FUTURE COMPUTING 2012

The Fourth International Conference on Future Computational Technologies and Applications

ISBN: 978-1-61208-217-2

July 22-27, 2012

Nice, France

FUTURE COMPUTING 2012 Editors

Dietmar Fey, University Erlangen-Nuremberg, Germany

Jaime Lloret Mauri, Polytechnic University of Valencia, Spain

FUTURE COMPUTING 2012

Foreword

The Fourth International Conference on Future Computational Technologies and Applications (FUTURE COMPUTING 2012), held between July 22 and 27, 2012 in Nice, France, targeted advanced computational paradigms and their applications. The focus was to cover (i) the advanced research on computational techniques that apply the newest human-like decisions, and (ii) applications on various domains. The new development led to special computational facets on mechanism-oriented computing, large-scale computing and technology-oriented computing. They are largely expected to play an important role in cloud systems, on-demand services, autonomic systems, and pervasive applications and services.

We take here the opportunity to warmly thank all the members of the FUTURE COMPUTING 2012 Technical Program Committee, as well as the numerous reviewers. The creation of such a broad and high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to FUTURE COMPUTING 2012. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the FUTURE COMPUTING 2012 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that FUTURE COMPUTING 2012 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the field of future computational technologies and applications.

We are convinced that the participants found the event useful and communications very open. We hope Côte d'Azur provided a pleasant environment during the conference and everyone saved some time for exploring the Mediterranean Coast.

FUTURE COMPUTING 2012 Chairs:

FUTURE COMPUTING Advisory Chairs
Cristina Seceleanu, Mälardalen University, Sweden
Hiroyuki Sato, The University of Tokyo, Japan
Miriam A. M. Capretz, The University of Western Ontario - London, Canada
Kendall E. Nygard, North Dakota State University - Fargo, USA
Vladimir Stantchev, Berlin Institute of Technology, Germany
Marek J. Druzdzel, University of Pittsburgh, USA

FUTURE COMPUTING 2012 Industry/Research
Francesc Guim, Intel Corporation, Spain
Wolfgang Gentzsch, Expert HPC, Germany

FUTURE COMPUTING 2012

Committee

FUTURE COMPUTING Advisory Chairs

Cristina Seceleanu, Mälardalen University, Sweden
Hiroyuki Sato, The University of Tokyo, Japan
Miriam A. M. Capretz, The University of Western Ontario - London, Canada
Kendall E. Nygard, North Dakota State University - Fargo, USA
Vladimir Stantchev, Berlin Institute of Technology, Germany
Marek J. Druzdzel, University of Pittsburgh, USA

FUTURE COMPUTING 2012 Industry/Research

Francesc Guim, Intel Corporation, Spain
Wolfgang Gentzsch, Expert HPC, Germany

FUTURE COMPUTING 2012 Technical Program Committee

Mohsen Askari, University of Technology Sydney, Australia
Jan Blech, fortiss GmbH, Germany
Radu Calinescu, Aston University-Birmingham, UK
Alberto Cano, University of Córdoba, Spain
Miriam A. M. Capretz, The University of Western Ontario - London, Canada
Massimiliano Caramia, University of Rome "Tor Vergata", Italy
Chin-Chen Chang, Feng Chia University, Taiwan, R.O.C.
Zhihua Cui, Taiyuan University of Science and Technology - Shanxi, China
Marc Daumas, INS2I & INSIS (CNRS), France
Isabel Maria de Sousa de Jesus, ISEP-Institute of Engineering of Porto, Portugal
Leandro Dias da Silva, Federal University of Alagoas, Brazil
Marek J. Druzdzel, University of Pittsburgh, USA
Francesco Fontanella, Università degli Studi di Cassino e del Lazio Meridionale, Italy
Wolfgang Gentzsch, Consultant, Germany
Victor Govindaswamy, Texas A&M University-Texarkana, USA
Michael Grottke, University of Erlangen-Nuremberg. Germany
Miguel Angel Guevara López, Universidade do Porto, Portugal
Francesc Guim, Intel Corporation, USA
Yongjian Hu, University of Warwick - Coventry, UK
Muhammad Iftikhar, Universiti Malaysia Sabah (UMS), Malaysia
Dalia Kriksciuniene, Vilnius University, Lithuania
Carlos León de Mora, Universidad de Sevilla, Spain
Lu Liu, University of Derby, UK
Yu-lung Lo, Chaoyang University of Technology, Taiwan
José María Luna, University of Córdoba, Spain

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

Hardware Accelerator for Low-Latency Privacy Preserving Mechanism

Junichi Sawada and Hiroaki Nishi
Graduate School of Science and Technology, Keio University
Kanagawa, Japan
Email: sawada@west.sd.keio.ac.jp, west@sd.keio.ac.jp

***Abstract*—With the recent growth in the quantity and value of data, data holders have come to realize the importance of being able to utilize information that is otherwise abandoned or concealed. In this situation, they face the difficulty of publishing data without revealing private information. Two of the methods used to protect private information when publishing data are privacy-preserving methods based on constraints known as k-anonymity and l-diversity. These methods enable the utilization of published data while preserving privacy, but incur a large computational cost. We solve this problem using a hardware architecture composed of Ternary Content Addressable Memory (TCAM), which can significantly accelerate the privacy preservation process. k-anonymity and l-diversity have not been studied in any significant way for efficient hardware implementation. Thus, this will be the first of its kind. An evaluation proves that an implementation of the proposed architecture on a reconfigurable device performs approximately 10-50 times faster than a RAM-based architecture.**

Keywords-hardware; reconfigurable device; privacy-preserving data publishing.

I. Introduction

Recently, the spread of Web services such as social networking services, blogs, and Internet shopping has emphasized the importance of users' information on Web servers and databases. Ubiquitous devices, sensor networks, and RFIDs will accelerate this situation. These types of applications generate information, including trends, which is valuable for service providers, social researches, and marketing. In this situation, data holders intend to share and utilize information that is otherwise abandoned or concealed.

In this situation, data holders face the difficulty of publishing data without revealing private information. Beyond the current methods of protecting against external cyber attacks, new methods of protecting private information when publishing data are needed. One of these is a privacy-preserving method based on a statistically proved constraint such as k-anonymity [1] or l-diversity [2] that enables data users to utilize published data under the constraint of preserving privacy. However, this method has a high processing cost, which makes it difficult to process high-throughput data streams such as the output of a database or network traffic that has been kept generated without pausing.

One of promising approaches to improve the performance is hardware implementation. We propose a TCAM-based hardware architecture for accelerating k-anonymity and l-diversity methods. These methods have not been studied in any significant way for efficient hardware implementation. Thus, this will be the first of its kind.

II. Related Work

Techniques to extract useful information without revealing privacy have been proposed for privacy-preserving data mining (PPDM) [3]. PPDM extracts useful information such as statistics and associations from more than one database with their secrecy preserved. In particular, the protection of private information when publishing data is called privacy-preserving data publishing (PPDP) [4], which is different from PPDM because it does not involve data mining. Two PPDP techniques are methods based on constraints, known as k-anonymity [1] and l-diversity [2], which are achieved by generalizing and suppressing "unique" data.

