEM

CLAUDIU COBÂRZAN[1]

ABSTRACT. A new ranking mechanism for the nodes that form a distributed video proxy-caching system is introduced. This mechanism is intended to help regulate the number of active nodes depending on a number of conditions (client request volume, available computing and storage resources etc.) while considering and differentiating between data served from different sources (local repository, active siblings, origin servers).

1. Introduction

The volume increase of multimedia data on the Internet together with its continuous growing popularity has lead to several approaches which aim both at ensuring easy access for clients demanding such data and the intelligent use of available resources. One such approach is to deploy one or more proxy-caches which ensures well known and desirable benefits: increased data availability, reduced latency, reduced bandwidth consumption.

The solution we propose is based on multiple proxy-caches which interact in a local area network. Various operations within such a system require a **ranking** mechanisms for the participating nodes. The current paper makes an overview of the system proposed in [1] and further developed in [2], presents its components and refines the ranking mechanism initially introduced in [2].

2. Overview of the Proxy-Caching System

In [1] and [2] we have introduced a video proxy-caching system which starts from a single proxy-caching node but later on can add/remove caching nodes depending on a number of factors like the volume and frequency of client requests, the network conditions and available computing and storage resources.

It makes use of two entities: the `dispatcher` which runs on the proxy-caching node(s) and the `daemon` which runs on the rest of the nodes within the LAN - the ones which could be used for hosting one new proxy-cache sometime in the future. Whenever necessary (see [1]), the `dispatcher(s)` use the running `daemon(s)` to start a new proxy-cache, by transferring needed code and possibly data. Besides its role

2000 *Mathematics Subject Classification.* 68-06, 68M14.
Key words and phrases. video proxy-cache, ranking.

in the expansion of the system, the `daemons` act as a second level on-node proxy-cache which forwards local client requests to one of the running, `dispatcher` served, proxy-caches.

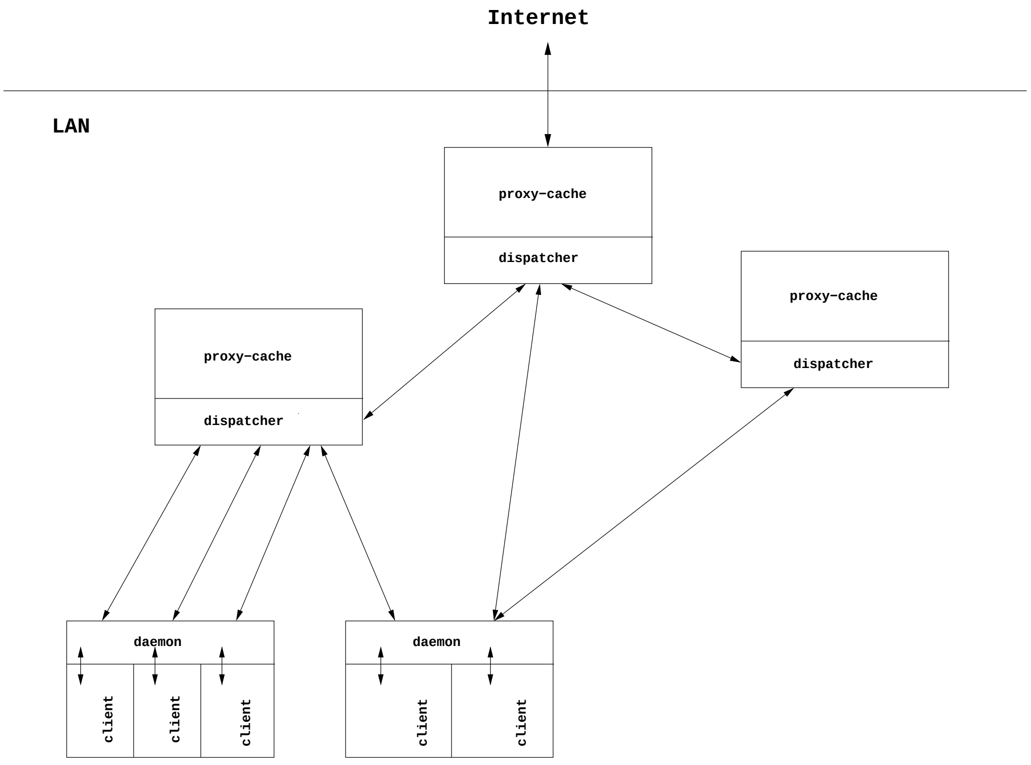


FIGURE 1. The proxy-caching system: `dispatchers` and `daemons` cooperating

The system was further refined in [4] and [3] by using genetic algorithms for deciding the values of the coefficients used when computing the *utility* values of the cached objects. Those coefficients are used to weight different characteristics of the stored data like the size, the number of requests, the moment of the last request etc. The utility values are extremely important since they are used when deciding which objects get discarded, replicated or moved within the system.

The goal was to determine the coefficients in such a manner that the overall *byte-hit ratio* of the systems is maximal.

3. Node Ranking

In [2] we have introduced a **ranking** mechanism for the caching nodes participating in a system like the one described in Section 2. This is intended to play an important part in the dynamics of the proposed distributed proxy-cache, since nodes are leaving the system (through the *hibernate* or *shut down* operations - see [1] and [2]) according to their rank. When the existing clients can be serviced by fewer caching nodes then the ones active, the node(s) with the smallest rank should be put in hibernation. The ranking of those nodes continues until they are shut down, so that in situations when a new proxy-caching node is needed (as a result of a *split* operation), the hibernating node(s) with the largest rank could become active again.

The idea behind the ranking system in [2] is to provide a way of rewarding the nodes that have served the largest amount of data while also considering the moment in time when the last request has been serviced. We intend to refine this mechanism at node level, more exactly we want to distinguish between data coming from cache hits, siblings hits (requests serviced by other participating caching nodes) and server delivered data. This is because the system's performance equally depends on each caching node's performance but also on the degree on which they interact with one another and with data providing servers.

The *rank* of a caching node $P_i \in P$ (P - the set of available proxy-caches, $i = 1..k$, $1 \leq k \leq n$, n - the number of nodes in the LAN) is computed as follows:

$$(1) \quad rank(P_i) = \frac{h_{cache} \times CacheSD(P_i)}{CS(P_i) \times (CT - LRT) \times (h_{sib} \times SibSD(P_i) + h_{ser} \times SerSD(P_i))}$$

where:

- h_{cache} = the number of requests served with data from the local repository;
- $CacheSD(P_i)$ = the amount of data, measured in kilobytes, that has been served by the proxy-cache $P_i \in P$ from the local repository;
- $CS(P_i)$ = the maximal amount of data, measured in kilobytes, that can be stored by P_i;
- CT = the current time;
- LRT = the last moment in time when a client's request has been serviced ($CT > LRT$);
- h_{sib} = the number of requests served with data relayed from siblings;
- $SiblingsSD(P_i)$ = the amount of data, measured in kilobytes, the proxy-cache $P_i \in P$ has relayed from its siblings;
- h_{ser} = the number of requests served with data relayed from origin servers;
- $ServerSD(P_i)$ = the amount of data, measured in kilobytes, the proxy-cache $P_i \in P$ has relayed from origin servers.

The formula is intended to assign the largest ranking values to nodes serving the largest amount of data from within the local repository when reported to the amount of data served from siblings or origin servers.

We can assume that ranking information is exchanged between active proxy-cache periodically (e.g. when the siblings are sending digests of their cache content).

4. Experiments and evaluation

The evaluation part will focus on the influence of the *hibernate* and *shut down* operations on the overall performance of the system (measured considering *byte-hit* *rate* as metric) when using the ranking formula in [2] as well as the newly proposed one (see (1)).

5. Conclusions and Future Work

The paper introduces a new way of ranking the nodes of a dynamic distributed video proxy-caching system. The idea is to differentiate between served data coming from different sources: local repository, active siblings, origin servers, while also considering the moment in time the last request has been serviced.

References

[1] Claudiu Cobârzan. Dynamic Proxy-Cache Multiplication inside LANs. In *Euro-Par 2005*, volume 3648 of *Lecture Notes in Computer Science*, pages 890–900. Springer, 2005.

[2] Claudiu Cobârzan and László Böszörményi. Further Developments of a Dynamic Distributed Video Proxy-Cache System. In *Proceedings of the 15th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP 2007)*, pages 349–357. IEEE Computer Society, 2007.

[3] Claudiu Cobârzan, Alin Mihăilă, and Cristina Mihăilă. Dynamics of a Utility based Distributed Video Proxy-Cache. In *10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC2008), September 26-29, 2008, Timisoara, Romania*, 2008. (accepted for publication in IEEE post-proceedings).

[4] Cristina Mihăilă and Claudiu Cobârzan. Evolutionary approach for multimedia caching. In *19th International Workshop on Database and Expert Systems Applications (DEXA 2008), 1-5 September 2008, Turin, Italy*, pages 531–536. IEEE Computer Society, 2008.

(1) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania

E-mail address: `claudiu@cs.ubbcluj.ro`

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 276–279

BBUFS: ARCHITECTURE OVERVIEW

DAN COJOCAR

Abstract. BBUFs (Babeş Bolyai University File System) is a peer to peer distributed file system designed to span wide areas and provide continuous access to persistent information. The file system is designed for ordinary Unix machines that are IPv6 capable. In this paper we present an architecture overview of the system. A prototype implementation is currently under development.

1. Introduction

We have witnessed during the past decades to a constant growth of information and performance of computing devices. Based on this evolution we are seeing many devices used to control and help different aspects of our day to day life. Weiser [6] in 1991 has envisioned devices that will add intelligence to ordinary objects such as cars, books and even buildings. But before such a revolution can occur, computer devices must become reliable and resilient to failures so that they will be transparent to the average user .

To achieve such a goal we need to persist information, so that when a device is not responding or failing, another device will be able to replace it or handle our request. This way the user will not be disturbed. Also using persistent data we are decoupling the behaviour from the device: the device can be rebooted or replaced and the behaviour is maintained. Maintaining the same behaviour on multiple devices will ensure the reliability and scalability [4].

Peer-to-peer systems and applications are distributed systems without any centralized control or hierarchical organization. All nodes from a system have identical capabilities and responsibilities, and all communication is symmetric. Some of the features that must be considered in a peer-to-peer system are: redundant storage, permanence, selection of nearby servers, anonymity, search, authentication, hierarchical naming, and efficient location of data items.

BBUFs is a peer to peer system that is using IPv6 as a network layer to solve the persisting framework, that will be used on various devices.

The rest of the paper is structured as follows. In Section 2 we briefly present an overview of the system, and in Section 3 we present some possible applications that may use our implementation. More details about the system architecture are

2000 *Mathematics Subject Classification.* 68M14, 90B18.
Key words and phrases. peer to peer file system, IPV6.

presented in Section 4. In Section 5 we are presenting the status of our prototype. Some conclusions and further work are given in Section 6.

2. System Overview

In this section we provide a brief overview of the decisions that we made when we started to implement the prototype. Details on the individual components are presented in Section 4.

Our fundamental unit is *BBUMeta*, all our objects are instances of classes that are derived from *BBUMeta* (as shown in Figure 1) [2]. More details about our internal objects are present in Section 4.

In BBUFs each shared directory is stored and replicated on multiple nodes. The replication process provides some advantages like:

- *availability* - with each replica of a directory the chances to loose a file are decreasing.
- *scalability* - we are able to serve more clients concurrently with more replicas.
- *fault tolerant* - the system will tolerate the failures that a device could encounter.

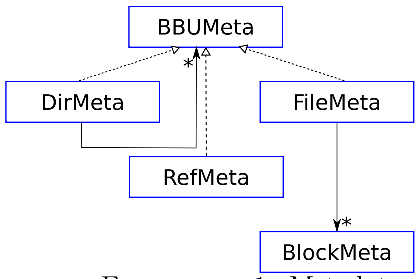


Figure 1. Metadata class diagram.

3. Applications

In this section we present some classes of applications that we think that could be build easily using the BBUFs system and that will have more advantages instead of building them in isolation.

One class of such applications are the medical applications where the patient is visiting different healthcare institutions in different places. A patient medical record will be a shared resource among them. Medical personnel can view and even modify his record concurrently. Even more, the doctors need to see the modifications in realtime, or almost realtime. Using the update and background synchronization mechanism offered by BBUFs many of these problems are already solved.

Another class of applications that will scale are the applications in advertising. Here we need to provide information to a specific target, in BBUFs language for some specific subnetwork. And the information updates should propagate in each location. Using BBUFs we can request that a particular shared content to be replicated only on specific networks.

Finally a backup application that will use BBUFs will easily keep track of files history because when a file stored, BBUFs will create a new version of that file for each update operation. The application only has to implement an interface for requesting files by date or by version.

4. System Architecture

In this section we describe the technologies that we are using to support the BBUFs file system. At the base of our system we have the *BBUMeta*, the superclass of our objects.

BBUMeta, as presented in Figure 1, is the base class for:

- *DirMeta* - the metadata object representing directories that are shared by our file system.
- *FileMeta* - metadata for representing a file that is shared by our system.
- *RefMeta* - a reference to an entry that is stored on another node.

BBUMeta and all derived objects are containing at least the following information:

- *name* - the name of the represented object.
- *n* - the minimum number of desired replicas that the system will try to maintain.
- *hash* - the SHA1 hash value [3] of the shared object.

Using these type objects we are building the following mechanisms: lookup, replication, and versioning.

4.1. Lookup.

One of the main contributions of BBUFs system is a new lookup mechanism based on IPV6 anycast addressing scheme. Using the IPV6 addressing scheme approach we eliminate the need of building and maintaining an overlay network structure found in all peer to peer implementations based on Chord or DHash [5]. As a consequence of using IPV6 addressing scheme we gain the following benefits:

- there is no need to duplicate the routing logic;
- when a new node joins the system we do not have to create data migration logic (like in DHash);
- the addressing space of IPV6 is considerably larger then IPV4 based implementations;
- using anycast and multicast, the lookup is preformed only on a restricted set of nodes.

Using the mapping algorithm *BBUFsMapper* [1] we are binding the directory name to an IPV6 address. Applying this mapper a client can easily obtain the host address where is located one of the replicas of the requested content.

4.2. Replication.

BBUFs *SyncDaemon* is a program that is responsible for maintaining data synchronized between nodes [2]. Periodically *SyncDaemon* performs the following tasks:

- Contact all the nodes in its *multicast group* to check for replicas health.
- Call health check routines to verify local shared data.
- It determines when to create a new replica.
- It initiates a synchronization process when it spots differences of content.

Using the *multicast group*, defined in IPv6, we are able to "talk" only with the nodes that are sharing the same content. This way we are not overloading our system with broadcast messages that recipient nodes are not interested.

4.3. **Versioning.** Using the *version* field that appear in each persistent object that our system is handling we are able to track each modification.

The *SyncDaemon* also notifies the change to other nodes that are in the same group as the old file, in order to perform update.

5. Prototype Status

Currently we are implementing a prototype that will serve for tests and evaluation. The system is written in Java, and is running on Unix machines that are IPv6 capable.

We have implemented the *lookup* mechanism and the *SyncDaemon.* The result of the tests that we have made so far have validated our proposal.

6. Conclusions and Further Work

In this paper we have presented the architecture of BBUFs peer to peer distributed file system. We have described the most important mechanisms of our prototype: lookup, replication and versioning. Further work still have to be done in the following directions:

- To compare the performance of our system to other implementations.
- To deal with security concerns that appear in this type of systems.
- To formalize the presented mechanisms.

References

[1] Dan Cojocar. BBUFs: A new lookup mechanism based on IPV6. In *Workshosp on Global Computing Models and Technologies, co-located with SYNASC 2008*, pages 45–48, 2008.

[2] Dan Cojocar. BBUFs: Synchronization mechanism. In *6th International Conference of Applied Mathematics (ICAM)*, page 46, 2008.

[3] Donald E. Eastlake and Paul E. Jones. Us secure hash algorithm 1 (sha1), 2001.

[4] Donald A. Norman. *The invisible computer.* MIT Press, Cambridge, MA, USA, 1998.

[5] Ion Stoica, Robert Morris, David Karger, Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable Peer-To-Peer lookup service for internet applications. In *Proceedings of the 2001 ACM SIGCOMM Conference*, pages 149–160, 2001.

[6] Mark Weiser. The computer for the 21st century. *SIGMOBILE Mob. Comput. Commun. Rev.*, 3(3):3–11, 1999.

Department of Computer Science, Babeş Bolyai University, 1 M. Kogălniceanu St., 400084 Cluj-Napoca, Romania

E-mail address: `dan@cs.ubbcluj.ro`

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 280–283

CONCEPT PAPER: GENERATING AND ASSESSING TEST PAPERS COMPLEXITY USING PREDICTIONS IN EVOLUTIONARY ALGORITHMS

TIBERIU BAN [(1)]

1. Introduction

The aim of this paper is to present the theoretical approach to assess the complexity of a given test paper. The rationale behind this is that there are certain association rules that can be discovered between task types, acting in such way that if a student is likely to mistake one of the tasks, then we can say with a computable confidence level that the student will be very likely to mistake the other tasks from the same rule as well.

At the moment experimental data is still gathered, in order to be able to start association rule discovery to support this theoretical approach. While completing this step might take some time if data is to be gathered from regular paper-based testing used both in high schools and in universities, an online e-learning platform is in the process of being designed and implemented that would offer the possibility to give online tests with the results already stored in the database and at the point where sufficient data exists for each individual student, this platform will be able to predict each students testing behavior, in order to prepare item tasks targeting the syllabus area where the student needs attention in order to improve his testing performance.

2. Data Description

This paper briefly presents the data model and the adapted way each test paper is stored as a transaction in the database. After this step is complete, an association rule discovery algorithm can be applied, in this case a version of APriori with modified formulae for the support and confidence level. The next step is to define the complexity level of a test paper as a positive value relative to the maximum score. The last step is to use an evolutionary algorithm in order to generate test papers (as collection of task types) that will be as close as possible to a given complexity level.

Lets define a test paper as a set of tasks that can either be correctly solved or incorrectly solved. Lets call a task incorrectly solved as a mistake. A mistake will be noted with 0 points and a task correctly solved will be noted with 1 point.

2000 *Mathematics Subject Classification.* 68-01, 68T05 .
Key words and phrases. association rules discovery, APriori algorithm, Evolutionary Algorithm, test paper complexity, data mining.

The main hypothesis this paper is acting upon is that the valid association between mistakes for various tasks exists not because of the tasks themselves, but rather due to the association of mistakes between the units of information those particular tasks are assessing. In other words its not the particular task we are interested in, but the sub unit of the syllabus that task is assessing. Therefore we will only gather information about the learning unit a particular task is referring to and not the task itself.

3. Knowledge Discovery Process

In order to discover Association Rules between mistakes belonging to a specific test paper the APriori algorithm can be used, since all its perquisites are met [2]. Also, the standard definition for the indicators of support and confidence, the way they are presented in [3] and refined in [1] will be used. Each test paper can be counted as one transaction, because mistaken tasks are relevant one to the other only if all these tasks belong to the same test paper. For this reason, we can consider that recording the mistakes for a given test paper complies with the ACID rule of transactions: atomicity, consistent, isolation and durability, according to their first specifications in [5].

However, because of the nature of the problem itself, given the fact that not all task types are present in test papers for which data has been already gathered, there might be insufficient data to predict strong association rules for item subsets that havent all been present in one single test paper. Still, using association rules already discovered, these can be applied to some extent in order to assess the complexity of a test paper that has not been presented to the students yet.

Implementing APriori algorithm follows the two major steps, as presented in both [1] and [4] generation of item subsets with enough support and extracting association rules from generated subsets. The minimum support level is calculated with respect to the rationale that only the test papers that contained all task types present in the given subset should be counted toward the total number of transactions, multiplied by the number of students that sat for that test, actually the total number of answer sheets that contained all the task types specified in the given item subset. This step requires additional passes through the database and needs improvement in the future.

4. Complexity computation

After having discovered enough association rules that pass the selected threshold of confidence, there is enough data to compute a value of the complexity for any given test paper. This value will be described in this section and will be used in order to assess the number of tasks that are likely to be mistaken by a single student, to the largest extent. In other words, in a worst case scenario, how many mistaken tasks is one generic student going to have in that particular test paper.

This value will be determined by searching for the largest subset of items (tasks) that are part of an association rule, with respect to the rationale that should a student mistake one of the items (tasks) from that items subset, the student is likely (with a probability equal to the confidence level of the association rule used) to mistake all the other tasks from the given subset. Moreover, if the tasks can be divided in several sets, each set being subject to an association rule, then each of these association rules can act independent.

In general, if the total number of tasks NR from a test paper can be divided in nr disjoint item subsets $(S_1, S_2, \ldots S_{nr})$, having $(n_1, n_2, \ldots, n_k)$ items each, each item subsets being subject to an association rule with the respective confidence level $(conf_1, conf_2, \ldots, conf_{nr})$, the average percent of points lost by students (from mistaking all these tasks) will be:

$$Complexity = \frac{\sum_{i=1}^{nr} n_i \cdot conf_i}{NR}$$

Should several ways of dividing the items (tasks) into disjoint item subsets exist and can be singled out without too much computation effort, at a theoretical level we could consider the average values computed for Complexity of the test paper.

Later on in the primary data gathering, when there is enough data for a specific student in order to be representative for the students behavior when sitting for a test paper, the above computed level of complexity can be altered. If enough data exists in order to compute the confidence level for the association rules used in computing the complexity of the test paper based only on the transactions recorded from that particulars student test paper behavior, then the above formula can be altered. This is one point for future extensions of this theory.

5. Generating Test Papers through the use of a Evolutionary Algorithm

After being able to assess the complexity level of a test paper using the above stated formula, the next step is to use this in generating test papers that would cover all possible task types (as a final evaluation test paper) with a specified number of tasks NR (therefore the maximum score being NR), and that would have the complexity level as close as possible to a given level Desired_Complexity.

This is easily achieved by using an evolutionary algorithm that would generate a population of candidates over a number of evolution cycles until either a desired complexity level is achieved (by an error of a given threshold), or a specified number of evolution cycles is computed. The fitness function used in order to select the fittest candidate in each passing will be as follows:

$$Fitness(Candidate) = |Desired\_Complexity - Complexity(Candidate)|$$

When deciding upon keeping one candidate C_1 over another candidate C_2 in the next evolution cycle, the fitness function should be Fitness(C_1) < Fitness(C_2). The goal is to minimize the Fitness function as close to zero as possible.

6. Future extensions

The theoretical approach presented in this paper has some future development directions already pointed out through out this paper.

For the sake of simplifying the theoretical model of a test paper, each task that is part of a given test paper was considered being worth 1 point. In real life this is often not the case, so the formula for complexity should be able to be scaled according to the number of points of all tasks that are involved in the association rule.

Another assumption that has been made in order to simplify the theoretical model was that for each test, all the students were present. Most of the times this is not the case for some test papers, therefore the model should be adapted in order to correctly compute the support level for a given item (task type) subset.

A third future development direction can be outlined at the moment where enough data is gathered for each individual student. In that particular case, the complexity of the paper can be computed taking into consideration the confidence level of association rules singled out from the transactions that describe that particular students behavior when sitting for a test paper.

References

[1] Pang-Ning Tan & Michael Steinbach & Vipin Kumar , *Introduction to Data Mining*, Addison Wesley US Edition, 2005

[2] PSotiris Kotsiantis & Dimitris Kanellopoulos, *Association Rules Mining: A Recent Overview*, GESTS International Transactions on Computer Science and Engineering, Vol.32 (1), 2006, pp. 71-82

[3] Rakesh Agrawal & Ramakrishnan Srikant, *Fast Algorithms for Mining Association Rules*, Proceedings of 20th VLDB Conference, Santiago, Chile, 1994

[4] Rakesh Agrawal & Tomasz Imielinski & Arun Swami, *Mining Association Rules between Sets of Items in Large Databases*, Proceedings of the 1993 ACM SIGMOD international conference on Management of data

[5] A. Reuter & T. Haerder, *Principles of Transaction-Oriented Database Recovery*, ACM Computing Surveys (ACSUR) 15 (4), pp. 287-317

(1) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania

E-mail address: tiberiu.ban@gmail.com

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 284–287

BBUFS: REPLICATION STRATEGIES

DAN COJOCAR$^{(1)}$ AND FLORIAN MIRCEA BOIAN$^{(1)}$

Abstract. BBUFs (Babeş Bolyai University File System) is a peer to peer distributed file system designed to provide fault tolerant shared content and quality of service for its clients. The file system is designed for ordinary Unix machines that are IPv6 capable. In this paper we present our proposal for enhancing the replicating process using different strategies.

1. Introduction

Peer to peer file systems are distributed systems without any centralized control or hierarchical organization [2]. In BBUFs each node is independent and is having identical capabilities and responsibilities [10].

Data replication is a technique used primary for the following reasons:

- *redundancy* - we need our system to be fault tolerant.
- *performance* - we need to load balance the requests to our nodes.

Among the advantages that the replication is offering we notice:

- *low latency* - by creating a new replica closer to user the system will provide better response times, resulting in a low latency.
- *low bandwidth* - this is also a direct consequence of our decentralized system, the client is able to talk directly to the node that is serving our content.
- *improved reliability* - by making copies of the content we do not risk of not being able to serve a client, if some of our nodes are failing.

BBUFs *SyncDaemon* [6] is capable of replicating shared content on different nodes. Until now the replication process was pretty simple: based on the desired number of copies that a shared directory was requesting, the *SyncDaemon* triggers a new replica, if the existing number of copies is below the requested number.

Using this technique *SyncDaemon* is ensuring that our system will maintain at least the requested number of copies.

In this paper we present the modifications that we made to our internal object and to the *SyncDaemon* to be able to create replicas for our shared content more efficiently.

2000 *Mathematics Subject Classification.* 68M14, 90B18.
Key words and phrases. peer to peer file system, IPV6, replication.

The rest of the paper is structured as follows. In Section 2 we present some related work. In Section 3 we present our proposal and some advantages and disadvantages. Some conclusions and further work are given in Section 4.

2. Related Work

There are many peer to peer file systems that are replicating data, however most of the work has focused on creating, deleting the replicas [4, 3, 7, 9, 11]. Also there are results for strategies that are optimizing replications for systems that are mapped on network with topologies like: tree and ring etc [1, 8].

In [12], Wu et. al present *proportional share replication*, a heuristic approach, however the algorithm does not guarantee to find optimal placement.

In BBUFs, using the group information that each node is aware [5], and information like: *from where is shared content requested most often* we are able to establish exactly the subnetwork where we should create a new replica.

3. Proposal

Using the information from *BBUMeta* [6], the fundamental object in BBUFs system, we are able to determine how many copies each shared content is requesting to our distributed system to maintain.

In BBUFs for each client request, among other information, the system will store the following information:

- *client location* - the IPV6 Internet address of the client that has made the request.
- *requested content* - the name of the shared content requested by the client.
- *request time* - the time and date of the request.
- *response status* - what was the response status for the specified request.

3.1. Replication based on access counters.

Using the above information the system periodically will update the metadata of our shared content, with information like the following:

- *access no* - the number of successful request that the node been served.
- *last access time* - when was the last time when someone has requested this content.

We have modified our version of *SyncDaemon* to watch over the access counter field: *access no* too. When this field will grow over a configured threshold value, configured by our file system administrator or the node administrator, the daemon will initialize a new replica. Using log information like *client location* will establish the subnetwork from where most of the queries are coming from and will trigger a replica there.

3.2. Replication based on weights.

Our clients or system administrators can enable *Adaptive replication using weights.* If for a *BBUMeta* object we have configured this property, then the system will perform the following steps:

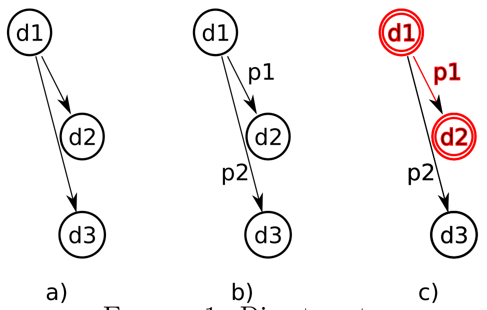


FIGURE 1. Directory tree with weights on branches.

1. For each such shared directory, eg. Figure 1a, using log information we will compute the weight of a branch in our tree, eg. *p1* will be the weight of the directory *d2* in relation to directory *d1*, see Figure 1b and *p2* the weight for *d3*.
2. Using *requested content* and *access no* the system will compute the weight. Periodically the weights are updated based on log history.
3. When the system needs to perform a replica for shared directories like *d1*, using the weights on branches will trigger replicas for shared content like *d2*, see Figure 1c, if for example the *p1* is greater then *0.5*.

Using weights the system can also performs *bulk* replications. Instead of transferring small directories, eg. only *d1* and then *d2*, using the knowledge that *d2* is likely to be requested after *d1* has been requested, the system will pack together the content of *d1* and *d2* and will transfer them in one replication process.

3.3. **Advantages and Disadvantages.** Using strategies like *Replication based on access counters* and *Replication based on weights* our *SyncDaemon* has the following advantages:

- Will be able to create more replicas in the same subnetwork if that shared content is requested by many concurrent clients of that subnetwork. As a consequence of this the system will load balance the queries to more nodes resulting in better response times to our clients and low bandwidth requirements per node.
- When our mobile clients start to access their dates from new locations, the system will trigger replicas in that location, and using weights will replicate all his most used data, resulting in better response times to our mobile clients on further requests.
- Using *bulk* replications will speed the replication process, and the nodes will have more time to serve our clients.

An disadvantage of using the above strategies is that the system will maintain a great number of replicas. The nodes are solving this, because when one node is low on space, using maintenance messages will establish if there are another nodes that have valid replicas. On successful response will perform delete operations on local replicas that are unused for some specified time, here we are using the *last access time* field from our metadata[6].

4. Conclusions and Further Work

In this paper we have presented some strategies that will enhance our system to perform location aware and smart replication using access counters and weights.

Using these strategies our system is able to offer quality of service to our clients and unload our busy nodes. Further work still have to be done in the following directions:

- Evaluate the performance of the replication system using different strategies.
- Compare the performance of our replication system to other existing implementations.
- Enhance the replication using other log information like location history of our mobile clients, etc.

References

[1] Myung M. Bae and Bella Bose. Resource placement in torus-based networks. *IEEE Trans. Comput.*, 46(10):1083–1092, 1997.

[2] Hari Balakrishnan, M. Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica. Looking up data in p2p systems. *Commun. ACM*, 46(2):43–48, 2003.

[3] William H. Bell, David G. Cameron, Luigi Capozza, A. Paul Millar, Kurt Stockinger, and Floriano Zini. Simulation of dynamic grid replication strategies in optorsim. In *GRID '02: Proceedings of the Third International Workshop on Grid Computing*, pages 46–57, London, UK, 2002. Springer-Verlag.

[4] Ann Chervenak, Robert Schuler, Carl Kesselman, Scott Koranda, and Brian Moe. Wide area data replication for scientific collaborations. *Int. J. High Perform. Comput. Netw.*, 5(3):124–134, 2008.

[5] Dan Cojocar. BBUFs: A new lookup mechanism based on IPV6. In *Proceedings of the 10th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2008)*, pages 358–361. IEEE Computer Society Press, 2008.

[6] Dan Cojocar. BBUFs: Synchronization mechanism. In *6th International Conference of Applied Mathematics (ICAM)*, page 46, 2008.

[7] M. M. Deris, J. H. Abawajy, and H. M. Suzuri. An efficient replicated data access approach for large-scale distributed systems. In *CCGRID '04: Proceedings of the 2004 IEEE International Symposium on Cluster Computing and the Grid*, pages 588–594, Washington, DC, USA, 2004. IEEE Computer Society.

[8] Konstantinos Kalpakis, Koustuv Dasgupta, and Ouri Wolfson. Optimal placement of replicas in trees with read, write, and storage costs. *IEEE Trans. Parallel Distrib. Syst.*, 12(6):628–637, 2001.

[9] Houda Lamehamedi, Boleslaw Szymanski, Zujun Shentu, and Ewa Deelman. Data replication strategies in grid environments. In *ICA3PP '02: Proceedings of the Fifth International Conference on Algorithms and Architectures for Parallel Processing*, page 378, Washington, DC, USA, 2002. IEEE Computer Society.

[10] Franjo Plavec and Tomasz Czajkowski. Distributed File Replication System based on FreePastry DHT. Technical report, University of Toronto, Ontario, Canada, 2004.

[11] Kavitha Ranganathan, Adriana Iamnitchi, and Ian Foster. Improving data availability through dynamic model-driven replication in large peer-to-peer communities. In *CCGRID '02: Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*, page 376, Washington, DC, USA, 2002. IEEE Computer Society.

[12] Jan-Jan Wu, Yi-Fang Lin, and Pangfeng Liu. Optimal replica placement in hierarchical data grids with locality assurance. *J. Parallel Distrib. Comput.*, 68(12):1517–1538, 2008.

(1) Department of Computer Science, Babeş Bolyai University, 1 M. Kogălniceanu St., 400084 Cluj-Napoca, Romania

E-mail address: `{dan,florin}@cs.ubbcluj.ro`

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 288–291

NEW DATA MINING TECHNIQUES FOR MACROFLOWS DELIMITATION IN CONGESTION CONTROL MANAGEMENT

DARIUS BUFNEA [(1)]

Abstract. State of the art approaches in Internet congestion control suggest the collaboration between streams in a so called macroflow, instead of the current approach, where streams compete with each other for scarce bandwidth. However, the macroflows granularity follows a simple approach, a macroflow being constructed on host pair bases. This paper presents new data mining techniques for grouping flows into macroflows based on their similar behavior over time.

1. Introduction

We are proposing in this paper a new method for grouping flows into macroflows based on their similar behavior. This paper generalizes and puts in a common template the methods suggested by the author in [2] and in [3], revealing that most state variables maintained inside the TCP/IP stack of a sender can be used in a similar fashion for macroflows identification. Also, we complement from a sender's perspective, the method designed to be implemented inside a receiver stack suggested in [4]. The advantage is a finer macroflow granularity which can be extended to all flows that share the same source LAN and the same destination LAN or even to the flows that share the same network bottleneck.