In recent years, many methods, especially for k-anonymity, have been studied. The main focus of some works is on privacy-preserving publishing of not static data, but dynamic data set, where new data can be added [5][6]. In this scenario, mainly two problems emerge; One is that the republication of the entire data set is needed whenever new data are added, and the other is the malicious inference available by analyzing the multiple versions of published data sets. To solve these problems, incremental update methods, which efficiently insert new data into the current data set without making it vulnerable, were proposed.

A clustering-based method based on k-anonymity for data streams was proposed with an eye on applications that need continuous privacy-preserving data publishing such as the publishing of telephone/network service records for network-traffic analysis, a search engine publishing a query log for online Web mining, and a stock exchange publishing its transactions [7]. Our approach is not based on clustering, and we focus on maintaining data in input order, which is important for some applications. It is different from those researches. Furthermore, our approach is based on hardware acceleration. The parallelism of our architecture efficiently accelerates the privacy preservation process.

The calculations of k-anonymity and l-diversity generally require a large cost. It has been shown that an optimal calculation of k-anonymity, which means results with the minimum information loss, is NP-hard [8]. The calculation of k-anonymity or l-diversity can be completed by repeatedly comparing each record against every other record as

 ISBN: 978-1-61208-217-2

an association-rule mining. This calculation requires a time complexity of $O(n^2)$ in the worst case for k-anonymity and in all cases for l-diversity. To improve performance, a promising approach is to exploit advanced hardware. This approach includes an implementation with network processors [9], exploitation of GPGPU or GPUTeraSort [10], improved performance of join with Cell/B.E. [11], and the implementation of special purpose hardware for stream data operations with FPGA [12][13][14][15].

The k-anonymity and l-diversity methods have not been studied in any significant way for efficient hardware implementation. Researches on the hardware implementation of an association-rule mining algorithm have been published [16][17][18]. These researches may be efficient for the k-anonymity and l-diversity methods in finding infrequent records that do not satisfy k or l. However, the requirements are different for k-anonymity or l-diversity and association-rule mining, as follows: 1. The comparison is computed for a whole record. It is not necessary to calculate the combination of elements in a record. 2. An implementation of generalization is needed. 3. Association-rule mining finds frequent records, whereas infrequent records are needed in the calculation of k and l. Moreover, association-rule mining finds a rule, not a record itself. Thus, after finding rules, another process may be needed to find records that match the extracted rules. Therefore, another implementation optimized for the privacy-preserving algorithm would be more effective than a hardware implementation of the association-rule mining algorithm.

III. Privacy Protection Model

k-anonymity and l-diversity are satisfied by using a generalization that replaces a value with a less specific, more general value, or masks a part of the value with "*." The special terms used in this paper are defined according to [1][2] as follows:

Data Table: A row is termed a tuple and a column is termed a field. Each field is said to be an attribute, which indicates the meaning of values.

Attribute: Attributes that can uniquely identify individuals such as names are termed explicit identifiers. Other attributes that in combination can uniquely identify individuals such as their birth date, ZIP, and gender are termed quasi identifiers.

Sensitive Attribute: Attributes that are not termed quasi identifiers and should not be associated with an individual, for example "diagnosis" in the case of medical data, are termed sensitive attributes. Other attributes, which are possibly the same as the quasi identifiers, are termed non-sensitive attributes and will be generalized. When the values of non-sensitive attributes are the same or have been generalized to the same values, the set of tuples is termed a *q*-block*.

Domain Generalization Hierarchy (DGH): A Domain Generalization Hierarchy (DGH) refers to a hierarchy that indicates how and how many times an attribute is generalized. For example, Figure 1 shows the DGH of the attribute "birth date."

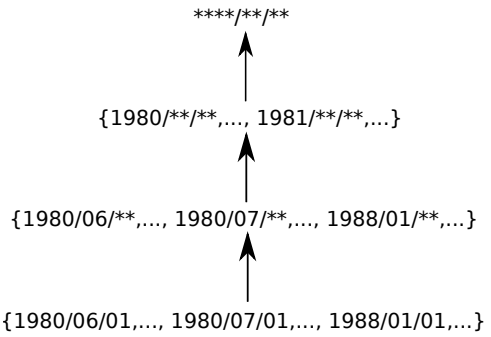


Figure 1. Birth date domain generalization hierarchy

Table I
DATA TABLE WITH SENSITIVE ATTRIBUTE "PROBLEM"

Gender	Birth	ZIP	Problem
Male	1963	02150	short breath
Male	1960	02140	chest pain
Male	1964	02138	chest pain
Male	1964	02138	obesity
Male	1964	02138	short breath

Table II
2-ANONYMIZED TABLE WITH SENSITIVE ATTRIBUTE "PROBLEM"

Gender	Birth	ZIP	Problem
Male	196*	021**	short breath
Male	196*	021**	chest pain
Male	1964	02138	chest pain
Male	1964	02138	obesity
Male	1964	02138	short breath

A. k-anonymity

A table T satisfies k-anonymity if each sequence of non-sensitive values in T appears with at least k occurrences.

Table I with the sensitive attribute "problem" is generalized into Table II where $k = 2$, as an example. Because there are at least 2 of the same tuples for each tuple in Table II, an adversary cannot distinguish one tuple from another in any *q*-block*. k-anonymity guarantees that a tuple cannot be distinguished from at least $k-1$ other tuples in the table.

However, k-anonymity may leak private information in a few cases, as mentioned in [2]. In a case where the sensitive values are all the same in a *q*-block*, an adversary can infer the sensitive value even if he cannot distinguish the tuple. In another case where an adversary has a strong background knowledge about the sensitive values, he may be able to infer non-sensitive values from the sensitive values. These vulnerabilities are both caused by a lack of diversity in the sensitive values. To solve this problem, l-diversity [2] has been proposed, as stated below.

Table III
GENERALIZED VALUE EXPRESSION IN TCAM

Value	TCAM	
	Data	Mask
0011****	00110000	00001111
00111***	00111010	00000111

Table IV
GENERALIZATION PROCESS IN TCAM

Value		TCAM	
Attribute 1	Attribute 2	Data	Mask
0000	0000	0000_0000	0000_0000
	↓generalize		
000*	000*	0000_0000	0001_0001
	↓generalize		
00**	00**	0000_0000	0011_0011

B. l-diversity

Table T satisfies l-diversity if every *q*-block* has at least l different values for a sensitive attribute.

Assume that the ith most frequent sensitive value appears r_i times and n types of sensitive values appear in a *q*-block*; l-diversity is defined by equation 1.

$$r_1 \leq r_l + r_{l+1} + ... + r_n \tag{1}$$

IV. PROPOSED ARCHITECTURE

A. TCAM

A hardware implementation of the k-anonymity and l-diversity method requires the following things:

- A fast search function for infrequent tuples that do not satisfy k-anonymity or l-diversity
- Implementation of the generalization, which means a search function compatible with "*," i.e., a wild card

These requirements can be achieved using Ternary Content Addressable Memory (TCAM). CAM is a memory that receives data as input and outputs the locations where the associated contents are stored. Comparison logic for each cell enables a search operation to be completed in a single memory access. By using CAM, it becomes clear whether or not the required privacy level is satisfied in CAM with the complexity $O(n)$. Additionally, TCAM has a mask circuit that allows a third matching state of "Don't care" for each cell for various-length matching. The generalization can be processed with this circuit as shown in Table III.