2. Formal Models

Our model is built around a highly accessed upload server (TCP sender) that maintains continuous data flows towards its clients. The goal is to infer in the incoming connection subsets containing connections having a similar behavior over time. A Congestion Manager running inside the TCP/IP stack of our upload server will treat such an inferred subset as a macroflow. We denote by S the upload server itself or its Internet IP address. Each incoming connection from a client is identified by a pair (C_{IP}:C_{port}) where C_{IP} is the client IP's address and C_{port} is the client used port for the outgoing connection. During a connection life time, server S will periodically measure and store values of some state variables such as the congestion window's size or the round trip time.

Round Trip Time Vectors. From the point of view of the upload server S, the

2000 *Mathematics Subject Classification.* 68M14, 90B18, 90B20.
Key words and phrases. congestion control, macroflow.

incoming connection $f = (C_{IP} : C_{port})$ during the time interval (t_b, t_e) is described by the *Round Trip Time (RTT) vector* $V = (r_1, r_2, \ldots, r_k)$ where: $(t_b, t_e) \subseteq (C_{IP} : C_{port})$ connection's life time; Δt is the interval between two consecutive measurements; $k = (t_e - t_b)/\Delta t$; r_i is the RTT value measured at the $t_b + \Delta t * (i-1)$ time moment. We say that the *RTT vector* associated to a connection describes the connection's behavior. For two connections f_1 and f_2 coming from the same client or LAN the RTT values measured at the same moment in time are quasi-identical. Therefore, their associated *RTT vectors* during the same time interval are also quasi-identical. This means that f_1 and f_2 manifest a similar behavior, which justifies their placement in the same macroflow.

Congestion Window Size Vectors. From the point of view of the upload server S, the incoming connection f=$(C_{IP} : C_{port})$ during the time interval (t_b, t_e) is described by the *Congestion Window Size (CWnd) vector* $V = (r_1, r_2, \ldots, r_k)$ where: $(t_b, t_e) \in (C_{IP} : C_{port})$ connection's life time; Δt is a fixed time interval; $k = (t_e - t_b)/\Delta t$; $r_i = 0$ if the congestion window size decreased at least once during the time interval $T_i = [t_b + \Delta t * (i-1), t_b + \Delta t * i)$, and $r_i = 1$ otherwise (e.g. the congestion window size increased or remained constant during that time interval), $1 \leq i \leq k$. For a connection f, the congestion window size represents its own estimation about the network's available transport capacity. A decrease of the congestion window size occurs when a congestion situation appears along the network path from S towards the destination host. If, during a larger time interval, the congestion window size decreases for two connections f_1 and f_2 in approximately the same time this means that congestion happens for both of them together, in the same moments. So it is very likely that these two connections share a bottleneck. For this reasons, it is justified to place f_1 and f_2 in the same macroflow.

Similarity and Distance Measures in the RTT Vector Space. We associated to a connection an *RTT vector* describing its behavior. The *RTT vector* reflects the RTT temporal evolution of that flow. Two connections will be considered more similar as they are more linearly correlated. A statistical measure for the linear correlation of two vectors is the *Pearson coefficient*. Given two connections, $f_1 = (C^1_{IP} : C^1_{port})$ and $f_2 = (C^2_{IP} : C^2_{port})$ measured during the time interval (t_b, t_e) and their associated *RTT vectors* $V_1 = (r_{11}, r_{12}, \ldots, r_{1k})$ and $V_2 = (r_{21}, r_{22}, \ldots, r_{2k})$, the *Pearson correlation coefficient* of f_1 and f_2 is defined as: $P(V_1, V_2) = \frac{\sum_{i=1}^{k} (r_{1i} - \overline{r_1}) \cdot (r_{2i} - \overline{r_2})}{\sqrt{\left(\sum_{i=1}^{k} (r_{1i} - \overline{r_1})^2\right)\left(\sum_{i=1}^{k} (r_{2i} - \overline{r_2})^2\right)}}$, where $\overline{r_1}$ and $\overline{r_2}$ are the mean values of V_1 and V_2. The similarity measure we use for comparing connections will be: $\overline{P}(V_1, V_2) = \frac{P(V_1, V_2) + 1}{2}$. For differentiating connections the distance function is defined by: $d_P(V_1, V_2) = 1 - \overline{P}(V_1, V_2)$.

Similarity and Distance Measures in the CWnd Vector Space. This section will reveal the distance and the similarity measures used in clustering process in the *CWnd vector* space. A *Cwnd vector* reflects the Cwnd timely evolution of that flow. Two connections will be considered more similar as they meet congestion together more often. We express next the similarity of two given connections, $f_1 = (C^1_{IP} : C^1_{port})$ and $f_2 = (C^2_{IP} : C^2_{port})$ measured during the time interval (t_b, t_e), in terms of their associated *CWnd vectors* $V_1 = (r_{11}, r_{12}, \dots, r_{1k})$ and $V_2 = (r_{21}, r_{22}, \dots, r_{2k})$.

Definition 1. Given a radius *step*, which is an integer number, $0 \leq step \leq k$, and a time interval $T_i = [t_b + \Delta t * (i-1), t_b + \Delta t * i)$, $1 \leq i \leq k$ we call f_1 and f_2:
a) *Congestion Neighbors* on interval T_i iif either: $r_{1i} = r_{2i} = 0$, which means that during T_i both streams faced congestion or $r_{1i} \neq r_{2i}$ and $\exists d \in \{1, 2\}$ so that $r_{di} = 0$ and $\exists j$, $\max\{1, i - step\} \leq j \leq \min\{k, i + step\}$ so that $r_{3-d,j} = 0$.
b) *Congestion Disassociated* on interval T_i iif $r_{1i} \neq r_{2i}$ and $r_{di} = 0$, $d \in \{0, 1\}$ and not $\exists$ j, $\max\{1, i - step\} \leq j \leq \min\{k, i + step\}$ so that $r_{1-d,j} = 0$.

Definition 2. Given a radius *step*, which is an integer number, $0 \leq step << k$ we define for f_1 and f_2 the following sets:
a) $CN(V_1, V_2) = \{i \mid f_1$ and f_2 are Congestion Neighbors on $T_i,\ i = 1..k\}$;
b) $CD(V_1, V_2) = \{i \mid f_1$ and f_2 are Congestion Disassociated on $T_i,\ i = 1..k\}$.

Given a radius *step*, $0 \leq step << k$, the congestion similarity coefficient of f_1 and f_2 is $CS(V_1, V_2) = \begin{cases} \frac{|CN(V_1,V_2)| - |CD(V_1,V_2)|}{|CN(V_1,V_2)| + |CD(V_1,V_2)|}, & \textit{if}\, |CN(V_1, V_2)| + |CD(V_1, V_2)| \geq 0, \\ 0, \textit{otherwise} \end{cases}$. For differentiating connections the congestion distance function is defined by: $d_C(V_1, V_2) = \frac{1 - CS(V_1,V_2)}{2}$.

3. Macroflows identification using clustering techniques

Let $F = \{f_1, f_2, \dots, f_n\}$ be the set of all incoming concurrent connections served by S. For the (t_b, t_e) time interval, the server will take samples of the state variables values that we choose to describe a flow's behavior. Function of the chosen state variables, we will use the corresponding distance and similarity measures. For the (t_b, t_e) time interval, we consider the measured *RTT* or *CWnd* vectors $V = \{V_1, V_2, \dots, V_n\}$, where V_i is the vector associated to the f_i connection, $f_i = (C^i_{IP} : C^i_{port}), V_i = (r_{i1}, r_{i2}, \dots, r_{ik})$, $i = 1..n$. We use an agglomerative hierarchical clustering algorithm [1] for grouping in macroflows the concurrent connections described by similar cwnd vectors. This bottom-up strategy starts by placing each connection in its own cluster (macroflow) and then merges these atomic clusters into larger and larger clusters (macroflows) until a termination condition is satisfied. At each iteration, the closest two clusters (macroflows) are identified. The distance between two clusters M_i and M_j is considered to be the maximum distance of any pair of objects in the cartezian product $M_i \times M_j$. If the distance between these two closest clusters does not exceed a given threshold *thr_max_dist*, we merge them and the algorithm continues by a new iteration. Otherwise, the algorithm stops.

```
Algorithm $MacroflowIdentification$ is:
Input: n, the number of concurrent connection at server S;
      $F = \{f_1, f_2, \dots, f_n\}$ the set of concurrent connection at S;
      $V = \{V_1, V_2, \dots, V_n\}, V_i = (r_{i1}, r_{i2}, \dots, r_{ik})$, $i = 1..n$, the vectors associated to the connections;
      $thr\_max\_dist$, the maximal distance threshold for two connections to be admitted in the same
```

```
macroflow.
Output: m, the number of macroflows inferred in the concurrent connections set;
        $M = \{M_1, \ldots, M_m\}$, the inferred macroflows, where
        $M_i \neq \phi, i = 1..m, \bigcup_{i=1}^{m} M_i = F, M_i \cap M_j = \phi, i, j = 1..m, i \neq j.$
  m := n; $M := \phi$;
  For i:= 1 to m do $M_i := \{f_i\}$; $M := M \cup \{M_i\}$; End For;
   While (m > 1) and ( $Continue$ ($M$, thr_max_dist, $M_{merge1}$, $M_{merge2}$) == true) do
    $M_{new} := M_{merge1} \cup M_{merge2}$;
    $M := M - \{M_{merge1}, M_{merge2}\} \cup \{M_{new}\}$;
    m := m-1;
  End While;
End Algorithm.
Function $Continue$ ($M$ the set of current macroflows, $thr\_max\_dist$, out $M_{merge1}$, out $M_{merge2}$):boolean
is
  $min\_dist := \infty$;
  For each $M_i \in M$
    For each $M_j \in M, M_j \neq M_i$
      $dist(M_i, M_j) = max\{d(v_r, v_t) | f_r \in M_i, f_t \in M_j\}$;
      If $dist(M_i, M_j) < min\_dist$
        $min\_dist := dist(M_i, M_j)$; $M_{merge1} := M_i; M_{merge2} := M_j$;
      End If;
    End For;
  End For;
  If $min\_dist < thr\_max\_dist$ Return True; Else Return False; End If;
End Function.
```

Function *Continue* determines the closest two clusters from the clusters set M. It will return true if these clusters are closer than thr_max_dist and false otherwise. For $d(v_r, v_t)$ we will use either $d_C(v_r, v_t)$ or $d_P(v_r, v_t)$, function of the chosen state variable.

4. Conclusions and Future Work

We suggested in this paper a data model for extending the macroflow granularity outside of the host-pair approach. Our method will prove its advantages in a Congestion Manager framework. As future work we plan to explore the use of different similarity measures and other state variables to compare the timely evolution of the connections being analyzed.

References

[1] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2001.

[2] D. V. Bufnea, A. Câmpan and A. S. Dărăbant. Fine-Grained Macroflow Granularity in Congestion Control Management. *Studia Universitatis*, Vol. L(1), pp. 79-88, 2005.

[3] A. Câmpan and D. V. Bufnea. Delimitation of Macroflows in Congestion Control Management Using Data Mining Techniques. *4th ROEDUNET International Conference, Education/Training and Information/Communication Technologies* - ROEDUNET '05, Romania, pp. 225-234, 2005;

[4] D. V. Bufnea. A New Method for Macroflows Delimitation from a Receiver's Perspective. *Proceedings of the IEEE 2nd International Conference on Computers, Communications & Control (ICCCC 2008)*, Felix Spa, Romania, Vol. III (2008), pp. 201-205.

(1) Faculty of Mathematics and Computer Science, Babes-Bolyai University, 1 M. Kogalniceanu, 400084 Cluj-Napoca, Romania

E-mail address: bufny@cs.ubbcluj.ro

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 292–295

HYPERGRAPHDB – A PEER TO PEER DATABASE SYSTEM

COSTA CIPRIAN[(1)]

Abstract. HypergraphDB [3] is an open source project that uses BerkleyDB in order to implement a hypergraph based database management system. Because of the arbitrary level of complexity supported by hypergraphs, it is a very good approach to manage structure reach information. This paper presents a summary the efforts to implement a distributed version of this DBMS, with accents on the choice of technologies and the availability/consistency guarantees. Most of the implementation was done during my participation to the Google SummerOfCode2008 project [12], where the HGDB project participated as part of the SIAI (Singularity Institute for Artificial Intelligence).

The increased size of the data that needs to be stored and processed is forcing the limits of existing paradigms for databases. Recent developments have proved that alternatives are being searched, even at the cost of not having ACID transactions ([9], [6]).

When faced with large databases, the standard response in industry is to distribute the database on multiple computers and, depending on the access patterns, spread the load across the entire range of databases. Most of the standard industry approaches limit the writes to only one of those databases or use 2-phase-commit to update all the databases at once and make sure that consistency is ensured. However, such approaches quickly become useless when dealing with large amounts of data ([8], [7]), so a more flexible approach is required, with a more relaxed consistency model.

For HypergraphDB we chose a peer-to-peer model that encapsulates horizontal partitioning. Each node of the network will have its own set of data and will collaborate with neighboring peers in order to update its state and the state of its peers. Each peer guarantees that it will make a best effort to synchronize with the other peers and that it will eventually reach a consistent state.

1. Technologies and General system description

For the peer-to-peer network implementation we used JXTA [4] because of its capabilities to hide the physical network topology and virtually make all nodes be at the same distance from one another (for example, JXTA can manage communication between two computers that are both behind a firewall). However, the project is

2000 *Mathematics Subject Classification.* 68P15, 68M14.
Key words and phrases. peer-to-peer, database replication.

not tightly coupled with this technology, so it is relatively easy to replace it with something else.

Another important decision to be made was the style in which the conversations between peers were to be implemented. We opted not to use a client-server approach because this would impose hierarchies that we see as limiting for scaling the application and also raise problems related to availability guarantees. So the all the conversations are based on FIPA (Foundation for Intelligent Physical Agents) standards, especially the communicative act library [2] and the ACL message structure [1] specifications.

2. Algorithms

The general principles the communication is built upon are:

(1) All communication is done asynchronously.
(2) Peers do not control each other, but each peer should be implemented so that it maximizes the output of the system. Workers should actually compete for jobs instead of clients competing for resources.

It is possible for an action to affect atoms on more than one peer and, as such, the problem of consistency is of actuality. Lets call all actions that affect the state of the atoms in a database "write actions". Given the design of our system we have the following requirements for the consistency implementations:

(1) Must be asynchronous. In database terminology we distinguish between eager and lazy propagation of write actions (eager propagates write actions within the scope of the transaction while lazy allows a transaction to commit locally and then the change is propagated in the system). HGDB should use lazy propagation.
(2) Must make a compromise between consistency and assumptions on the delivery order of the messages. We can not rely on the fact that group communication ensures a total ordering on all messages.
(3) Must allow selective replication. That means that only a part of the atoms in the peer database are replicated to another peer (non-disjunctive sets of atoms can be replicated to different peers).
(4) Assume that no peer is designated as holding the primary copy of an atom.

A good paradigm to are the epidemic algorithms for database replication ([5], [10]), but it is important to define what ordering can be imposed on the messages that are exchanged between peers. An important order is the causality relation between transactions (if transaction B was executed after observing transaction A, all peers must execute transaction B after transaction A). While this is important, there are systems that might require a more optimistic approach and might not be willing to pay the price of constant election mechanisms (elections assume independence of peer failures and not all systems have this property). As a first implementation we ignored the causality order guarantee, and go with a very optimistic approach (assume that the probability of the same atom being updated on two peers before the second peer observes the update made by the first is zero). Another important aspect of the implementation is the mechanism by which peers can catch up with modifications after being disconnected. The choice was between a push technique (where updates are constantly pushed to peers until they come back on line) and a pull technique (where the peer queries for updates when it comes back online). It is proven that pull techniques have a higher convergence rate (stale peers get up to date faster) [11]. The solution we chose was to have each peer, at start-up, present its state to the other peers and then receive all updates.

The main issue here is how does peer B, based on the description of the state of peer A, figure out what to send to peer A and in which order. Because peers that have announced their interest in a certain event (an operation on an atom) might not be online and available at the time the event happens, the system must guarantee that the event will eventually reach the peer (we assume down times are not long). Each peer maintains a log of events. The log will impose the ordering of the events (it is important that at all peers events from the same source are applied in the same order). Each event is given a version number (might be a time stamp). The log order is a stronger constraint then the causality relation between two transactions, so using the log order is ok, although not optimal.

One aspect to consider here is that, due to partitions, not all peers will see all events, in other words, if i<k<j, it is possible for a peer to apply event j after event i (if he is not interested in event k). In order for peers to know what events are required in order to apply an event j, at the moment the event is created in the local log, it is also computed what partitions it belongs to and who is the previous event from that partition.

Log records are required until all interested peers have received them and the originating peer knows that they have been received. Each peer will have a matrix T(i, k) = v where i is the current peer, k is a known peer and v is the last event that originated from i and i knows k received it. It is obvious that a log record v is required as long as there is at least one k such that T(i, k)<v. Every time a peer i signals an event to another peer j, it will also send the version number of the event and the version number of the previous event that should be at j; j will decide based on the its knowledge about the last event received from i if it is up to date and can apply the event or not. If yes, j sends a confirm message to peer i which will update his knowledge about the knowledge of j. If not, it will send a request for all the events that were missed by j.

3. Conclusions and future work

By implementing this project we demonstrated a new way to approach database replication and distribution by making the system a peer-to-peer network of database nodes that are exchanging content under low consistency and high availability scenarios. There is still a lot to be done until the distributed HGDB version can be used in real life scenarios, but it is already getting some exposure in projects related to text indexing, where the updates can take longer to propagate.

References

[1] FIPA ACL Message Structure Specification.
http://www.fipa.org /specs/fipa00061/SC00061G.html.

[2] FIPA Communicative Act Library Specification.
http://www.fipa.org /specs/fipa00037/SC00037J.html.

[3] Hypergraphdb. http://www.kobrix.com/hgdb.jsp.

[4] Jxta. https://jxta.dev.java.net/.

[5] D. Agrawal, A. El Abbadi, and R. C. Steinke. Epidemic algorithms in replicated databases (extended abstract). In *PODS '97: Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 161–172, New York, NY, USA, 1997. ACM.

[6] Matthias Brantner, Daniela Florescu, David Graf, Donald Kossmann, and Tim Kraska. Building a database on s3. In *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 251–264, New York, NY, USA, 2008. ACM.

[7] Emmanuel Cecchet, George Candea, and Anastasia Ailamaki. Middleware-based database replication: the gaps between theory and practice. In *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 739–752, New York, NY, USA, 2008. ACM.

[8] Jim Gray, Pat Helland, Patrick O'Neil, and Dennis Shasha. The dangers of replication and a solution. In *SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, pages 173–182, New York, NY, USA, 1996. ACM.

[9] Sanny Gustavsson and Sten F. Andler. Self-stabilization and eventual consistency in replicated real-time databases. In *WOSS '02: Proceedings of the first workshop on Self-healing systems*, pages 105–107, New York, NY, USA, 2002. ACM.

[10] Peter J. Keleher and Ugur Cetintemel. Consistency management in deno. *Mob. Netw. Appl.*, 5(4):299–309, 2000.

[11] Dejan Kostić, Alex C. Snoeren, Amin Vahdat, Ryan Braud, Charles Killian, James W. Anderson, Jeannie Albrecht, Adolfo Rodriguez, and Erik Vandekieft. High-bandwidth data dissemination for large-scale distributed systems. *ACM Trans. Comput. Syst.*, 26(1):1–61, 2008.

[12] OpenCog. Google summer of code 2008 projects. Website. http://opencog.org/wiki/GSoCProjects2008.

$^{(1)}$ Faculty of Mathematics and computer Science, Babes-Bolyai University, Cluj-Napoca, Romania

E-mail address: costa@cs.ubbcluj.ro

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 296–299

MOVING EXCESS DATA INTO EXTERNAL PEER-TO-PEER STORAGE

RARES FLORIN BOIAN$^{(1)}$ AND DAN COJOCAR$^{(2)}$

Abstract. Storage space shortage can be solved either by deleting unneeded files or by adding additional storage. Most of the time, the fastest way to solve the problem is to move data on external storage. This paper continues previous work on solving such situations automatically, transparently, and using as external storage a peer-to-peer file system. The current work addresses performance and data availability issues raised by the costly network activity and the peer-to-peer system lack of guarantees. The proposed solutions rely on the integration of the system with the BBUFs file system for performance improvement and on a predictive algorithm for choosing the "victim" blocks to be moved to external storage (BBUFs).

1. Introduction

Running out of disk space is not a frequent problem, however when it appears it is always a significant one. Even though the cost per gigabyte has decreased dramatically lately, adding more hard–drives to a machine or upgrading to a larger one is not always an option. For home users, such problems are often difficult due to lack of technical knowledge and skill, or simply due to limitations of the cheap home PC hardware. The usual solution to this problem is the purchase of a large capacity external hard–drive where the user can move the data. Even so, today's media files span multiple gigabytes of data, and tend to quickly fill-up the available space. The proposed system offers the user an effort–free almost infinite storage space to deal with the urgent situations when the disk fills up.

The work presented here is the continuation of the system presented by Boian et al. in [1]. The solution offered in [1] uses a peer-to-peer system for storing the data not fitting on the local hard–drive. The peer-to-peer system relies on the Chord algorithm [2] for locating the exported blocks of data. To make the entire system transparent to the user, the implementation was done as a user-level file system using the FUSE library [3].

The current work integrates the existing system with an efficient peer-to-peer file system BBUFs [4, 5] that guarantees O(1) information retrieval. Improvements are

2000 *Mathematics Subject Classification.* 68P20, 68N25, 68M14.
Key words and phrases. distributed file systems, peer-to-peer systems, storage limitation.

also brought to the data block handling by the FUSE file system through the use of a predictive algorithm for choosing the blocks to be uploaded into external storage.

2. Related Work

Peer-to-peer file systems are not a new idea. There are several such systems that offer the users the possibility to store their data in an overlay network with having a centralized server. Examples of such systems are UsenetDHT [6], CFS [7], DHash [8], and IgorFS [9]. These systems store content offered by the user and try to offer a near perfect guarantee for data availability. As discussed by Chun et al. in [10], this results in significant network usage and may be relaxed. Chun offers an alternative solution which insures data *durability* instead of data *availability*. This means that no data will be lost, but there are higher chances that data may not be available when requested.

Our system requires a solution which insures data availability in a very fast manner, which is the reason for choosing BBUFs for external storage.

3. Selecting Data Blocks To Be Exported

The FUSE file system acts as a transparent layer between the operating system and the user, offering our system features through the POSIX file system interface. The logic inside this user–level file system must handle keep the free space on the local hard–drives above a certain level. Whenever the free space falls below this threshold, our system must select blocks of data to upload into BBUFs. The algorithm for this was presented in [1] and was using a simple Least–Recently–Used (LRU) approach, by searching through the files for the oldest access times. This approach is currently improved by taking into consideration the relationship between files for predicting which of them are less likely to be accessed next.

The LRU algorithm states that if a file has not been accessed recently, it is likely that it will not be required in the near future. While this does a good job of selecting candidates for export into BBUFs, it can be improved by adding into the calculation the the processes and the files they access.

3.1. **File Grouping By Accessing Process.** When looking for files to upload to external storage, the algorithm should try to insure that none of the currently accessed or recently accessed files will be exported. It should also insure that none of the files likely to be needed by the running processed should be exported.

The LRU algorithm solves already the first item above. To address the second we need to try to find patterns in the way processes access files. For instance, we can assume that the files opened by a text editor are all related to the same task. This means that whenever one of those files is accessed, it is likely that in the near future the rest of them will be accessed as well. To record this pattern we will consider that all files opened by the same process are related to the same task. All such file will then be marked with the PID of the accessing process. It may seems as a better idea to mark those files with the process path, but in reality, the same program may be used to solve several tasks. Marking the files with the program path, will create false relationships between files. For instance, marking all the document files on the disk with the path to the document editor, will imply that all documents are related to the same task, which is very unlikely. We avoid this by using the process PID which only marks on instance of that program. Since the same file can be accessed by several processes, and hence belong to several tasks, we will store N PIDs for each file, as shown in section 3.2.

When looking for file to be uploaded we will then avoid exporting any files not accessed in a long while (found by the LRU algorithm) if they were accessed by the same PID as recently accessed files.

3.2. **File Access Record.**

Each file stored in the FUSE file system will be attached a record as shown below. Every time a file is accessed, the oldest PID/Date pair int he record will be replaced by the current time and the accessing process' PID.

File path	PID 1	Date 1	PID 2	Date 2	...	PID N	Date N

4. External Storage

As mentioned above, availability is crucial to our system. While durability is also necessary, it is not sufficient. Data to be uploaded is selected automatically by the system, with limited knowledge about its semantics. Exporting a data block from a large data file, which later on becomes unavailable, may result in malfunctioning of the process requesting that block. Such unavailable blocks will result in an error which can only be reported to the accessing process as an I/O error. Such errors generally cause the process to stop. The BBUFs file system offers availability guarantees and O(1) data retrieval, which makes it the best choice for our system.

4.1. **BBUFs.**

The BBUFs file system is a peer-to-peer file system relying on IPV6. The major benefit of BBUFs is the O(1) information retrieval mechanism. Using a check-sum, BBUFs finds directly the address of the machine storing the requested data, and in more than 99% of the cases that data is found there.

BBUFs also offers the possibility to specify the number of replicas to store in the system. This is an important feature that can be used to insure data availability in a flexible manner. Thus, instead of trying to insure 100% availability for all the data blocks exported from the local system, we can use a lower number of replicas when exporting blocks belonging to less crucial files such as video or music.

4.2. **Integration with BBUFs.**

The integration of our system with BBUFs requires the replacement of the existing *UPLOAD* and *DOWNLOAD* commands with the equivalent commands of BBUFs. In addition, it will require the development of an algorithm to determine the number of replicas to store in BBUFs. The number of replicas for a block of data will be calculated based on the following parameters:

(1) Block access count
(2) Type of the file containing the block
(3) The relationship between the replica number and availability percentage

Given the number of replicas, BBUFs will store the data block on that many nodes. In case any of these nodes fail, BBUFs adapt and keep the required number of replicas by creating a new one from one of the nodes still available.

References

[1] F. Boian and R. Boian, "Solving storage limitations using a peer-to-peer web file system," in *Proceedings of the 10thInternational Symposium on Symbolic and Numerical Algorithms for Scientific Computing*, Timisoara, Romania, September 2008.

[2] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan, "Chord: a scalable peer-to-peer lookup service for internet applications," in *Proceedings of the ACM SIGCOMM*, San Diego, USA, 2001.

[3] "File system in user space," http://fuse.sourceforge.net.

[4] D. Cojocar, "BBUFs: A new lookup mechanism based on ipv6," in *Proceedings of the 10thInternational Symposium on Symbolic and Numerical Algorithms for Scientific Computing*, Timisoara, Romania, September 2008.

[5] D.Cojocar, "BBUFs: Synchronization mechanism," in *Proceedings of International Conference on Applied Mathematics*, Baia–Mare, Romania, September 2008.

[6] E. Sit, F. Dabek, and J. Robertson, "UsenetDHT: A low overhead "Usenet" server," in *Proc. of the 3rd International Workshop on Peer-to-Peer Systems*, February 2004.

[7] F. Dabek, M. Kashoek, D. Karger, R. Morris, and I. Stoica, "Wide-area cooperative storage with CFS," in *Proceedings of the ACM Symposium on Operating Systems Principles*, Banff, Canada, October 2001.

[8] J. Cates, "Robust and efficient data management for a distributed hash table," Master's thesis, Massachusetts Institute of Technology, May 2003.

[9] B. Amann, B. Elser, Y. Houri, and T. Fuhrman, "IgorFS: A distributed P2P file system," in *Proceedings of the International Conference on Peer-To-Peer Computing*, Aachen, Germany, September 2008.

[10] B.-G. Chun, F. Dabek, A. Haeberlen, E. Sit, H. Weatherspoon, M. F. Kaashoek, J. Kubiatowicz, and R. Morris, "Efficient replica maintenance for distributed storage systems," in *Proceedings of NSDI06*, June 2006, pp. 45–58.

(1) Department of Computer Science, Babes-Bolyai University, Str. Mihail Kogalniceanu Nr 1, 400084 Cluj-Napoca, Romania

E-mail address: `rares@cs.ubbcluj.ro`

(2) Department of Computer Science, Babes-Bolyai University, Str. Mihail Kogalniceanu Nr 1, 400084 Cluj-Napoca, Romania

E-mail address: `dan@cs.ubbcluj.ro`

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 300–303

A SELF-CONFIGURING MIDDLEWARE FOR DEVELOPING CONTEXT AWARE APPLICATIONS

TUDOR CIOARA[(1)], IONUT ANGHEL, IOAN SALOMIE, MIHAELA DINSOREANU, AND ANCA RARAU

ABSTRACT. This paper introduces a self-configuring middleware that manages the context information acquisition and representation processes targeting the development of context aware applications. The context information is represented using three sets: context resources, actors and policies. An agent based context model management infrastructure generates and administrates the context artifacts at run time. The self - configuring property is enforced by monitoring the real world context in order to detect context variations or conditions for which the context artifacts must be updated.

1. INTRODUCTION

An important challenge in developing context aware applications is the dynamic nature of their execution environment which makes the process of context information acquisition and representation extremely difficult to manage [1]. During the context information acquisition process, the sources of context information (e.g. sensors) can fail or new context information sources may be identified. The context acquisition and representation processes need to be reliable and fault tolerant [4]. A pervasive application cannot wait indefinitely for an answer from a temporary unavailable context resource and many times the payoff for not taking into consideration the new available context resources can be very high. The solution this problems is to use the self-* paradigms to introduce some degree of autonomy for the context acquisition and representation processes [3].

In this paper we define a pervasive self-configuring middleware that uses a context management infrastructure to gather context information from sensors and generate a run-time context representation. The context information is modeled in a programmatic manner using three sets: the context resources set, the context actors set and the context policies set. The context model management infrastructure is implemented by using mobile agents that generate and administrate the context model artifacts at run time. The middleware self-configuring feature is implemented by monitoring and evaluating the environment changes in order to keep the context artifacts updated.

2000 *Mathematics Subject Classification.* 68Q85, 68M14, 68T05, 68W15.
Key words and phrases. Self-Configuring, Context Awareness, Autonomic Computing.

2. The pervasive middleware

The pervasive middleware conceptual architecture (see Figure 1) defines three main layers: (i) the acquisition layer that captures the context information from real world contexts, (ii) the context model layer which represents the context information in a machine interpretable way and (iii) the context model management infrastructure layer that manages the context representation.

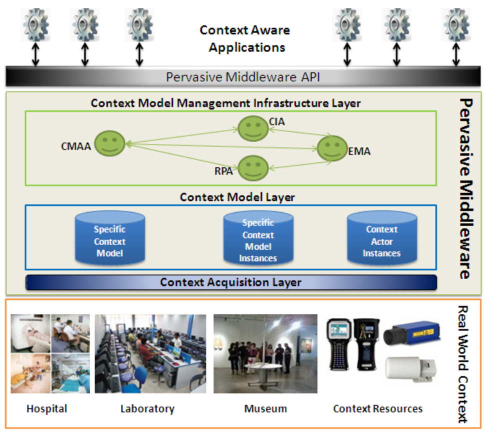


Figure 1. The Pervasive Middleware

The context information acquisition layer design takes into consideration the following aspects: (i) the sensor information retrieval mechanism and (ii) the visibility of the sensor information to middleware upper layers. From the middleware perspective we have defined both push and pull types of sensor information retrieval mechanisms. The push mechanism uses event listeners gather the context information from sensors while the pull mechanism uses a query based approach which allows the context information to be provided on demand.

The context representation layer uses the RAP context model [2] to represent a real world context in a programmatic manner (readable for the pervasive application build on top of the middleware). The context is defined as a triple: $C = <R, A, P>$ where R is the set of context resources that generates and / or processes context information, A is the set of actors which interact with context resources in order to satisfy their needs and P is the set of real world context related policies. In order to provide an accurate representation of the real world context, the following context representation artifacts are defined: specific context model, specific context model instance and context-actor instance. The *specific context model* is obtained by mapping the context model onto different real contexts and populating the sets with real context specific elements. A *specific context model instance* contains the set of context resources with which the middleware interacts, together with their values in a specific moment of time. The *context-actor instance* contains the set of context resources with which the actor can interact, together with their values in a specific moment of time.

The context model management infrastructure layer is based on four types of intelligent, cooperative BDI type agents: Context Model Administering Agents, Context Interpreting Agents, Request Processing Agents and Execution and Monitoring Agents. The *Context Model Administering Agent (CMAA)* is the specific context model manager. Its main goal is the synchronization of the context model specific artifacts with the system execution environment. The *Context Interpreting Agent (CIA)* semantically evaluates the information of a context instance and tries to find the context instance "meaning" for the pervasive application. The *Request Processing Agent (RPA)* processes the actor requests. It identifies and generates the action plans that must be executed for serving an incoming request. The *Execution and Monitoring Agent (EMA)* processes the plans received from the RPA agent and executes every plan action using the available services.

3. The self-configuring feature

At middleware level the self - configuring feature is implemented by monitoring the real world context in order to detect the context variations for which the context artifacts need to be updated and synchronized. We have identified three causes that generate context variation: (1) adding or removing context sources (resources, actors, policies) to/from the real world context, (2) actors' mobility within the real world context and (3) changes of the resources property values.