By shifting mask bits, data can be generalized from the bottom bit-by-bit, as shown in Table IV.

The calculation of k-anonymity is completed by counting the number of tuples that are the same as the one specified in a search. TCAM can complete the calculation in a single cycle by searching for the tuple. Unfortunately, the calculation of l-diversity requires a process that is not as simple as that for k-anonymity because a calculation of the frequencies of the values of sensitive attributes in a *q*-block* is required. Based on the l-diversity principle expressed by equation 1, if the probability of the occurrence of a value for a sensitive attribute in a *q*-block* is less than $1/l$, a tuple that has that value has absolutely satisfied l-diversity. Thus, the probability of the occurrence of a value for a sensitive attribute in a *q*-block* can be written as $p(s|q*)$, where s denotes the sensitive value, and if the probability satisfies equation 2, the tuple is said to satisfy l-diversity.

$$p(s|q*) \leq \frac{1}{l} \tag{2}$$

Our approach to calculate the probability is to utilize variable-length matching just as in the generalization, which allows the calculation to be completed in two phases. In the first phase, a search operation is executed just as in the calculation of k-anonymity, and the total number of tuples that are the same as the one specified in the search is counted. The result of the first phase shows how many times a value for a sensitive attribute of the current tuple occurs in the *q*-block*. In the second phase, unlike the first phase, a search operation is executed with the values for the sensitive attribute masked. The result of the second phase shows the total number of tuples in a *q*-block* that include the tuple currently being processed. The probability $p(s|q*)$ of the tuple can be calculated using the two values obtained in these two phases.

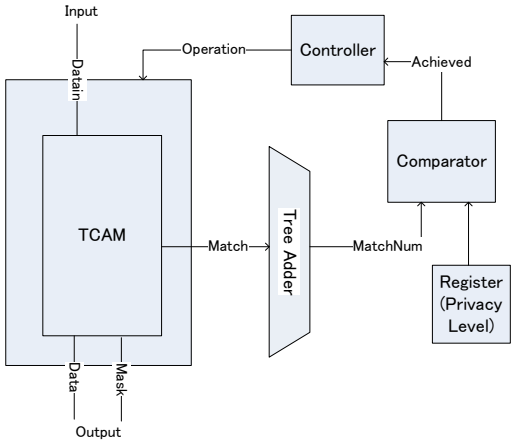


Figure 2. TCAM-based architecture for k-anonymity and l-diversity

B. Whole Architecture

Figure 2 shows the hardware architecture proposed in this paper. The TCAM has a match line for each entry and outputs are connected to the tree adder, which counts up the total number of asserted match lines. The comparator compares the input with the previously configured privacy level, and the output indicates whether or not the privacy

level is satisfied. The controller manages the write and read addresses, and shifts the generalization process from the current tuple to the next tuple after completing the generalization, which is necessary if the privacy level of the current tuple is not satisfied. Finally, all of the sets of data and mask values are output when all of the tuples in TCAM satisfy the privacy level. The algorithm is executed as stated below.

1) Input data into the TCAM
2) Search the TCAM for a tuple
3) Generalize the tuple if its privacy level is not satisfied
4) Search for the next tuple
5) Repeat step 3 and 4 until all of the tuples in the TCAM satisfy the required privacy level
6) Output all sets of data and mask values in the TCAM and go back to step 1

The details of this algorithm are described in the following.

Algorithm Calculation of k and l

```
1:  loop
2:    /*Input*/
3:    while TCAM is not full do
4:      write_tuple_to_TCAM;
5:    end while
6:    /*Calculation of k and l*/
7:    repeat
8:      flag ← 0;
9:      for each tuple in TCAM do
10:       search_for_tuple;
11:       if k or l is not achieved then
12:         generalize_tuple;
13:         flag ← 1;
14:       end if
15:     end for
16:   until flag is 0
17:   /*Output*/
18:   while TCAM is not empty do
19:     read_tuple_from_TCAM;
20:     result ← data | mask;
21:   end while
22: end loop
```

Because it is important to ensure the levels of tuples in the generalization hierarchy are as same as possible in computing the algorithm, the search operation at each tuple is executed only once per loop, and the loop is repeated until all of the tuples in the TCAM satisfy the required privacy level. In the output process, if the output is only one tuple, the whole transaction, including tuples that have already been output, may not satisfy l-diversity. This is why the proposed architecture exchanges tuples perfectly. In this case, because it is the same to process a divided transaction one-by-one, the architecture can work in parallel.

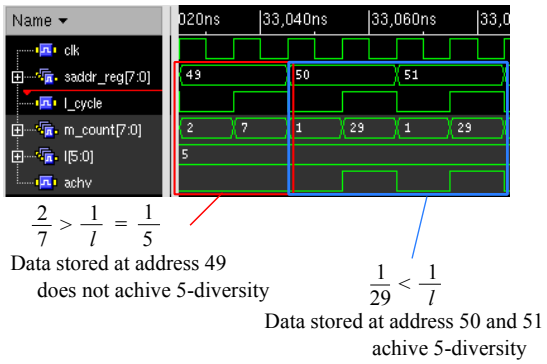


Figure 3. Observed waveforms

V. RESULTS

A. Throughput

The proposed architecture is implemented on a Xilinx Virtex5 FPGA board (XC5VLX330T) using the tool Xilinx ISE 12.3. Because the TCAM cannot be implemented using Block RAM and requires a large hardware cost, the TCAM IP core provided by Xilinx is implemented to improve resource utilization. Additionally, although the TCAM needs to output data and mask values in the proposed architecture, the TCAM IP core has no output ports for stored values. To solve this problem, the proposed architecture is emulated by storing data and mask values into not only TCAM but also Block RAMs. In the case of the 256x256 TCAM, which is the maximum entry size on XC5VLX330T, hardware usage is 45,691 LUT FF pairs and the maximum frequency is approximately 70 MHz. Figure 3 shows a capture of a SimVision waveform window. The signals described in this figure are as follows: 1. saddr_reg indicates a memory address where a tuple currently being processed is stored. 2. l_cycle indicates the two phases of the calculation of l-diversity described in IV-A. 3. m_count indicates the output of the tree adder, namely the total number of matched tuples. 4. l indicates the required l. 5. achv indicates whether the current tuple achieves l-diversity.

The throughput evaluation is performed by processing Internet traffic, specifically browsing history obtained at our laboratory to simulate a trend survey on the Web. The destination IP address is the quasi identifier and used for the non-sensitive attribute, while the Web title is the sensitive attribute. The destination IP address is processed as 32 bit data, and generalized one bit by one bit, up to 32 times. Table V shows the data layout of the data set used for this evaluation.