Context variation generated by adding or removing context elements. During the context information acquisition process, the sources of context information can fail or randomly leave / join the context. These changes generate a context variation that is detected by the context acquisition layer and sent to the CMAA Agent which creates a new specific context model adapted to the new real world context. Next, we evaluate the context variation degree generated by context resources ΔR in relationship with its associated threshold T_R . The same reasoning is used to determine the variation related to the context policies ΔP and the context actors ΔA with their thresholds T_P, and T_A. The context resources set variation ΔR is generated by adding or removing a context resource *r* to / from the pervasive application execution environment. The context resource set variation is calculated using the set difference operation applied in two consecutive moments of time: *t* and *t+1* , where *t+1* represents the moment when the resource *r* became available:

$$\Delta R = \{ R^{t+1} \setminus R^t \} \text{ U } \{R^t \setminus R^{t+1}\} \quad (1)$$

If Card(ΔR) $\geq T_R$ a new specific context model is generated by adding or removing the context resources contained in ΔR. The overall real world context variation ΔENV is given by the union of all context elements' variation:

$$\Delta ENV = \Delta R \text{ U } \Delta A \text{ U } \Delta P \quad (2)$$

The self-configuring threshold is defined as: $T_{Self-Configuring}$= min(T_R, T_A, T_P). The CMMA agent should start the execution of the self-configuring process and generate a new specific context model when Card(ΔENV) $\geq T_{Self-Configuring}$.

Context variation generated by actor's mobility Due to their mobility, the actors are changing their environment location and implicitly the set of resources with which they interact. CMAA identifies this variation and generates (i) a new context-actor instance and (ii) a new specific context model instance. In order to evaluate the context variation generated by actors' mobility we use the isotropic context space concept, defined in [2]. A context space is isotropic if and only if the set of real world context resources is invariant to the actors' movement. Usually, a context space is non-isotropic, but it can be split into a set of disjunctive isotropic context sub-space volumes in which the isotropy degree variation is the empty set. Such a volume is called context granule. The space isotropy variation ΔIZ is non-zero only when an actor a moves between two context granules. If for an actor $\Delta IZ_a \neq \emptyset$, then the self-configuring process executed by the CMMA agent generates a new context-actor instance.

*Context variation generated by changes of resources property values.*A context resource is a physical or virtual entity which generates and / or processes context information. In order to evaluate the context variation generated by the changes in the resource property values, we define a function K_{val} that associates the resource property to its value. CMAA calculates the context variation generated by changes of resource properties' values ΔRPV as presented in (3) and creates a new specific context model instance when Card(ΔRPV) ≥ 0.

$$\Delta RPV = K_{val}(R^{t+1}) - K_{val}(R^{t}) = \{(k_1, val_1{}^{t+1} - val_1{}^{t}), \ldots, (k_n, val_n{}^{t+1} - val_n{}^{t})\} \quad (3)$$

References

[1] I.Salomie A. Rarau, K. Pusztai. Multifacet item based context-aware applications. *Int. Journal of Computing and Information Sciences*, 3(2):10–18, 2006.

[2] I. Anghel M. Dinsoreanu I. Salomie, T. Cioara. Rap - a basic context awareness model. In *4th IEEE Int. Conf. on Intel. Comp Communication and Processing*, 2008.

[3] C. Martel M. Cremene, M. Riveill. Autonomic adaptation based on service-context adequacy determination. *Electronic Notes in Theoretical Computer Science*, 2007.

[4] R. Montanari P. Bellavista, A. Corradi. Mobile computing middleware for location and context-aware internet data services. In *ACM Trans. on Internet Tech.*, volume 6, 2006.

(1) Technical University of Cluj-Napoca, 15 Daicoviciu str, Cluj-Napoca, Romania
E-mail address: `Tudor.Cioara@cs.utcluj.ro`

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 304–307

CHALLENGE-RESPONSE ENTITY AUTHENTICATION TECHNIQUES

HOREA OROS(1) AND FLORIAN M. BOIAN(2)

Abstract. Entity authentication is a process by which one party obtains evidence regarding the identity of a second party involved in a protocol, and that the second party effectively participated to the process, being active when the evidence is obtained or immediately prior to the time the evidence is acquired. This paper presents general techniques used in entity authentication protocols, techniques based on challenge-response mechanism. These techniques use time-variant parameters in the form of random numbers, sequence numbers or timestamps. We investigate challenge-response techniques based on symmetric and asymmetric cryptography.

1. Introduction

The idea in challenge-response protocols is that an entity (the claimant) proves its identity to another entity (the verifier) by providing evidence regarding a secret that is known to be associated to the claimant, without unveiling that secret to the verifier during protocol execution. (In some mechanism the secret is known to the verifier and it is used to verify the answer; in some other mechanism the secret does not have to be known by the verifier). This proof is made by providing an answer to a time variant challenge, answer that is computed using the secret associated with the entity and the challenge. Ussualy, the challenge is a nonce (*n*umber *once*) choosen by one entity (randomly and secretely) at the beginning of protocol execution. If the communication line is monitored by an attacker, the answer from one execution of the protocol does not offer him any information that he may use in subsequent instances of the protocol, because the challenges will differ. Much of the work in the field of authentication-identification protocols was initiated by Needham şi Schroeder [4]. We have investigated these protocols in [6].

2. Time-variant parameters

Time-variant parameters that are used to distinguish an instance of a protocol execution from another are also named *nonce* (*n*umber *once* - numbers that are used only once for one purpose.

2000 *Mathematics Subject Classification.* 94A60, 94A62.
Key words and phrases. authentication, protocol, challenge-response, nonce.

Definition 2.1. *A* nonce *is a numerical value that is used only once for the same purpose. Typically, the role of a nonce is to prevent undetectable replay attacks.*

2.1. **Random numbers.** Challenge-response mechanism may use random numbers, to offer guarantees regarding uniqueness and timeliness, and to prevent some sort of replay and interleaving attacks. Random numbers may be used to offer unpredictability, to prevent chosen plaintext attacks.

2.2. **Sequence numbers.** A sequence number (serial number or counter value) plays the role of a unique message identification number and is usually used to detect message replay. Sequence numbers are specific to a pair of entities and must be associated implicitly or explicitly with the originator and receiver of a message; sequence numbers for messages from A to B will be different from those from B to A.

2.3. **Timestamps.** Timestamps are used to offer timeliness and uniqueness guarantees, in order to be able to detect message replay. Timestamps can be used to implement acces privileges for a limited period of time and detect forced delays.

3. Challenge-response - symmetric techniques

In challenge-response mechanism based on symmetric techniques the claimant and the verifier have to share a key. In closed systems with a small number of users, each pair of users can share a symmetric key; in large systems that use techniques with symmetric keys, the identification protocols involve an online trusted server that shares a long term secret key with each entity. The online server creates/transmits a session key to be shared to a pair of entities that wish to authenticate one to another.

3.1. **Challenge-response based on symmetric key encryption.** The Kerberos protocol Kerberos [3], [5], [7] and Needham-Schroeder protocol[4], offer entity authentication with symmetric key encryption and implies the usage of an online trusted third party. These two protocols offer also key transport.

We describe below three simple techniques based on ISO/IEC 9798-2 [1]. These imply the existence of a secret key (without requiring an online trusted server). In this case two entities may carry out unilateral entity authentication with a single message using timestamps or with two messages if random numbers or sequence numbers are used. Mutual authentication requires an additional message. The claimer proves its identity by demonstrating its knowledge regarding the secret key that he uses to encrypt the challenge (and optionally additional data).

(1) *unilateral authentication using timestamps*:

$$A \to B : E_K(t_A, B^*) \quad (1)$$

(2) *unilateral authentication using random numbers*: to prevent the usage of timestamps one can use random numbers, the cost being an additional message:

$$\begin{array}{ll} A \leftarrow B : r_B & (1) \\ A \to B : E_K(r_B, B^*) & (2) \end{array}$$

(3) *mutual authentication using random numbers*:

$$\begin{array}{ll} A \leftarrow B : r_B & (1) \\ A \rightarrow B : E_K(r_A, r_B, B^*) & (2) \\ A \leftarrow B : E_K(r_B, r_A) & (3) \end{array}$$

3.2. **Challenge-response based on keyed one-way functions.** The encryption algorithm used in the above mechanism may be replaced by a one-way function or a non-reversible function applied to the key and to the challenge. The modifications that are made to the 9798-2 mechanism will result in ISO/IEC 9798-4 mechanism.

The revised challenge-response mechanism with three messages based on MAC function h_K offers mutual authentication. The resulting protocol, SKID3, is due to Bosselaers [2].

4. Challenge-response - asymmetric techniques

In identification based on challenge-response one can use public-key techniques, in which the claimer proves its knowledge about the private key using one of the following modes: the claimer deciphers a challenge encrypted with its public key or the claimer digitally signs a challenge

We have mechanisms that uses only public-key encryption and witness:

$$\begin{array}{ll} A \leftarrow B : h(r), B, P_A(r, B) & (1) \\ A \rightarrow B : r & (2) \end{array}$$

and mechanisms that uses digital signature schemes: X.509 mechanism (the strong authentication protocols ITU-T X.509 with two and three messages specifies identification techniques based on digital signatures with timestamps and digital signatures with challenge-response), 9798-3 mechanisms:

(1) *unilateral authentication with timestamps*:

$$A \rightarrow B : cert_A, t_A, B, S_A(t_A, B) \quad (1)$$

(2) *unilateral authentication with random numbers*: Timestamps may be replaced with random numbers, the cost being an additional message:

$$\begin{array}{ll} A \leftarrow B : r_B & (1) \\ A \rightarrow B : cert_A, r_A, B, S_A(r_A, r_B, B) & (2) \end{array}$$

(3) *mutual authentication with random numbers*:

$$\begin{array}{ll} A \leftarrow B : r_B & (1) \\ A \rightarrow B : cert_A, r_A, B, S_A(r_A, r_B, B) & (2) \\ A \leftarrow B : cert_B, A, S_B(r_B, r_A, A) & (3) \end{array}$$

Conclusion

We have presented and analyzed challenge-response authentication techniques based on symmetric and asymmetric cryptography. Time-variant parameters are essential ingredients of these protocols. These parameters provide security against replay and interleaving attacks.

Authentication with challenge-response based on symmetric techniques can be obtained by using symmetric encryption schemes or keyed hash functions.

Authentication with challenge-response based on asymmetric techniques can be obtained by the claimer proving its knowledge about the private key using one of the following modes: he deciphers a challenge encrypted with its public key or he digitally signs a challenge.

References

[1] ISO/IEC 9798-2. Information technology - security techniques - entity authentication - part 2: Mechanisms using symmetric encipherment algorithms. Technical report, International Organization for Standardization, Geneva, Switzerland, 1994.

[2] A. Bosselaers and B. Preneel, editors. *Integrity Primitives for Secure Information Systems: Final Report of RACE Integrity Primitives Evaluation RIPE-RACE 1040*, volume LNCS 1007, New York, 1995. Springer-Verlag.

[3] J. Kohl and C. Neuman. The kerboros network authentication service, September 1993. Network Working Group Request for Comments: 1510.

[4] R. M. Needham and M. D. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, December 1978.

[5] B. Clifford Neuman and Theodore Ts'o. Kerberos: An authentication service for computer networks. *IEEE Communications Magazine*, 32(9):33–38, September 1994.

[6] Horea Oros. Security protocols for access control. PhD Report, Babeş-Bolyai University, Cluj-Napoca, March 2008.

[7] J. G. Steiner, B. C. Neuman, and J. I. Schiller. Kerberos: an authentication service for open network systems. In *Usenix Conference Proceedings*, pages 191–202, Dallas, Texas, February 1988.

(1) University of Oradea, St. Universitatii No. 1, 410087 Oradea, Romania
E-mail address: `horos@uoradea.ro`

(2) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania
E-mail address: `florin@cs.ubbcluj.ro`

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 308–311

WEB SERVICE COMPOSITION APPROACH BASED ON A GRAPH OF SERVICE CELLS

VIORICA R. CHIFU[(1)], IOAN SALOMIE[(1)], AGOTA RIGER, VALENTIN RADOI, AND DANIELA INOAN

Abstract. This paper proposes a new method for the automatic composition of semantic Web services. The proposed method combines graph construction and sub-graph search to determine all the composition candidates. First, the method constructs a directed acyclic graph, which represents the functional semantics of the Web services as well as the semantic dependency between the inputs and the outputs of the services. Second, composition sub-graphs that satisfy the functionality requested by a client are searched for and ranked according to certain criteria such as user preferences and QoS. Finally, the method selects the best service composition for a given user request. To evaluate the method, an experimental framework that automatically composes services has been implemented.

1. Introduction and Related Work

In the area of automatic Web service composition, we present an approach based on a directed acyclic graph having service cells as nodes. The approach uses the semantic descriptions of the services and the service cell abstractions. The semantic descriptions of the Web services, written in the SAWSDL language[1], contain semantic annotations as references to concepts in a common ontology. The service cell abstraction is specified by a Web service operation along with additional information about the service operation. The proposed method combines graph construction and sub-graph search to determine all the composition solutions. Starting from a given goal node of the graph, the method finds the subgraph of optimal cost between the goal node and the source node, given a set of user preference constraints. The main contributions of our work are: (i) defining a new method for Web service composition based on the service cell graph structure, and (ii) developing an algorithm for the user preference aware automated service compositions. To validate our approach, we have implemented an experimental framework that automatically composes services, and we have demonstrated its effectiveness with the help of an scenario from the trip planning domain. Previous efforts related to the automatic service composition include various approaches, such as AI planning [2, 3]or semantic approaches[4]. The composition based on AI planning considers the service composition problem as a

2000 *Mathematics Subject Classification.* 68P20, 68Q55, 68T30, 68U35, 68M14.
Key words and phrases. semantic Web services, Web service composition, graph of service cells.

planning problem while the composition based on semantic approaches proposes the use of semantics as a key to increase the automation.

2. Semantic Web Service Composition

Our service composition algorithm works upon a pre-processed initial directed acyclic graph, making use of some additional data structures in order to increase the speed and reduce the search space. The initial graph is built by chaining the available services based on semantic matching between their input and output parameters.

Prerequisites of the Composition Algorithm. In order to apply the composition algorithm, a pre-processing step is performed, consisting of the following stages.

Service Cell: For each service operation, we generate a corresponding service cell (SC). An SC is a wrapper of a Web service operation. Each SC has a set of inputs and a set of outputs representing the input and output parameters of the service operation. Additionally, QoS parameters (i.e. availability, execution cost) are taken into consideration and stored into the data structure of the SC.

The *initial graph of Service Cells*: We construct the initial directed acyclic graph of SC's by chaining the available SC's based on semantic matching between their input and output parameters. More exactly, for each input of an SC we generate a set of SC providers. The set of SC providers for an input of an SC represents a set of edges from the outputs of other SC's towards that input. Every edge is established based on semantic matching between an output concept (out_i) of a service cell $SC_i$and an input concept (in_j) of a service cell SC_j. We consider three types of semantic matching: exact $(out_i \equiv in_j)$, plugIn $(out_i \subseteq in_j)$, and subsume $(in_j \subseteq out_i)$, where $\subseteq$ is the subsumption relation between concepts according to the common ontology.

Start Service Cell and *end Service Cell*: We generate two special SC's, *startSC* and *endSC*. The *startSC* is a special SC which has only outputs, representing the inputs requested by the user for the composed Web service, while the *endSC* has only inputs representing the outputs requested by the user for the composed Web service.

Evaluation Service Cell: The user hard constraints are also modeled with a special type of SC, namely *evalSC*. An example of user hard constraint is that the price for a holiday planning should not exceed 1000 Euro. *evalSC* evaluates the current solution for a service composition request and returns an overall score based on the user hard constraints. The score is computed as a weighted sum of the hard constraints specified by the user. This *evalSC* has a single output representing the score of the solution, and a variable number of inputs, representing all the outputs of the SC's participating in that particular composition.

Composition Algorithm. Starting from the initial directed acyclic graph of SC's, the algorithm solves the composition problem in four stages: *preliminary stage*, *search stage*, *simulated execution stage*, and *selection and execution stage*. In the *preliminary stage*, three special SC's are added to the initial graph: *startSC*, *endSC*, and *evalSC*. In the *search stage*, the composition solutions which satisfy the functionality requested by a client are searched for in the graph and ranked according to certain criteria. The search algorithm starts from the *endSC* and tries to find a sub-graph of *SC*'s which leads to the *startSC*. We say that we found a composition solution if all the *SC*'s from the sub-graph have been saturated (i.e. we obtain data for all of their inputs).When a composition solution has been found, all the outputs of the *SC*'s participating in that particular composition should be linked to the *evalSC*, but only some of them (based on user provided hard constrains) will be taken into account in the *simulated execution stage*. The *evalSC* also gathers all the relevant data (e.g. *QoS*) from the *SC*'s involved in that particular composition in order to compute an initial score of the composition. After all the possible composition solutions have been determined, the *simulated execution stage* of the algorithm can start. First, the minimum set of input data which cover the requirements of all the composition possibilities is determined and the user is asked to provide them. By analyzing the outputs of the composition solutions, a relevant set of data fields can also be presented to the user, in order for him/her to set up the hard constrains on these outputs. With these user provided inputs, each solution is executed on the real-world Web services, in forward direction. During the execution, a mapping is done to convert the logical, semantic data labels into primitive data types. Also, a *filtering* is applied to some edges of the graph for satisfying the user soft constraints (for instance, the user prefers, from a list of available flights, only evening flights). In this *simulated execution stage*, not all the services of the composition solution are executed. Some services will be considered locked for execution because they are world-changing services. During the *selection and execution stage*, as real-world data are obtained from the outputs of the *SC*'s along the solution, these data are fed into the *evalSC*. At the end, the *evalSC* can compute an overall composition score based on the initial composition score and on the user hard constraints.

3. Framework Implementation

To validate our approach, we have implemented an framework that automatically composes services, and we have demonstrated its effectiveness with the help of a scenario from the trip planning domain. The framework components are presented below.The *WS Repository* is a repository of semantically annotated services. The semantic descriptions generated for each service are based on the *Ontology*. The *Graph Construction Engine* is the component which is responsible for the construction of the initial graph. The *Semantic Matchmaker* is used by the *Graph Construction Engine* to semantically match the inputs and outputs of the *SC*'s that are chained. The *Graph Storing Module* stores the graph in an XML based language, called Service Graph Description Language (*SGDL*). *SGDL* saves all the relevant information about every node of the graph (i.e. operation of the cell, inputs/outputs, providers, QoS rating).The *Ontology driven GUI* guides the user in the composition process by providing a controlled language that uses the ontology concepts. The *Composition Engine* takes as input the user goal and generates service composition solutions. The *Constraint Analyzer* analyzes each service composition solution by considering the non-functional requirements of the goal, i.e. the user preferences in terms of hard and soft constraints. It selects a Web service composition that meets all the non-functional requirements for the goal. The *Execution Engine* is responsible for executing the services included in the selected service composition solution.

4. Conclusions

In this paper we have presented a novel approach for the automatic Web service composition, based on a graph of SC's. Our composition algorithm is a combination of backward chaining (for finding the composition solutions) and forward chaining (for executing the selected composition solution). The algorithm uses the semantic service descriptions to impose the correctness of the possible solutions. Also the algorithm allows for correcting the solutions when some services become unavailable, by providing alternative solutions. By comparison with other approaches, our composition algorithm can overcome the performance issue by working directly on the preprocessed graph. As such, a given query can achieve a quick response during the process of finding the composition solutions.

References

[1] J. Kopecky, et al. "SAWSDL: Semantic Annotations for WSDL and XML Schema". IEEE Internet Computing, 11(6): p. 60- 67. 2007

[2] E. Sirin, et al., "HTN planning for Web Service composition using SHOP2", Journ. of Web Semantics. 1(4): p. 377-396. 2004

[3] I. Paik, et al., "Automatic Web Services Composition Using Combining HTN and CSP", IC-CIT,Fukushima, Japan, October 2007: IEEE computer society

[4] L. Cabral, et al., IRS-III: A Broker for Semantic Web Services Based Applications, the 5th International Semantic Web Conference. Athens, USA, November 2006: LNCS.

(1) Department of Computer Science, Tehnical University of Cluj-Napoca, Baritiu 26-28 Cluj-Napoca, Romania

E-mail address: {Viorica.Chifu,Ioan.Salomie}@cs.utcluj.ro

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 312–314

RDDNS – RESOURCE-BASED DYNAMIC DNS

ANDREI CRACIUN [(1)] AND ADRIAN STERCA [(2)]

Abstract. This paper tries to solve the problem of resource location in Peer 2 Peer network topologies using Domain Name Servers. A resource is registerred as a DNS record on different name servers by each resource provider. When a peer node requests the resource, it is searched using the dynamic resource-based DNS and the first reply will give the closest resource.

1. Introduction

Usually, when talking in terms of P2P file sharing systems [3], resources are provided by services that run on machines connected to the network. So the elements that define a resource are:

- the address of the machine on which the resource is hosted
- the port on the machine that is used by the service that provides the resource
- an id that identifies the resource

Considering this, an application searching for the resource should receive a response containing all 3 elements.

2. RDDNS Architecture

We can imagine that a P2P network will have a DNS domain that will be used for the search.
Example: Let our network be *somep2p.net*. Let there be 2 providers P1 and P2, 3 resources Res1, Res2 and Res3 and 2 requesters Req1 and Req2. P1 provides Res1 and Res2 and P2 provides Res2 and Res3. There are also 2 DNS servers DNS1 and DNS2 that are both responsible for resolving requests for the *somep2p.net* domain. DNS 1 is closer to P1 and DNS2 is closer to P2.

An example of a possible scenario is the following:

(1) P1 sends 2 register request for *Res1.somep2p.net* and *Res2.somep2p.net* to DNS1.
(2) P2 sends 2 register request for *Res2.somep2p.net* and *Res3.somep2p.net* to DNS2.
(3) Req1 tries to get Res2 and makes a request to both DNS1 and DNS2.

2000 *Mathematics Subject Classification.* 90B18, 68M20.
Key words and phrases. dynamic DNS, peer-to-peer systems.

(4) DNS1 answers first providing to Req1 the address of the P1, the port on which the service is opened and the resource's unique ID.

A possible RDDNS response can be: *Res1@195.132.123.23:12354*.

Regarding the fact that the resource found by using this method is the "closest" one, we rely on the fact that the provider is responsible for registering the resource on a machine that is considered to be the closest. This means that the distance between the requester and the provider is significantly minimized. There are some cases on which this method does not find the top closest resource, but these are rare.

3. Resource IDs

Let there be a resource called *Res.* What is required is an ID that should identify in an unique manner the resource. One important restriction when choosing an ID is that the ID should comply with the DNS requirements (RFC2136). We argue that there cannot be given a specific way to create the resource ID as this should be specific to the resource type. Also the resource ID generation should take into consideration how much information has the requester about the resource. To be more specific we can consider a system on which the resource is a service (i.e. a server application). In this case the name of the service is known by the requester, so the ID can be the name itself (e.g. mytalkservice.somep2p.net). If a file sharing system is considered, the resource ID can be either an MD5 hash on the file (but in this case a parallel search mechanism should be available; that mechanism will provide a way to obtain the hash by making a search using words contained by the filename) or some identification information (such as name and size) can be concatenated to provide an id (e.g.: myfile.txt.10277.somep2p.net).

4. Closest provider

When a provider registers its resource to a DNS server it is responsible for the fact that this DNS server is the closest to it. This can be achieved either by using a network mechanism (e.g. number of hops) or by pre-configuration (in the same manner as the DNS servers are set up on a regular network configuration).

The requester will ask its own DNS server for the address of the resource. The request will be forwarded and the first DNS server that will send a response will be the closest one (in terms of speed). This way by having two shortest paths, the sum path (this may not be the concatenation of the paths, but a sub path of the concatenation) is the shortest one (this is similar to the principle on which Dijkstra's shortest path algorithm is based).

In our example (Fig. 1) we suppose that the speed between all routers in the Internet is the same, and the only think that matters is the number of connections (hops) between 2 nodes. Provider 1 finds as closest DNS server, NS3 (path length is 2). Provider 2 finds as closest DNS server, NS1 (path length is 3). The requester creates the DNS request and the first one which responds is NS1, path length is 5.

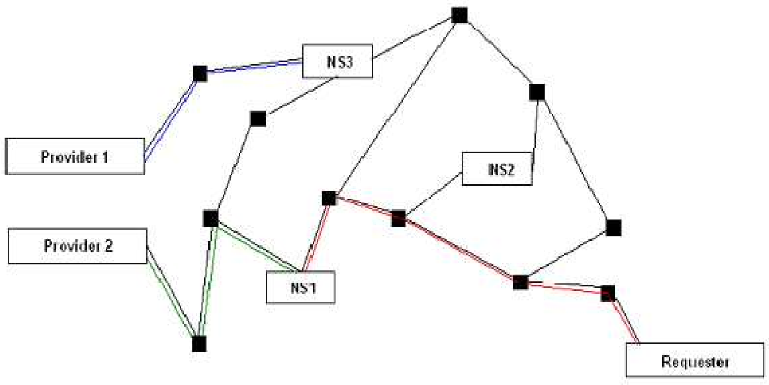


Figure 1. Network topology example

References

[1] P. Mockapetris, *RFC 1034: Domain Names - Concepts and Facilities*, November 1987, available at http://www.ietf.org/rfc/rfc1034.txt.
[2] P. Mockapetris, *RFC 1035: Domain Names - Implementation and Specification*, November 1987, available at http://www.ietf.org/rfc/rfc1035.txt.
[3] http://en.wikipedia.org/wiki/Peer-to-peer.

(1) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania
E-mail address: `rubik@cs.ubbcluj.ro`

(2) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania
E-mail address: `forest@cs.ubbcluj.ro`

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering, Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 315–318

CLUSTERING ALGORITHMS IN OODB FRAGMENTATION – A COMPARATIVE EVALUATION

ADRIAN SERGIU DARABANT[1]

ABSTRACT. The design of Distributed Databases relational (DDBs) or object oriented (DOODs) involve two major steps: fragmentation and data allocation. These two steps have always been difficult to apprehend and conduct by the average database administrator. It comes to no surprise that there aren't many implementations of real true distributed databases as the design process is often too complex. When a database fragmentation is designed it is difficult to asses the quality of the obtained fragments before the actual implementation is in place. In the following we present a comparative evaluation on the quality of fragmentation on DOODs in the context of cluster based horizontal fragmentation.

Horizontal fragmentation is the process of splitting databases entity instances into multiple subsets, each assigned afterwards to different nodes of the system. All fragmentation know at the present date need and use some prior knowledge about the future system. One type of required information is about the structure of the data or relationships between entities (relations or classes). The second type of information regards the applications that will run in the final system once the distributed database design is implemented. The later is in fact a sine-qua-non for the design process. There is no fragmentation method known to the author that can be performed without the knowledge about the applications (queries, updates,etc) that will actually be running in the system. Of course, in any normal real life case it would be difficult to have prior knowledge about *all applications*, so in practice the applications that have the greatest impact on the system are used. Some researchers [2] state that the most active 20% of user queries will stand for 80% of total data accesses.

1. CONTRIBUTIONS

This paper presents a comparative evaluation between two fragmentation algorithms we propose: *the hierarchical fragmentation and k-means clustering fragmentation.* Both algorithms are based on clustering techniques [1].

We also compare our methods with some of other fragmentation methods presented in [2, 4, 5, 6, 7]. Most of these are derived from fragmentation methods based on relational models which are simpler than the Object Oriented Data Model (OODM) and do not capture all its features and characteristics.

2000 *Mathematics Subject Classification.* 68P15, 68M20, 68M14.
Key words and phrases. distributed databases, design, evaluation, object oriented.

As (OODM) can be generally considered a superset of the relational data model, we can assume that the same implications and results will hold when fragmenting a relational data set.

We focus in this paper on comparing the quality of horizontal object oriented fragmentation obtained by applying our customized clustering algorithms. They are presented in detail in [6]. Basic hierarchical and k-means centroid-based algorithms [1] are well known techniques in the clustering theory. Our work emphasizes on enhancing these algorithms in order to apply them in distributed database fragmentation. We focus on showing that database fragmentation by clustering could be a lot more simpler to apply than traditional methods, while maintaining or enhancing the fragmentation quality. The comparative study is performed for object models with simple attributes and methods by computing a cost function over the obtained results.

2. Fragmentation with k-means and Hierarchical clustering

Our fragmentation method takes as input a set of simple predicates from the most representative queries. We attach to each object numerical values representing the way that object respects the simple predicates. We obtain thus, for each object, a numerical vector that stores quantitative and qualitative information about object selection by all predicates. We represent all objects as N dimensional vectors(points) in the N dimensional space of query predicates. The distance(or similarity) between two points in this space reflects how strong are the two corresponding objects related in terms of behavior to user queries. Sets of strong related objects are good candidate to form distinct fragments.

For the instance set of a given class, represented in our N dimensional space as vectors, we define the notions of distance(similarity). Similarity computation is based on well known metrics as Manhattan and Euclid or uses implicit similarity functions as the vectorial cosine.

Finally, we only need some algorithmic way for determining sets of similar/close objects in the space of user queries. We use our enhanced versions of the k-means and hierarchical clustering algorithms to find the sets of strongly similar objects that form the resulting fragments.

3. Results

By using an example database and a set of user queries available to public we applied our fragmentation methods and measured the fragmentation quality. In order to evaluate the fragmentation quality we use the partition evaluator proposed in other similar works as [4, 2].

$$PE(C) = EM^2 + ER^2 \tag{1}$$

The evaluator (PE) computes the cost of accessing local data (EM) and remote data (ER) when running the set of user queries over the fragments of a class. As the value of the cost increases, the quality of fragmentation is weaker.

$$EM^2(C) = \sum_{i=1}^{M}\sum_{t=1}^{T} freq_{ts}^2 * |Acc_{it}| * \left(1 - \frac{|Acc_{it}|}{|F_i|}\right) \tag{2}$$

$$ER^2(C) = \sum_{t=1}^{T} \min\left\{\sum_{s=1}^{S}\sum_{i=1}^{M} freq_{ts}^2 * |Acc_{it}| * \frac{|Acc_{it}|}{|F_i|}\right\} \tag{3}$$

The quality of fragmentation expressed as the cost of evaluating queries against the resulting database is expressed in the Figure 1:

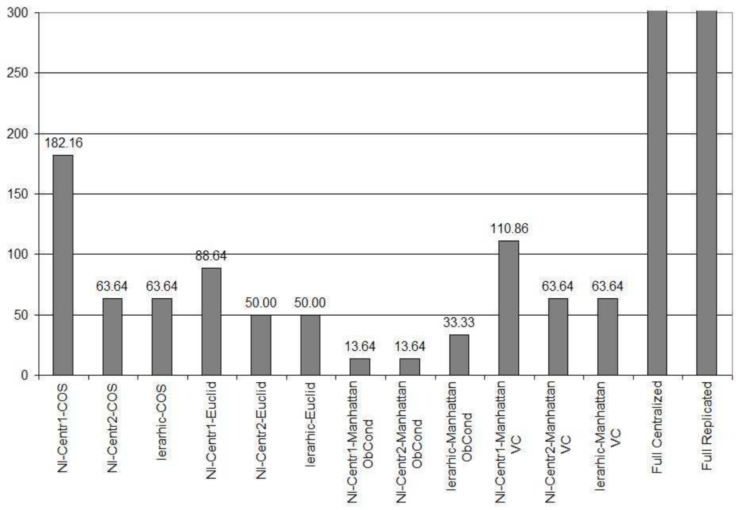


Figure 1. The cost when evaluating queries against the fragmented database.

4. Conclusions

In this paper we present a comparative evaluation of the fragmentation quality obtained by applying our *k-means and hierarchical clustering* algorithms. Experimental results show that applying the algorithms is straightforward for a given set of queries and the results are usually better or within the same limits as the ones obtained with traditional fragmentation methods. We compared the fragmentation costs(quality) for multiple variations of the algorithms and similarity measures in order to extract the best combination. As observed experimentally the Manhattan similarity with the k-means algorithm generally obtains the best scores as it reduces error propagation during clustering by re-assigning objects to their proper clusters.

References

[1] Han, J., Kamber, M., Data Mining: Concepts and Techniques, The Morgan Kaufmann Series in Data Management Systems, 2000.

[2] Karlapalem, K., Navathe, S.B., Morsi, M.M.A. - Issues in distribution design of object-oriented databases. In M. Tamer Ozsu, U. Dayal, P. Valduriez, editors, Distributed Object Management, pp 148-164, Morgan Kaufmann Publishers, 1994.

[3] Karlapalem, K., Li, Q., Vieweg, S. - Method Induced Partitioning Schemes in Object-Oriented Databases, In Proceedings of the 16th Int. Conf. on Distributed Computing System (ICDCS'96), pp 377-384, Hong Kong, 1996.

[4] Ezeife, C.I., Barker, K. - A Comprehensive Approach to horizontal Class Fragmentation in a Distributed Object Based System, International Journal of Distributed and Parallel Databases, 33, pp 247-272, 1995.

[5] Karlapalem, K., Li, Q. Partitioning Schemes for Object-Oriented Databases, In Proceedings of the Fifth International Workshop on Research Issues in Data Engineering-Distributed Object Management, pp 42-49, Taiwan, 1995.

[6] Darabant, A. S, Campan, A. - Semi-supervised Learning Techniques: k-means Clustering in OODB Fragmentation, In Proc of the IEEE Intl Conf on Computational Cybernetics ICCC 2004, pag: 333 338, Wien, Austria

[7] Ravat, S. - La fragmentation d'un schema conceptuel oriente objet, In Ingenierie des systemes d'information (ISI), 4(2), pp 161-193, 1996.

[8] Bertino, E., Martino, L. - Object-Oriented Database Systems; Concepts and Architectures, Addison-Wesley, 1993.

[9] Bellatreche,L., Karlapalem, K., Simonet, A. - Horizontal Class Partitioning in Object-Oriented Databases, In Lecture Notes in Computer Science, volume 1308, pp 58-67, Toulouse, France, 1997.

[10] Baiao, F., Mattoso, M. - A Mixed Fragmentation Algorithm for Distributed Object Oriented Databases, In Proc. Of the 9th Int. Conf. on Computing Information, Canada, pp 141-148, 1998.