Figure 4 gives a throughput comparison between the proposed architecture and RAM-based architecture where the search operation is serialized. The comparison is performed with l-diversity, which requires a larger processing cost than k-anonymity.

Because the throughput is calculated by assuming that a

Table V
DATA LAYOUT OF THE DATA SET USED FOR THE EVALUATION

Destination IP address (actually 32 bit data)	Web title
012.XXX.XXX.XXX	Google
012.XXX.XXX.XXX	Facebook
123.XXX.XXX.XXX	Google
234.XXX.XXX.XXX	FIFA.com

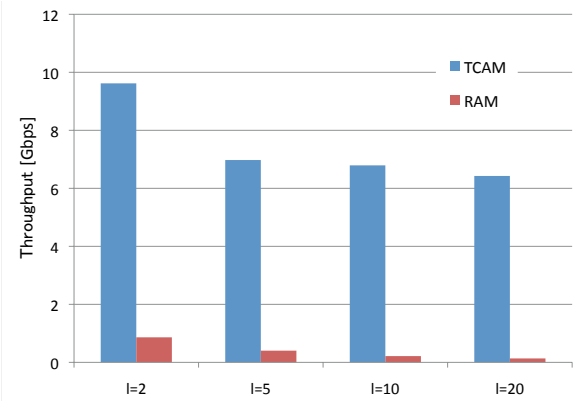


Figure 4. Performance comparison against RAM-based architecture

tuple corresponds to a packet, and the average packet size is 1,000 bytes, the architecture can process Internet traffic, in this case the browsing history in a network of at least 1 Gbps. In the case of another environment, more evaluations are needed.

In this evaluation, both the TCAM-based architecture and the RAM-based architecture process the exact same dataset. Thus, the experimental result shown in Figure 4 means that the TCAM-based architecture performs approximately 10-50 times faster than the RAM-based architecture. The CAM takes advantage of its specialty with the increase of l, as shown in the throughput differences between the two architectures, because the number of processes increases. Moreover, compared to software implementation (C++, single-threaded, 3.0 GHz Quad-Core Xeon CPUx2, 8GB DDR3) of the same algorithm, it achieves approximately 400-900 times higher throughput, as shown in Figure 5. This evaluation is also performed by processing the same dataset in the same way.

B. Information Loss

A generalized table typically has less useful information. In order to evaluate the information loss, the theoretic metric information loss (IL) of data table T, written $IL(T)$, is defined by referring to $Prec$ in [1]. Let $t \in T = \{t_1, ..., t_N\}$ be a tuple, $QI_T = \{A_1, ..., A_{N_A}\}$ be the quasi identifier, DGH be a height of a generalization hierarchy, and h be a height of a generalized value in a generalization hierarchy.

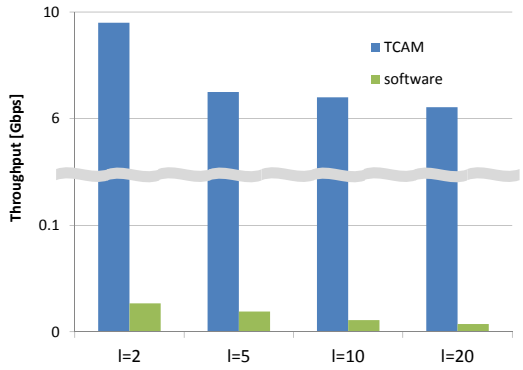


Figure 5. Performance comparison against software implementation

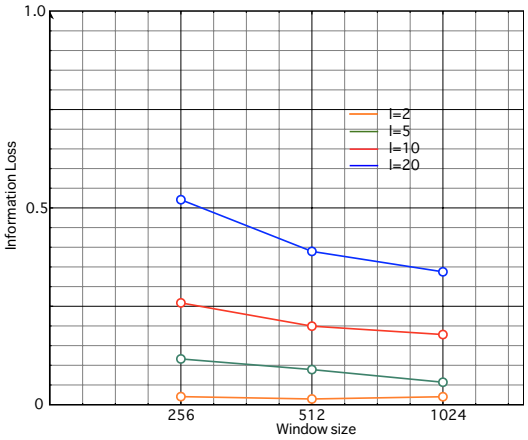


Figure 6. Trade-off between information loss and window size

Then, the equation is defined as 3.

$$IL(T) = \frac{\sum_{t_j \in T} \sum_{A_i \in QI_T} \frac{h}{|DGH_{A_i}|}}{|T| \cdot |QI_T|} = \frac{\sum_{j=1}^{N} \sum_{i=1}^{N_A} \frac{h}{|DGH_{A_i}|}}{N \cdot N_A} \quad (3)$$

IL indicates how deep a value has been generalized with a value from 0 to 1.

To process data stream, it has to be divided in windows and processed one-by-one. Because it is difficult to satisfy a privacy level in a small window size, a trade-off exists between the information loss and a window size as shown in the graph in Figure 6.

Focusing on the utility of published information, smaller k or l would be chosen as the privacy level because it will result in smaller IL and useful information. When l is small, the increase in IL caused by the window-size constraint is also small as shown in Figure 6. In that case, the window size can be small without increasing IL, and that results in low hardware cost.

 ISBN: 978-1-61208-217-2

VI. Conclusion

A hardware architecture for the privacy-preserving algorithm based on constraints known as k-anonymity and l-diversity was proposed. The work was based on the TCAM, which enables a fast search operation. A generalization process, which is necessary for the calculation of the algorithm, was also efficiently enabled with variable-length matching. Overall, the FPGA implementation of the proposed architecture performed approximately 10-50 times faster than a RAM-based architecture where the search operation was serialized.

References

[1] L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, vol. 10, no. 5, pp. 571–588, 2002.

[2] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "L-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, March 2007.

[3] Y. Lindell and B. Pinkas, "Privacy Preserving Data Mining," *Journal of Cryptology*, vol. 15, pp. 177–206, 2002.

[4] B.-C. Chen, D. Kifer, K. LeFevre, and A. Machanavajjhala, "Privacy-Preserving Data Publishing," *Foundations and Trends in databases*, vol. 2, pp. 1–167, January 2009.

[5] J.-W. Byun, Y. Sohn, E. Bertino, and N. Li, "Secure anonymization for incremental datasets," in *Secure Data Management*, ser. Lecture Notes in Computer Science, 2006, vol. 4165, pp. 48–63.

[6] X. Xiao and Y. Tao, "M-invariance: towards privacy preserving re-publication of dynamic datasets," in *Proceedings of the 2007 international conference on Management of data*, ser. SIGMOD '07. ACM, 2007, pp. 689–700.

[7] B. Zhou, Y. Han, J. Pei, B. Jiang, Y. Tao, and Y. Jia, "Continuous privacy preserving publishing of data streams," in *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, ser. EDBT '09. ACM, 2009, pp. 648–659.

[8] A. Meyerson and R. Williams, "On the complexity of optimal k-anonymity," in *Proceedings of the 23rd SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, ser. PODS '04. ACM, 2004, pp. 223–228.