(1) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania

E-mail address: dadi@cs.ubbcluj.ro

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 319–322

ON EVALUATING THE PERFORMANCE PARAMETERS IN A DISTRIBUTED SYSTEM

COSTEL ALDEA [1] AND FLORIAN MIRCEA BOIAN [2]

Abstract. Parallel and distributed applications have a variable workload due to the fact that these types of applications contain software and hardware components that vary a lot as well as due to their evolution in general, and due to the evolution of computers and networks in particular. With respect to this evolution many different benchmarks to evaluate the performance of distributed and parallel algorithms were developed. The benchmarks are often based on different performance factors starting with execution time, failures, hardware counters, bandwidth measurement between memory and CPU, network bandwidth etc. In this paper we introduce a performance factor based on the approximation error. We also propose an algorithm for the estimation of this performance factor.

1. Introduction

Performance is defined based on the response time and the computing capacity. Furthermore, the performance of a distributed system is defined and evaluated using different measurable performance factors like: instructions per second, CPU speed, the number of floating point operations per second or the total amount of time need for a benchmark execution [7].

Using the approximation spaces we introduce a performance factor which can be measured like other performance factors (execution time, bandwidth etc). For example, when using matrix operations, in the parallel and distributed computing there are often cases when approximations are made (for example a high dimensional matrix is approximated through a low dimensional matrix). The approximation errors are actually number sequences that are contained into an approximation space on which corresponding norms are defined. In the matrix multiplication case it is desirable that the approximation of the product obtained by aproximating the initial matrices be exact. Each matrix approximation and matrix operation approximation (sum, product, etc.) leads to an approximation number sequence. We define a performance factor f by dividing the approximation number sequence given by the original matrices approximation to the approximation numbers sequence given by matrix operation result. The factor f characterizes the performance in two ways: approximations made and the relation between them is related to the quality of approximation so that the

2000 *Mathematics Subject Classification.* 46N10, 68M20.
Key words and phrases. Distributed system, performance, approximation error.

operation result (sum, product) is exact and approximations can be made more than once until the result is appropriate so that the response time is variable.

It can be noticed that this form of the performance factor based on the approximation number is more general and it is valid for any approximation not only for the operation with the approximation of matrices through smaller matrices. In [6] Aldea treats the sequence spaces as special approximation spaces based on the idea that the approximative ideals of operators are particular sequence spaces [4] and proves the inequality on which the definition of the factor f is based on the approximation numbers on a normed space X.

For evaluating of implied performance factors in a parallel and distributed system one uses special methods called *benchmarks*[7]. The main characteristics of a benchmark system are: it determines unique standardized factors for characterizing hardware and software systems; it proposes optimal values for factors on which the system design relies, etc.

2. Preliminary results

Let X be a normed space and $S = \{y \in X : \| y \|^* < \infty\}$. We define, $\forall x \in X$, the approximation number sequence $(E_n(x))$ as follow:

$$E_n(x) = \inf\{\| x - y \|_X \colon y \in S, \| y \|^* < n\}, n = 1, 2, ...$$

where $\| \cdot \|^*$ is a norm on X.

We define the sequence of approximation numbers $(a_n(x))$, for $\forall x \in \ell_\infty$ as follow [6]:

$$a_n(x) = \inf\{\| x - \overline{x} \|_\infty : \ card(\overline{x}) < n\}, n = 1, 2, ..., \tag{1}$$

where $\overline{x} = (x_1, ..., x_{n-1}, 0, 0, ...)$ is a subsequence for which $card(\overline{x}) < n$.

The approximation number sequence (1) of the sequence $x \in X$ is the sequence $(E_n(x))$ for the particular case when $X = \ell_\infty$ [6].

Proposition 1

The numbers $a_n(x)$ observe the inequality [6]:

$$\sum_{n=1}^{k} a_n(x_1 + x_2) \leq 2 \sum_{n=1}^{k} (a_n(x_1) + a_n(x_2)), k = 1, 2, ..., x_1, x_2 \in \ell_\infty. \tag{2}$$

In this paper we consider $X = \mathcal{M}_2$, so that the inequality implies that the sum or product (for e.g.) of the matrices approximation errors is devided by the approximation error of matrix operation results.

3. Main results

By using the inequality (2) we introduce the performance factor f computed as fraction between the inequality members and we propose an algorithm for estima-ting the factor in the case of matrix addition and matrix product. To simplify the calculus, we use the two dimension matrices ($M \in \mathcal{M}_2$).

If in the inequality (2) we substitute 2 by f in the case of the product of two matrices $A, B \in \mathcal{M}_2$ we have:

$$E_T(A \cdot B) \quad \leq f_{prod} \quad \cdot E_T(A) \cdot E_T(B) \tag{3}$$

While $A, B \in \mathcal{M}_2$ we have:

$$E_1(A) \quad = \quad \max |a_{ij}|, \forall i, j = \overline{1, n} \tag{4}$$

$$E_2(A) \quad \leq \quad \| A - A_1 \|, \text{ where } A_1 \in \mathcal{M}_2 \text{ and } rang(A_1) < rang(A) \tag{5}$$

The approximation errors for a diagonal matrix are equal to the values on the main diagonal [5].

We present the parallel thread based algorithm which estimates the factor f.

The steps of the parallel algorithm ***PAf2M2*** (Parallel algorithm for factor f estimation in the case of the sum and the product of two matrices from $\mathcal{M}_2$):

S1. for each $a_{ij}, b_{ij} \in \{-\alpha, \alpha\}, \alpha \in \mathbb{R}_+$ determines the matrices A, B for which:
   (a) $rang(A) = 2$ and $rang(B) = 2$,
   (b) $A + B$ is diagonal matrix and $A \cdot B$ is diagonal matrix;

S2. it computes the approximation numbers $E_1(A)$ and $E_1(B)$ (using (4));

S3. for each matrix A, B, computes using (5), the inferior rank matrices A_1, B_1 to evaluate the approximation numbers $E_2(A)$ and $E_2(B)$. It generates the A_1 and B_1 elements using the incrementation step $\epsilon, \epsilon \in \mathbb{R}_+, \epsilon \to 0$ in the interval $(-|\max(a_{ij})| + \epsilon, |\max(a_{ij})| - \epsilon)$. We choose from the A_1 generated matrices the one with the minimum value of norm $\| A - A_1 \|$;

S4. it computes $E_T(A + B)$ and $E_T(A \cdot B)$ as the sum of the diagonal elements of the sum matrix, and respectively, of the product matrix ($E_T(A) = E_1(A) + E_2(A)$);

S5. it computes the total approximation error for the sum, respectively, for the product of the matrices by using $E_T(A) + E_T(B)$ and $E_T(A) \cdot E_T(B)$;

S6. it estimates the factor f for sum (f_{sum}) and also product (f_{prod}) from inequality (3) :

$$f_{sum} \geq \frac{E_T(A + B)}{E_T(A) + E_T(B)}, \quad f_{prod} \geq \frac{E_T(A \cdot B)}{E_T(A) \cdot E_T(B)} \tag{6}$$

The algorithm ***PAf2M2*** was implemented in Java using peer thread model [2]. In this model, each thread is responsible for its input data. The peer knows the type of the input at the beginning and has its own means for working with input data. In ***PAf2M2*** algorithm peers compute the following: after receiving a matrix it computes the E_1 number using (4) - step S2 or after having E_1 numbers it generates inferior matrices and it computes E_2 numbers using (5) - step S3.

4. Conclusion

Analyzing the data obtained through the program ***PAf2M2*** execution using $\alpha = 3$ and $\epsilon = 0.1$ it can be observed that $f_{sum} \in [1, 1.40]$ and $f_{prod} \in [0.35, 1.91]$. It can also be observed that, based on the approximations made for the matrices the studied performance factor f for matrix product f_{prod} and sum f_{sum} is variable.

The running parameters of the program ($\alpha = 3$ and $\epsilon = 0.1$) give $f_{sum} \leq 2$ and also $f_{prod} \leq 2$ which is in concordance with the theory [4].

Relativ to performance, based on the data obtained through the program ***PAf2M2*** execution two cases for the factor f_{prod} are distinguished:

(a) when $f_{prod} \leq 1$, for example $f_{prod} = 0.35$, it results, by using (3), that $E_T(A \cdot B) \leq 0.35 \cdot (E_T(A) \cdot E_T(B))$; thus, the product of approximation errors for the initial matrices is bigger than the approximation error for the result (which is preferable);

(b) when $f_{prod} > 1$, for example $f_{prod} = 1.91$, then the product of approximation errors for the initial matrices is smaller than the approximation error for the result (from good approximation with smaller errors of the initial matrices are obtained larger error values for the result). In the second case new approximations are needed and the execution time and the computational performance are negatively affected.

In this paper we have discussed the estimation of the performance factor in the case $X = \mathcal{M}_2$, but this performance factor can be estimated in the case of any approximation space X using the corresponding norms.

5. References

[1] K. P. Birman, *Building secure and reliable network applications*, Worldwide Computing and Its Applications, Springer Berlin, vol. 1274, pg. 15-28, 1997.

[2] Boian F.M. şi alţii, Programare concurentă pe platforme Unix, Windows, Java, Editura Albastră, Cluj, 2002.

[3] Coulouris G.F., Dollimore J.B., Kindberg T., *Verteilte Systeme. Konzepte und Design 3., ueberarbeitete Auflage*, Pearson Education Limited, 2002.

[4] N. Tiţa, Aldea C., *Capitole speciale de teoria operatorilor (Interpolation and tensor product stability)*, Ed. Univ. Transilvania Braşov, 2005.

[5] N. Tiţa, *Approximation spaces and bilinear operators*, Studia Univ. Babeş-Bolyai, Ser. Math. 35 (1990), 4, 89-92.

[6] C. Aldea, *On treat the sequence spaces as special cases of approximation*, Proceedings of the 12th WSEAS International Conference on Computers, ISBN 978-960-6766-85-5, pg. 383-386, 2008.

[7] http://www.spec.org/benchmarks.html#hpg.

(1) "Transilvania" University of Braşov, Iuliu Maniu Str. 50, RO 500091

E-mail address: `costel.aldea@unitbv.ro`

(2) Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania

E-mail address: `florin@cs.ubbcluj.ro`

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 323–326

ENHANCING YAHOO! SEARCH RESULTS USING LINKED OPEN DATA

CIPRIAN AMARIEI, EMANUEL ONICA, AND SABIN BURAGA

Abstract. The common web searches in the present day can no longer be fulfilled through simple page content based crawling. Usually, an informational query made through a search engine is far less often targeted only to finding a specific result containing the asked terms and is much more focused on getting the most relevant response in a specific context and eventually broaden up this context by offering related information without losing the initial search idea. To satisfy such user needs, methods based on the semantic mechanisms of the web emerged, many of these using Linked Open Data (LOD) in their functionality. We present certain ideas of enhancing the search results obtained through one of the current semantic search solutions – the Yahoo Search Monkey, mainly based on custom LOD exploitation.

1. Introduction

One of the most important issues in web search is the relevancy of the results. A common definition of relevancy [1] may be expressed as the ability of the search platform to obtain information that closely satisfies the needs of the user. Unfortunately, even taking into consideration the case when the search result will closely match the concept the user had in mind when entering the query, this will not guarantee that starting from that result, further searches are not initiated in order to find other related information, but from a different context. We consider the following scenario: a search based on key phrase "fried tomatoes". Besides finding possible culinary aspects related results, the user may be informed about a movie or a book that matches the same query.

One possible way of interconnecting the mentioned concepts in the context of web search is using Linked Open Data (LOD). A concise definition of LOD would be a "style of publishing data on the web that emphasizes data reuse and connections between related data sources" [2]. According to [3] recent trends in Semantic Web focus "almost exclusively" on LOD rather then Annotated Web. Thus, by making use of LOD annotations, additional queries related with the main search subject, like the exemple presented, are possible. A similar idea is described also in the context of an *entity-centric* search engine in [4].

2000 *Mathematics Subject Classification.* 68N01, 68U35.
Key words and phrases. search, linked open data, Web.

We propose such an entity centric solution idea based on the current support for semantic search within the Yahoo Search platform. The next section shortly describes the Yahoo Search Monkey framework and the idea of an entity centric enhancement using LOD. In the section 4, we detail a proof of concept implementation of a use case starting from the movie search example we presented above.

As related work we mention an Entity Centric Semantic Search Engine using Sindice infrastructure combined with the capabilities of Okkam [4]. Another application of an entity centric approach is Zemanta.

2. Background of Web Search

The queries could be classified into three categories [5]: *navigational*, *informational*, laptop), and *transactional*.Our previous example might be included in the *informational* area according to the original intent of the user. However, the initial query could "evolve" into one of the other two classes.

The search categories specified are further refined in [6]. Our "fried tomatoes" example belongs to the *undirected* class (queries targeted to learn any information about the topic). The user's intent behind the query is not restrained to specific information. In [6] the results of an analysis showed that in two of three sets of queries the most were *undirected* – around 30%. Therefore, we consider important the study of *undirected* queries as one of the predominant classes of web searches.

Another query goal subcategory of *informational* class [6], is *list* – to obtain a list of candidate web sites to further help the user regarding the topic. According to the results obtained in [6], which are regarded as distinct by classifying a query in only one of the categories during observation tests, we believe that the *list* class intersects, with the *undirected* one. Actually such links could be established between other goals too, having the *list* goal as a "hub" for linking them. Based on user's behavior we may imply a possibility of more than one goal, or a goal evolution of the query, that could be also translated into additional queries.

3. LOD Query Enhancement using the Yahoo Search Monkey

Yahoo, one of the major web search engines, has introduced support for semantic search through the associated Search Monkey Platform. However, this still is in an early state having limitations [7] like no SPARQL [8] endpoint crawling and no LOD crawling support. To build an entity centric search, our approach overcomes these drawbacks by a certain degree as we describe bellow.

Yahoo Search Monkey (YSM) is essentially a framework dedicated to developers for creating add-ons which can be associated with the results obtained from web queries entered via this search engine.

Two add-on types are defined: *Enhanced Result*, that reconfigures the search result itself and *Infobar* – an expandable HTML pane which is attached to a typical display of the Yahoo query result exhibiting additional information. In our proof of concept, described below, we have used the latter for its flexibility.

The developer may use as a data source an approach based on XSLT transformation applied on the query results page. To achieve a more versatile way of acquiring

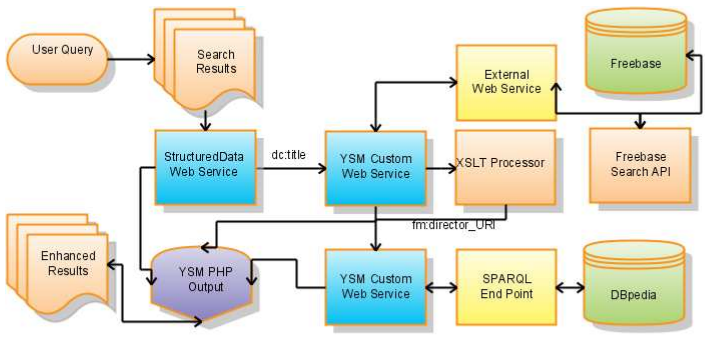


Figure 1. Linked Movie Monkey Infobar architecture

needed data we opted for *custom data services* based on external web services which is the other data source type defined by YSM.

This involves the internal supported data sources like Yahoo Index (Structured-Data Service in Figure 1) and other custom data services which in order to obtain input parameters needed for our purpose are linked based on a custom service ID generated by the framework. Though this is not very well documented, we concluded to be unique.

The custom service chaining described above is completely distinct from actual using the external web services which means an even further extension of service linking. Therefore the platform offered us the possibility of putting in connection a virtually unlimited series of either YSM data services or external web services.

Such an approach gives the developer a large range of possibilities for enhancing the final results obtained through such a link of services, limited however by the response time aspects. These possibilities can include LOD browsing, stepping over the limitations caused by the lack of actual support, ontology matching and alignment, or common operations like text processing.

Also in direct relation with using external web services we also would like to point out another limit stated in the official YSM documentation, to crawl a SPARQL [8] endpoint, that our solution overcomes, at least at a certain degree. So an SPARQL endpoint was used as an external web service by passing parameters to a fixed query template. Although quite inflexible, it is a possible way to enrich a query result with data gathered through SPARQL.

In the next section, we detail the proof of concept of a simple use case, in which any term search results are enriched with additional possible related movie information. In the example, we are using several of the mentioned techniques in order to achieve this bringing a normal search closer to an entity centric search.

4. Use Case: Linked Movie Monkey

We describe an YSM application as basic proof of concept for the notions presented above. The add-on has the purpose to enhance every query result received by the user with an additional response containing a movie reference on related topic. As mentioned before, a query implying “fried tomatoes” with results describing culinary aspects, might have attached an YSM Infobar displaying “Fried Green Tomatoes” as a related movie. Solution architecture is depicted by Figure 1.

5. Conclusion

In the present, there are not many YSM applications using the service chaining mechanisms described in this paper, the majority being focused on using the platform in presentational purposes. From our point of view, this situation has roots in the lesser complexity of developing an YSM based on the page content related data services, and not in the lack of interest on LOD opportunities.

Related to our enhancement approach, and specifically the described movie use case, we could say that many of the depicted steps may be simplified or substantially improved. Our purpose was not actually getting the closest related movie, but to offer a straightforward approach on YSM data services and external web services or wrappers pipelining, proving how multiple semantic Web technologies can be linked to obtain the desired result, with real benefits for the end-user.

We proved the feasibility of the platform for supporting an entity centric search approach query enhancement using LOD to make possible the semantic mash-ups.