[9] B. Gold, A. Ailamaki, L. Huston, and B. Falsafi, "Accelerating database operators using a network processor," in *Proceedings of the 1st international workshop on Data management on new hardware*, ser. DaMoN '05. ACM, 2005.

[10] N. Govindaraju, J. Gray, R. Kumar, and D. Manocha, "GPUTeraSort: high performance graphics co-processor sorting for large database management," in *Proceedings of the 2006 international conference on Management of data*, ser. SIGMOD '06. ACM, 2006, pp. 325–336.

[11] B. Gedik, P. S. Yu, and R. R. Bordawekar, "Executing stream joins on the cell processor," in *Proceedings of the 33rd international conference on Very large data bases*, ser. VLDB '07. VLDB Endowment, 2007, pp. 363–374.

[12] R. Mueller, J. Teubner, and G. Alonso, "Streams on wires: a query compiler for FPGAs," *Proc. VLDB Endow.*, vol. 2, no. 1, pp. 229–240, 2009.

[13] ——, "Data processing on FPGAs," *Proc. VLDB Endow.*, vol. 2, pp. 910–921, August 2009.

[14] L. Woods, J. Teubner, and G. Alonso, "Complex Event Detection at Wire Speed with FPGAs," *Proc. VLDB Endow.*, vol. 3, pp. 660–669, September 2010.

[15] J. Teubner and R. Mueller, "How soccer players would do stream joins," in *Proceedings of the 2011 international conference on Management of data*, ser. SIGMOD '11. ACM, 2011, pp. 625–636.

[16] Z. Baker and V. Prasanna, "Efficient hardware data mining with the Apriori algorithm on FPGAs," in *13th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, 2005. FCCM '05*, April 2005, pp. 3 – 12.

[17] ——, "An Architecture for Efficient Hardware Data Mining using Reconfigurable Computing Systems," in *14th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, 2006. FCCM '06*, April 2006, pp. 67 –75.

[18] Y.-H. Wen, J.-W. Huang, and M.-S. Chen, "Hardware-Enhanced Association Rule Mining with Hashing and Pipelining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 6, pp. 784 –795, June 2008.

 ISBN: 978-1-61208-217-2

Distributed-Shared CUDA: Virtualization of Large-Scale GPU Systems for Programmability and Reliability

Atsushi Kawai, Kenji Yasuoka
Department of Mechanical Engineering,
Keio University
Yokohama, Japan
Email: kawai@kfcr.jp, yasuoka@mech.keio.ac.jp

Kazuyuki Yoshikawa, Tetsu Narumi
Department of Informatics and Engineering,
University of Electro-Communications
Tokyo, Japan
Email: kaz82019@jed.uec.ac.jp, narumi@cs.uec.ac.jp

***Abstract*—One of the difficulties for current GPGPU (General-Purpose computing on Graphics Processing Units) users is writing code to use multiple GPUs. One limiting factor is that only a few GPUs can be attached to a PC, which means that MPI (Message Passing Interface) would be a common tool to use tens or more GPUs. However, an MPI-based parallel code is sometimes complicated compared with a serial one. In this paper, we propose DS-CUDA (Distributed-Shared Compute Unified Device Architecture), a middleware to simplify the development of code that uses multiple GPUs distributed on a network. DS-CUDA provides a global view of GPUs at the source-code level. It virtualizes a cluster of GPU equipped PCs to seem like a single PC with many GPUs. Also, it provides automated redundant calculation mechanism to enhance the reliability of GPUs. The performance of Monte Carlo and many-body simulations are measured on 22-node (64-GPU) fraction of the TSUBAME 2.0 supercomputer. The results indicate that DS-CUDA is a practical solution to use tens or more GPUs.**

***Keywords*-GPGPU; CUDA; distributed shared system; virtualization.**

I. INTRODUCTION

Optimization of communication among several hundreds of thousands of CPU cores is one of the main concerns in high performance computing. The largest supercomputer [1] has nearly a million cores, on which the communication tends to be the bottleneck instead of the computation.

On modern massively parallel systems, several (typically 4–16) CPU cores in one processor node share the same memory device. The design of a program should take this memory hierarchy into account. A naive design that assigns one MPI (Message Passing Interface) process to each core causes unnecessary communication among cores in the same node. In order to avoid this inefficiency, a hybrid of MPI and OpenMP is often used, where each OpenMP thread is assigned to one core, and one MPI process is used per node.

Moreover, a significant fraction of recent top machines in the TOP500 list [2] utilize GPUs. For example, the TSUBAME 2.0 supercomputer [3] consists of 1,408 nodes, each containing 3 NVIDIA GPUs. In order to program GPUs, frameworks such as CUDA [4] or OpenCL [5] are necessary. Therefore, a program on massively parallel systems with GPUs needs to be written using at least three frameworks, namely, MPI, OpenMP, and CUDA (or OpenCL).

However, even a complicated program using all three frameworks may fail to take full advantage of all the CPU cores. For example, if one thread is assigned to a core to control each GPU, only a few cores per PC would be in use, since only a few GPUs can be attached to a PC. There are typically more cores than GPUs on a single node, and utilizing these remaining cores is also important.

We propose a Distributed-Shared CUDA (DS-CUDA) framework to solve the major difficulties in programming multi-node heterogeneous computers. DS-CUDA virtualizes all GPUs on a distributed network as if they were attached to a single node. This significantly simplifies the programming of multi-GPU applications.

Another issue that DS-CUDA addresses is reliability of GPUs. Consumer GPUs such as GeForce sometimes are prone to memory errors due to the lack of ECC (Error Check and Correct) functions. Hamada *et al.* [6] reported around a 10% failure rate for one week of execution on GeForce GTX 295 cards. Furthermore, even with server-class GPUs, erroneous code may cause faulty execution of successive parts of the program, which is difficult to debug on a multi-GPU environment. DS-CUDA increases the reliability of GPUs by a built-in redundancy mechanism.

The virtualization of multi-GPUs in DS-CUDA also alleviates the increased burden to manage heterogeneous hardware systems. For example, some nodes in a GPU cluster might have a different number of GPUs than others, or even no GPUs. With DS-CUDA the user no longer needs to worry about the number of GPUs on each node.

A key concept of DS-CUDA is to provide a global view of GPUs for CUDA based programs. Global view of distributed memory is one of the key features of next generation languages, such as Chapel [7] and X10 [8]. These languages greatly reduce the complexity of the program compared to MPI and OpenMP hybrid implementations. However, they do not provide the global view on GPUs, since GPUs can only be accessed through dedicated APIs such as CUDA and OpenCL.

 ISBN: 978-1-61208-217-2

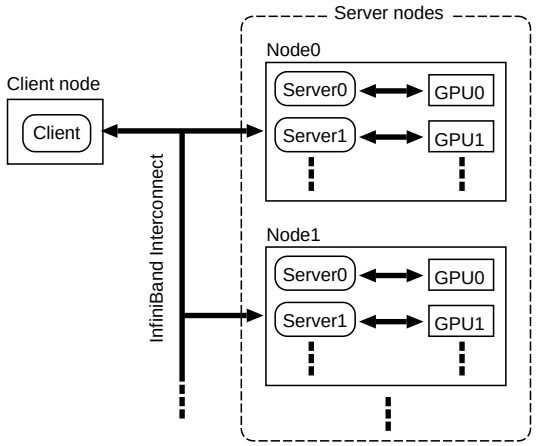


Figure 1. The structure of a typical DS-CUDA system. The client program using CUDA runs on a client node, while DS-CUDA server programs run on each server node for each GPU.