References

[1] D. Schueren and F. Badcock, Creating Information Gravitation in the Firm, in *ILTA*, 2008.
[2] T. Heath, Looking ahead to Linked Data on the Web Vol. 1, Talis, 2008.
[3] P. Mika, Microsearch: An Interface for Semantic Search, in *Workshop on Semantic Search, 5th European Semantic Web Conference* Vol. 334, CEUR, 2008.
[4] G. Tumarello and R. Cyganiak, Early Demonstrator "Entity Centric Search", in *Okkam - Enabling a Web Of Entities, Grant Agreement No. 215032*, 2008.
[5] A. Broder, A taxonomy of web search, in *SIGIR Forum* Vol. 36, pp. 3–10, 2002.
[6] D. Rose and D. Levinson, Understanding User Goals in Web Search, in *13th International Conference on World Wide Web*, pp. 13–19, ACM, 2004.
[7] *Yahoo Search Monkey*, http://developer.yahoo.com/searchmonkey/, 2008.
[8] J. e. a. Perez, Semantics and Complexity of SPARQL Vol. 4273, Springer, 2006.

(1) “A.I.Cuza” University of Iaşi, 16, Berthelot Street – 700463 Iaşi, Romania
E-mail address: {camariei,eonica,busaco}@info.uaic.ro

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering,
Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 327–330

SERVER-SIDE MOBILE APPLICATIONS

ANDREI CRACIUN [(1)]

Abstract. The main purpose of this paper is to prove that nowadays mobile devices can serve successfully as hardware and software support for several types of server-side applications. Also we'll give some implementation solutions for most topologies, including the worst case scenarios. Another very important aspect of the current paper is that all the solution given should work in a real life environment.

1. State of the art

Mobile devices have been created having in mind the fact that their role in distributed applications will be the role of a client and most of the time as a "non-smart" client (because of the poor computational power not many features can be found on mobile based application, comparing to let's say the desktop or the web version of the same application).

During the last few years the technical capabilities of the mobile devices (mobile phones, PDAs) have been increased in order to face the new challenges required by the market (specially media requirements). Also the requirements for connected mobile devices forced the manufacturers to add more and more features as nowadays it is a common thing for a mobile device to have both WAN (GPRS, EDGE, 3G) [1] and LAN (Wi-Fi) [2] capabilities.

2. Connectivity, networks, nodes

The most important part of a server side application is it's visibility. The highest visibility an application can get is if any device with an Internet connection can access it. Unfortunately the policies of the mobile carriers do not allow mobile devices to have public IP addresses because all traffic has to be paid by the user. On the other hand, when talking about mobile server-side applications is the fact that the type of the connection (GPRS[1], WiFi[2]) and the topology of the network is very dynamic.

Connections are made between network nodes. Network nodes are usually represented by machines. We always say: I ping-ed that machine. Ive made an ssh connection [3] to that machine. Is this true? Yes and no. Yes because actually that really happens and no because a machine is a little too generic. We actually connect

2000 *Mathematics Subject Classification.* 90B18, 68M20.
Key words and phrases. mobile applications, server-side.

to an application, a server application. So a network node is an application (server or client application).

While studying mobile network topologies we've reached to the conclusion that there are two types of networks, depending on the life cycle of the connection between nodes. We'll define these two types in the following statements:
Let N be the set of nodes in a network.
If a, b are nodes in a network there is a direct connection between a and b if a and b share the same network (IP network).
Static network: If randomly chosen n and m out of N and if there is a direct connection between n and m that connection will not be changed.
Dynamic network: If randomly chosen n and m out of N and if there is a direct connection between n and m that connection can be changed and no assumption can be made on it.

Considering the elements defined above the following problem raises: Supposing that n and m are two nodes in a dynamic network, the problem is to create an IP connection between n and m.

3. Dynamic networks topologies

As said before in a dynamic network no assumption can be made upon the connection between nodes. This means that connections can be broken, new connections can be made or some of the nodes can change the state into a NAT[4] type of connection.

The worst case is: Having in mind a permanent connection between 2 nodes located in a dynamic network we can consider an worst case scenario (in terms of network topology). The worst case scenario reproduces when the two nodes we want to connect are both in a NAT [4] topology, and the gateway that does NAT[4] for each of them changes.

An example: We have two persons (Romeo and Juliet) who both have phones (lets call the phones R and J). R is using a different mobile provider than J and both phones have a WLAN connection available. The scenario is the following:
1.Romeo is at home and R is connected to the internet via the local WLAN[2] connection.
2.Same thing for Juliet.
3.Romeo wants to go to visit Juliet and as he gets far from home; the WLAN[2] connection is no longer available and R is connected now through 3G.
4.While R leaves the city the 3G is no longer available and R is connected through GPRS[1].
5.In the same time Juliet comes to visit Romeo and she leaves home now using an EDGE[1] connection.
6.When J arrives at Rs and R at Js they both use the others WLAN[2] connection.
7.Being upset for not finding Juliet at home Romeo went to a local in order to have

a coffee. Now R is connected through the open WLAN[2] at the coffee shop.

So, in any of those moments the 2 phones R and J should be able to send data from one to another.

4. A solution

The system that we are trying to implement has two main components:

- the rendez-vous server (R server) having as a main role the keeping of a live server connection in order to be accessible from any network
- the name server (N server) which is responsible for keeping the name of the server application

The two machines must have public internet addresses (IP) in order to be accessible from any machine/device in the internet.

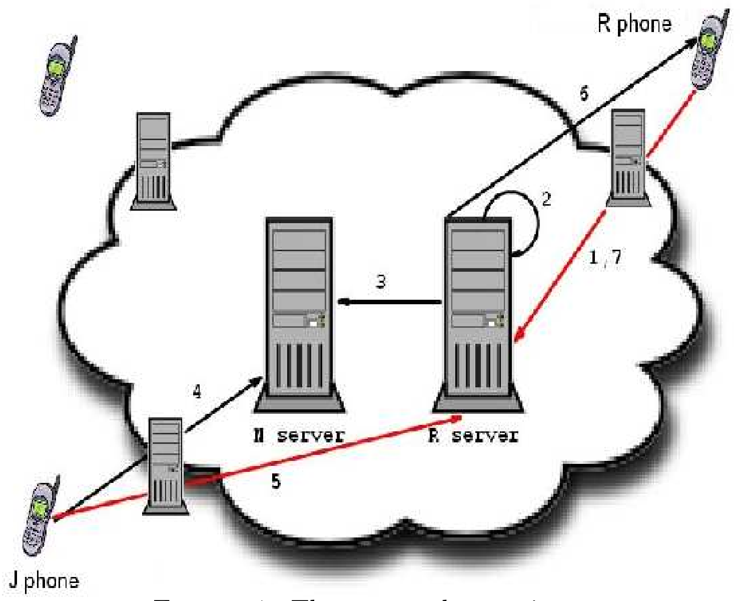


Figure 1. The proposed scenario

The scenario (see Fig. 1)
1.a connection is opened from the server phone to the R server in order to notify it that a new application is available. During this step the server phone provides all the data needed for back notification.
2.the R server opens a server connection on a required or random port and closes the connection with the server phone.
3.the new application name along with the port is registered to the N server by R.
4.the client phone requests from the N server the host and the port where the application can be found.

5.a new client connection is made by the client phone to the R server on the port indicated by the N server.
6.the R server notifies the server phone on the fact that there is a new inbound connection (this is the back notification).
7.the server phone creates a new connection to the R server and the R server will associate the 2 phone connections.

With this scenario it is possible to create connections between two node located in a dynamic network in a client transparent way.

We've identified some areas where the system developed by us can be used. A first important domain is home automation. The main usage is that a mobile device (phone or PDA) is located in the house and it plays the role of a bridge between the home appliances and the remote device which is used as a remote controller. We've realized two projects, one using Internet (IP) and one using GSM network. The main goal of the projects was to prove that in real life mobile devices can be used to host server side applications. One other domain identified is remote controlled machines. We are currently developing a project which has as a main goal making of a boat that can execute a mission based on GPS coordinates. The boat is controlled by a mobile device located on it.

5. Conclusion

As proved in the statements above it is clear that there are several classes of server side applications that can be hosted by mobile devices. Also there are a lot of advantages of using a lite platform as a mobile one, such as power consumption and space occupied by the device. Considering these aspects we think that this subject should be investigated mainly in order to find new approaches of the technology and to solve the problems created by the client oriented design of the mobile devices.

References

[1] Guide to GSM, GPRS, 3G, EDGE, HSDPA – http://www.clove.co.uk/viewTechInfo.aspx?Content=3B2BD491-6465-4C70-ABDB-5A12A06C3D8D&Category=TECHINFO
[2] Wireless LAN - http://en.wikipedia.org/wiki/Wireless_LAN
[3] The SSH Protocol - http://www.snailbook.com/protocols.html
[4] Network address translation - http://en.wikipedia.org/wiki/Network_address_translation
[5] Bluetooth serial connection - http://developers.sun.com/mobility/apis/articles/bluetooth_gps/part1
[6] X10 (industry standard) - http://en.wikipedia.org/wiki/X10_(industry_standard)
[7] X10 CM11 interface - http://www.uk-automation.co.uk/computer-interface-p-996.html
[8] Windows Mobile 5 - http://www.pocketpccentral.net/wm5brief.htm

[1] Faculty of Mathematics and Computer Science, Babes-Bolyai University, Cluj-Napoca, Romania

E-mail address: rubik@cs.ubbcluj.ro

KNOWLEDGE ENGINEERING: PRINCIPLES AND TECHNIQUES
Proceedings of the International Conference on Knowledge Engineering, Principles and Techniques, KEPT2009
Cluj-Napoca (Romania), July 2–4, 2009, pp. 331–334

VIRTUALIZATION, THE SOLUTION FOR DYNAMIC IT

MIRELA CARMEN BIZON (FLOREA)[(1)]

ABSTRACT. In a world in which things are changing rapidly at global level, when organizations are spread functionally all over the planet, what gives priority to competitors is mainly the imperative spirit of innovation. Information technology and communications are omnipresent components and integrating parts of all the organizations' work environment. Employees communicate with partners electronically, without ever meeting face to face. The disappearance of limits regarding time, location, culture, business relation, software, leads to new and diverse ways of process organization.

Within the economical environment that is influenced by the financial crisis, when the main objective of organizations is to decrease expenses, the IT environment feels the impact of recession and it must adapt by assuming a new approach that can improve flexibility, efficiency and performance.

The IT projects are required to diminish costs, to increase effectiveness, and all these without security breech or system instability generation. From the point of view of the IT activity, of optimizing at each and every level (place, equipment, administration, human resources, energy efficiency, etc.), the solution for all these problems is represented by virtualization technology.

The word "virtualization" originates in the well-known procedure of "partitioning", which means dividing one physical server into more servers. Virtualization was defined as a programming layer that separates physical hardware from application (Virtualization) and it was first introduced in the 1960's.

First of all, in the 1990's, the "virtualization" concept was used to re-create different work environments on only one hardware partition. The Gartner Group made public at the Gartner Symposium / Itxpo yearly event, held in Orlando, SUA, its traditional study about new key technologies of the next year - "Top 10 Strategic Technology Areas for 2009". The conclusion of the Gartner specialists was that the year 2009 will be the year of virtualization, of the new server generations ("Beyond Blades"), of the Web Oriented architectures, of the specialized systems, of mash-ups, of Social Software and Social Networking, of Business Intelligence technology and of "Green IT". [2]

2000 *Mathematics Subject Classification.* Knowledge in Distributed Computing.
Key words and phrases. Virtualization, Flexibility, Virtual Machine (VM), Green IT, Distributed Server Architecture.

Implementing such a server virtualization application allows only one physical platform to run more virtual machines simultaneously. Each of these machines has its own processor, its own memory, network interface, and the operation system that functions on each virtual machine is called "guest". From the functional point of view, each virtual machine is autonomous and it does not "know" that the resources are parted so that only one platform could have more servers (with the same operating systems or even more different systems). Regarding the virtualization technologies, one can usually choose either proprietary technologies or OpenSource.

1. The benefits of virtualization

Virtualization increases calculating capacity. Normally, the servers are dedicated to only one application and to only one operating system, the degree of use of the servers in a datacenter is about 17% according to a study made by the company. Virtualization allows more systems and applications to function on only one physical server. With more virtual machines consolidated within a variety of hardware resources, the utilization rate is monitored continuously and it increases up to 80%. Virtualization helps us use the servers more effectively, the servers in which we have invested time, money and knowledge. Decreasing the number of servers lowers the cooling and supplying costs and it generates a considerable reduction of the occupied space.

But not only physical factors are important. From a software point of view, the companies can use the resources more flexibly or they can increase the availability of services. The gained competitiveness cannot be calculated in financial terms, but the advantage is visible. When a visible machine is too loaded because of virtualization, a part of the virtual machines can be rapidly and easily moved to another physical machine (server). This phenomenon is called migration. Concretely, the image of a virtual machine is copied (a file) which is to be moved to another hardware system.

Another functionality of virtualization is represented by the testing capacity. Most of the software companies but also the IT personnel of companies use virtualization to test diverse applications or procedures that run on diverse operating systems. Cloning in such situations and generating a new virtual machine can be accomplished in only a few minutes, as compared to installing a physical machine which can last 2-3 hours. The configurations can be changed much easier and they don't require a physical machine for each modified configuration. The different testing environments can run simultaneously on the same virtual machine, too, which implies a better analysis of the test products.

Another benefit of the virtualization implementation is represented by the continuity of the business. In case of system errors that cannot be remediated, the data can be rapidly recovered. The immediate relocation function will allow the execution of the load balance operation of the virtual machines even while these are running.

2. Data storage and data protection

The challenges of the virtual work environment are the storage method and administrated data protection. Replication can be made either within a "host" operating system, more specifically within the same physical machine, by using dedicated virtual machines, the method of hard disks, or within a "guest" system - in the traditional way - by the help of a recovery physical machine. Replication from "guest" supposes that the disk can be written at host platform system level while all the other applications will run at folder level.

The most secure and the most often used method is installing a target server and a backup solution within the operating system of the host platform to realize the replication of data directly on the target server. The target server monitors the activity of the host operating system and in case that a problem might occur, a script will run and will install an identical virtual server on the target server using the replication of the data on the virtual disk.

Protecting multiple applications, replication of more virtual servers on a target operating system implies permanent analysis of data and used replication technologies because software and hardware compatibility problems may occur. The target server must be able to install at any moment a virtual machine for each source server.

3. Case study. Projecting a virtual system within a public institution

According to the Romanian Constitution, art.135, point 5, the public institutions came into being by power acts or by dispositions of the Central or Local Public Authorities in order to accomplish commercial activities or to do non- patrimonial public services. There are structures organized in each county and in Bucharest.

For the local and central administration, the computerized evidence is needed. At the starting point of IT development, data storage process was not a problem, thanks to the data type used (text and numeric data). The acquisition and the usage of a server and using the corresponding storage system was enough. This is the Single Server Architecture (SSA). At present the DSA - Distributed Server Architecture - infrastructure is utilized, which is based on the usage concept for application servers, data base servers, e-mail, schedule, internal application ERM/ERP/CRM.

This is not the best solution because once occupied the maximum storage capacity, adding a new server is necessary to supplement the stocking space. A brief analysis of this situation discovers a lot of problems: the lack of a centralized storage system, the users and the running processes are dependant on the best functioning of the computer communication infrastructure, lack of a modern system of continuing the activity and service restoration automatically in case of damage, many different software and hardware configurations that are used, which leads to the impossibility to apply centralized management, exploitation costs that keep on increasing (electric power, cooling Systems, storage Systems, physical space, personnel, personnel specialization, etc.).

The solution of these problems can be virtualizing the processing resources, the servers, installing virtualizing platform. This application ensures the centralized management of the entire computerized communication activity will decide the dynamic allocation of activities on the physical machines, will ensure system functionality in case of damage.

4. Conclusion

The market of virtualization solutions is increasing continuously even during the crisis period, especially because using virtualization solutions will become a necessity in order to turn nowadays costs into more effective ones and to survive in the market.

From the point of view of the home users, virtualization will mean the appearance of on-line operation systems, which means that the personal computer will use graphics cards and the motherboards online, on the Internet.

Regarding the small and medium enterprises, the next step in the IT industry is integrated management of calculating resources, network, stocking environments and virtualization in a unique system, the purposes being: maintaining perfection in the IT environment, increasing productivity and improving the companies' flexibility. Thus, the IT departments must become more than they are being, when they are only maintaining the system. Their role is to innovate.

References

[1] http://it.toolbox.com/wiki/index.php/Virtualization#References
[2] http://www.gartner.com/it/page.jsp?id=777212
[3] Edward L Haletky-VMware ESX Server in the Enterprise: Planning and Securing Virtualization Servers, Pearson Education, 2008
[4] Mitch Tulloch with the Microsoft Virtualization Team-Solution From the Desktop to the Datacenter, 2009
[5] Chris McCain-Mastering VMware Infrastructure 3, Wiley Publishing, Inc., Indianapolis, Indiana, 2008

$^{(1)}$ Pitesti, Arges
E-mail address: `cami.florea@yahoo.com`