Similar ideas for virtualizing GPUs have been implemeted in rCUDA [9][10], vCUDA [11], gVirtuS [12], GViM [13], and MGP [14]. However, vCUDA, gVirtuS and GViM virtualize the GPUs to enable the access from a virtual machine in the same box, and they do not target remote GPUs. MGP is a middleware to run OpenCL programs on remote GPUs. Their idea is similar to ours, but they only support OpenCL, and not CUDA. rCUDA is also a middleware to virtualize remote GPUs, and it supports CUDA. In this sense, rCUDA is quite similar to DS-CUDA. The primary difference between rCUDA and DS-CUDA is that the former aims to reduce the number of GPUs in a cluster for lowering construction and maintenance cost, while the latter aims to provide a simple and reliable solution to use as many GPUs as possible. To this end, DS-CUDA incorporates a fault tolerant mechanism, that can perform redundant calculations by using multiple GPUs. Errors are detected by comparing the results from multiple GPUs. When an error is detected, it automatically recovers from the error by repeating the previous CUDA API and kernel calls until the results match. This fault tolerant function is hidden from the user.

In this paper, we propose a DS-CUDA middleware. In Section II, the design and implementation are explained. In Section III, the performance measured on up to 64 GPUs is shown. In Section IV, conclusions and future work are described.

II. Implementation

In this section, we describe the design and implementation of DS-CUDA.

A. System Structure

The structure of a typical DS-CUDA system is depicted in Figure 1. It consists of a single client node and multiple server nodes, connected via InfiniBand network.

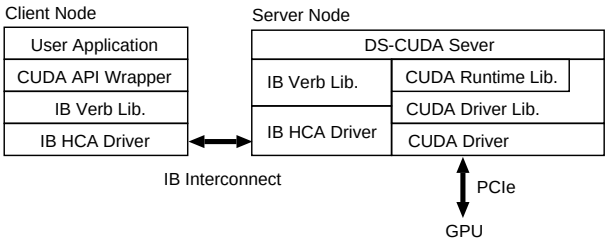


Figure 2. The software-layer stack. On the client node, the user application program calls CUDA API wrappers instead of native CUDA APIs. On server node, DS-CUDA server calls actual cuda APIs. Communication between client and server nodes are performed by InfiniBand Verbs by default.

Each server node has one or more CUDA devices (*i.e.*, GPUs), each of which is handled by a server process. An application running on the client node can utilize these devices by communicating with the server node over the network.

B. Software-Layer Stack

Figure 2 shows the software-layer stack of both the client and server node. On the client node, the application program is linked to the DS-CUDA client library. The library provides CUDA API wrappers, in which the procedure to communicate with the servers are included. Therefore, the CUDA devices on the server nodes can be accessed via usual CUDA APIs, as if they were locally installed.

By default, the client-server communication uses InfiniBand Verbs, but can also use TCP sockets in case the network infrastructure does not support InfiniBand.

C. CUDA C/C++ Extensions

Access to CUDA devices are usually done through CUDA API calls, such as `cudaMalloc()` and `cudaMemcpy()`. As mentioned above, these are replaced with calls to CUDA API wrappers, which communicate with the devices on the server nodes.

There are, however, several exceptions. Some CUDA C/C++ extensions, including calls to CUDA kernels using triple angle brackets, *e.g.*, `myKernel<<<g,b>>>(val,...)`, access the CUDA devices without explicit calls to CUDA APIs.

The DS-CUDA preprocessor, `dscudacpp` handles CUDA C/C++ extensions. Figure 3 summarizes the procedure: In order to build an executable for the application program, the source codes are fed to `dscudacpp`, instead of usual `nvcc`. The sources are scanned by `dscudacpp`, and the CUDA C/C++ extensions are replaced with remote calls which load and launch the kernel on the server side. Then, it passes the modified sources on to the C compiler `cc`. Meanwhile, `dscudacpp` retrieves the definitions of kernel functions and passes them on to `nvcc`. The kernel functions are compiled by `nvcc`, and kernel modules are

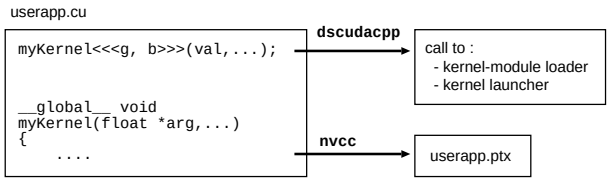


Figure 3. CUDA C/C++ Extensions are preprocessed by `dscudacpp`. GPU kernel `myKernel` is compiled by native `nvcc` compiler and also converted by `dscudacpp` to calls to the kernel-module loader and kernel launcher.

generated in `.ptx` format. During the execution of the application program, the kernel modules in the client node are transferred to the server nodes upon request.

D. Virtual Device Configuration

The application program sees virtual devices that represent real devices on the server nodes via the CUDA API wrapper. The mapping of the real to virtual devices is given by an environment variable `DSCUDA_SERVER`. Table I gives an example of the mapping.

```
sh> export DSCUDA_SERVER= \
    "node0:0 node0:1,node1:0 node1:1"
```

Device0 on Server Node0 is mapped to the virtual Device0, Device1 of Node0 and Device0 of Node1 are mapped to virtual Device1, and so on. Note that two devices, Device1 of Node0 and Device0 of Node1, are mapped to a single virtual Device1, that represents a 2-way *redundant device*. The mechanism of redundant devices will be described in the next section.

E. Fault-Tolerant Mechanism

A virtual device can have redundancy, in order to improve reliability of the calculations performed on the device. That is, multiple CUDA devices on the server nodes can be assigned to a single virtual device on the client node. Identical calculations are performed on the redundant devices, and the results are compared between the redundant calculations. If any of the results do not match, the client library invokes an error handler.

By default, the handler tries to recover from the error. It reproduces all CUDA API calls after the latest successful call to `cudaMemcpy()` of transfer type `cudaMemcpyDeviceToHost`. The application program may override this behavior, if it is not desirable.

F. Functional Restrictions

Although DS-CUDA provides transparent access to CUDA devices over the network, its function is not fully compatible with the native CUDA framework. The current version has the following restrictions:

(a) The graphics relevant APIs, such as OpenGL and Direct3D interoperability, are not supported.

Table I
AN EXAMPLE OF A MAPPING OF REAL DEVICES ON THE SERVER NODES TO VIRTUAL DEVICES ON THE CLIENT NODE.

Client-side virtual device	Server-side real device
Device 0	Device 0 of node0
Device 1	Device 1 of node0 and device 0 of node1 (2-way redundancy)
Device 2	Device 1 of node1

(b) Only CUDA Toolkit 4.0 is supported.
(c) Some capabilities to set memory attributes, including page lock and write combining, are not supported.
(d) Asynchronous APIs are implemented as aliases to their synchronous counterparts.
(e) Only the CUDA Runtime APIs, a part of CUFFT and a part of CUBLAS are supported. The CUDA Driver APIs are not supported.

The graphics APIs listed in (a) are meaningless for remote devices, whose video outputs are not used. Rest of the restricted capabilities, (b)–(e), are planned to be supported in the near future.

III. MEASURED PERFORMANCE

In this section, we show performances of the DS-CUDA system measured on a fraction of the TSUBAME 2.0 [3] GPU cluster. The fractional system consists of 22 nodes, each houses two Intel Xeon processors (X5670) and three NVIDIA Tesla GPUs (M2050, x16 Gen2 PCI Express, 64Gbps).

In Section III-A, some measurements on our prototype GPU cluster are also shown. The cluster consists of 8 nodes, each houses an Intel Core i5 processor (i5-2500) and an NVIDIA GeForce GPU (GTX 580, x16 Gen1 PCI Express, 32Gbps).

In the both systems, the nodes are connected via InfiniBand (X4 QDR, 40Gbps) network.

A. Data Transfer

Here, we show data transfer speeds of the `cudaMemcpy()` wrapper function. Curves labeled "IB Verb" in Figure 4 show the results. The left and right panels are for our prototype system and the TSUBAME 2.0, respectively. For comparison, the speeds of native `cudaMemcpy()` are also shown as "local" curves.

On the both systems, the effective bandwidth is around 1GB/s for large enough data size ($\geq$ 100kB). This speed is marginal for some productive runs as shown in later sections and [9]. Also, we should note that we still have room for improvement in the effective bandwidth. In [10], the effective bandwidth of nearly 3GB/s is reported. The high data bandwidth is achieved by eliminating the main memory to main memory copy using *GPU direct* [15] technology. Furthermore, they overlapped the network transfer and memory copy.

 ISBN: 978-1-61208-217-2

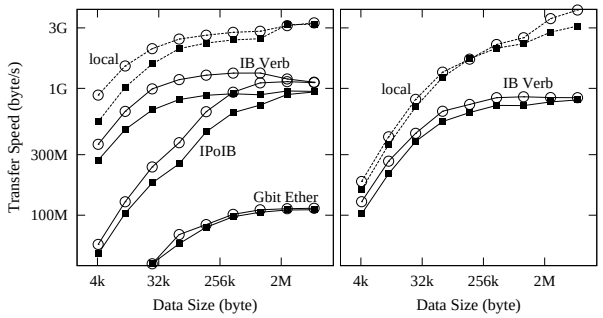


Figure 4. Performance of the `cudaMemcpy()` wrapper function. Data transfer speeds are plotted against data payload size. The left panel shows the results on the prototype system, and the right shows that on the TSUBAME 2.0. Curves with open circles are results for transfer from the client to the server (transfer type `cudaMemcpyHostToDevice`), those with filled squares are from the server to the client (`cudaMemcpyDeviceToHost`).

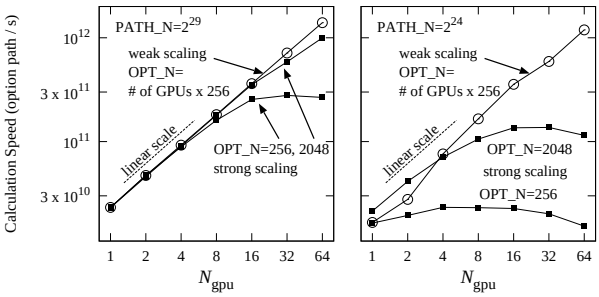


Figure 5. Performance of `MonteCarloMultiGPU`, an implementation of the Monte Carlo approach to option pricing, included in the CUDA SDK. Calculation speeds are plotted against the number of GPU devices, $N_{\rm gpu}$. The number of options calculated for is denoted by `OPT_N`. The number of integration paths is denoted by `PATH_N`. Weak scaling graphs show good scalability for both cases, but for strong scaling with smaller calculation cost shown in right panel is worse.

Poor performance at smaller data size ($\leq$ 100kB) is observed on the TSUBAME 2.0. We observe performance degradation for "local" curves, too. This is likely to be caused not by the network latency, but by the memory-access latency inside the server node.

Curves labeled "IPoIB" and "Gbit Ether" in the left panel are results with data transfer using the TCP socket over InfiniBand and over Gigabit Ethernet, respectively. These are shown just for comparison.

B. MonteCarloMultiGPU

Next, we show the performance of `MonteCarloMultiGPU`, an implementation of the Monte Carlo approach to option pricing, which is included in the CUDA SDK. In the source code, the number of options calculated for is given by a parameter `OPT_N`, and the number of integration paths is given by `PATH_N`. We measured the calculation speed for various combinations of `OPT_N`, `PATH_N` and the number of CUDA devices, $N_{\rm gpu}$, involved in the simulation.

Figure 5 shows the results. Calculation speeds (defined as the total number of paths processed per second) are plotted against $N_{\rm gpu}$. The left and right panels are for `PATH_N` $= 2^{29}$ and `PATH_N` $= 2^{24}$, respectively. Three curves in each panel are for runs with `OPT_N` fixed to 256 and 2048 in order to see strong scaling, and those with `OPT_N` scaled by $N_{\rm gpu}$ to see weak scaling.

In the left panel, the result shows 95% weak scaling. Although strong scaling goes down to 68% (`OPT_N`=2048) and 18% (`OPT_N`=256) at runs with 64 devices, they show better scalability for runs with smaller $N_{\rm gpu}$.

The results in the right panel also show ideal weak scalability. Strong scalings are, however, lower than 10% even with `OPT_N`=2048. In this case, runs with more than a few devices are not practical.

C. Many-Body Simulation

Now we show the performance of the simplest gravitational many-body simulation. In the simulation, gravitational forces from all point masses (hereafter we denote them "j-particle") are evaluated at the location of all point masses ("i-particle") by a naive $O(N^2)$ algorithm. In parallel runs, i-particles are split into fractions, and each of them are sent to one CUDA device, while all j-particles are sent to all devices.

Figure 6 shows the results. Calculation speeds (defined as the number of pairwise interactions between two particles calculated per second) are plotted against the number of devices, $N_{\rm gpu}$. The results with $N = 128$k shows fairly good scalability for up to 8 devices. Those with $N \leq 32$k scales only up to a few devices, in which case the locally installed 1–4 devices would be a better choice than DS-CUDA.

We should note that in production runs, fast algorithms such as the Barnes-Hut treecode [16] of $O(N \log N)$ and the FMM [17] of $O(N)$ are often used, whose performance exceed that of $O(N^2)$ algorithm at large N, say $\geq 10^5$. According to our preliminary result with a serial treecode, reasonable balance is achieved when one CUDA device is assigned to one CPU core. In that case the workload on the device is roughly equivalent to that of the $O(N^2)$ algorithm with $N/N_{\rm gpu}$ i-particles and 10k j-particles. Using the $O(N^2)$ parallel code, we measured the performance at that workload and found it scales well at least up to $N_{\rm gpu} = 8$.

D. Molecular Dynamics Simulation with Redundancy

We performed a molecular dynamics simulation of a NaCl system. Figure 7 shows the temperature against time. We used 512 atoms, and Tosi-Fumi potential [18] is used for the interaction between atoms. The initial temperature is set to 300 K and atom positions are integrated with a Leap-Frog method with a time-step of 0.5 fs for 2 ps (40,000

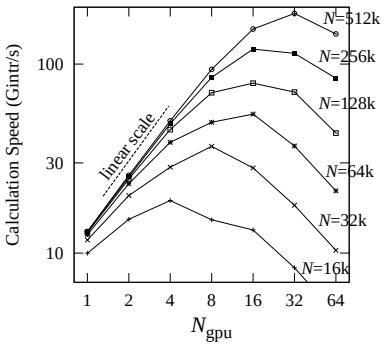


Figure 6. Performance of gravitational many-body simulations. The number of pairwise interactions between two particles calculated per second is plotted against the number of CUDA devices, N_{gpu}, for various number of particles in the system, N.

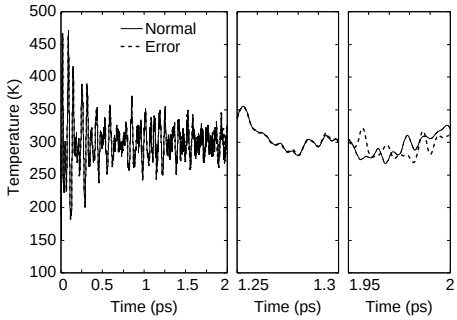


Figure 7. Temperature of a system from a molecular dynamics simulation is plotted against time (pico second). Middle and right panels are a close-up of the left one. Solid curves (`Normal`) show the correct result, while dashed curves (`Error`) show the result including errors.

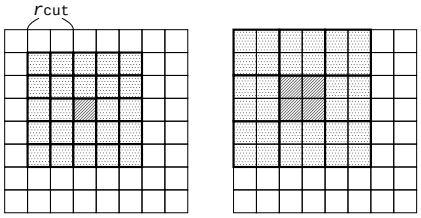


Figure 8. Communication region in the cell-index method. The left panel shows the communication without DS-CUDA, while the right panel shows the communication with DS-CUDA. The thick hatched region is the part a node has to manage, and thin hatched region is the region that it has to communicate among surrounding nodes. The granularity of communication and calculation becomes larger when DS-CUDA is used. Therefore, both the communication and calculation would be accelerated.

steps). Solid curves show the correct result, while dashed curves show the result including artificial errors. The error is artificially generated with a special DS-CUDA server. It randomly injects a bit error every 6 Mbyte in the data transferred from the server to the client. Using this technique we are able to emulate a faulty GPU. As shown in the right panel in Figure 7, the error causes different behavior of the temperature. Note that a single bit error may cause different results after a long simulation. If a bit error is critical, the simulation may stop immediately.

In our prototype system, we constructed a 2-way redundant device, that consists of a normal DS-CUDA server and an error-prone one. When we performed the simulation using our redundant device, we were able to obtain the correct results. The point is that the application program is not changed at all, and reliable calculation with redundant operation is achieved with the DS-CUDA system automatically.

IV. Conclusion and Future Work

We proposed DS-CUDA, a middleware to virtualize GPU clusters as a distributed shared GPU system. It simplifies the development of code on multiple GPUs on a distributed memory system.

Performances of two applications were measured on a 64-GPU system, where good scalability is confirmed. Also, the usefulness of the redundant calculation mechanism is shown, which distinguishes this work from other related work.

In the following part, we will discuss some issues under investigation.

A. Hybrid Programs with DS-CUDA

In some applications, time spent on the CPU cores, or communication between the cluster nodes and GPUs, can be the bottleneck of the overall calculation time. In such cases, DS-CUDA alone cannot offer much improvement in speed. However, combining with MPI parallelization, it may offer better performance than what can be achieved by MPI only. In the following, cell-index method [19] is discussed as example of such an application.

The cell-index method is a method to reduce the calculation cost of many-body interactions, such as gravity, Coulomb, and van der Waals forces. In this method, a cutoff length of interaction between two particles, r_{cut} is defined, and particles do not interact beyond this length. When interactions are calculated on a cluster of nodes, spatial domain decomposition is used and communication is needed only among neighboring nodes.

Consider a simulation space that is composed of $8 \times 8 \times 8 = 512$ cells and the cutoff length r_{cut} is double the cell size. The simulation is performed using a cluster of 512 nodes, each equipped with one GPU, and one MPI process is running on each cluster node.

A node is responsible for calculation of forces on the particles inside one cell. Each node requires data from the surrounding 124 (=$5^3 - 1$) cells to perform the calculation (recall that r_{cut} is double of the cell size). The left panel of Figure 8 illustrates a two dimensional slice of the simulation space. In order to calculate the forces on particles inside the thick-hatched cell, data must be transferred from the thin-

 ISBN: 978-1-61208-217-2

hatched cells.

For example, if 8 nodes are handled by one DS-CUDA node, the corresponding 8 cells, instead of a cell, are taken care by one MPI process with 8 GPUs. In this case, the number of MPI processes in total is reduced to 64, and each has to communicate with 26 ($= 3^3 - 1$) nodes, as shown in the right panel of Figure 8.

By using DS-CUDA, the performance and the programmability are improved in the following sense: (1) the number of MPI process is reduced to 1/8; (2) the total amount of communication among cluster nodes are reduced. In the right panel of Figure 8, the thick-hatched 8 cells communicate with 208 ($= 6^3 - 2^3$) cells. These 8 cells need to communicate with 992 ($= 8 \times (5^3 - 1)$) cells, if DS-CUDA is not used; (3) load imbalance among MPI processes becomes smaller, since the number of particles handled by one node increases 8 times on average.

B. DS-CUDA as a Cloud

A GPU cluster virtualized using DS-CUDA can be seen as a cloud, that offers flexibility, power saving, and reliability. The flexibility means that an arbitrary amount of GPU resource can be derived on request. The power saving means that some part of the cluster nodes can be suspended while there are no running jobs on the GPU. The reliability means that calculation errors can be recovered by the automated redundancy mechanism described in Section II-E.

If we could implement a function to dynamically migrate GPU resources between different nodes, and combining it with the redundant mechanism, a fault-tolerant system can be constructed. For example, if an unrecoverable error occurred on a server node, the malfunctioning node could automatically be retired and a new one could be joined, without stopping the simulation.

ACKNOWLEDGMENT

This research was supported in part by the Japan Science and Technology Agency (JST) Core Research of Evolutional Science and Technology (CREST) research programs "Highly Productive, High Performance Application Frameworks for Post Petascale Computing", and in part by Venture Business Promotion program of University of Electro-Communications. Authors thank Dr. Rio Yokota for his support on refining the manuscript.

REFERENCES

[1] M. Yokokawa, F. Shoji, A. Uno, M. Kurokawa, and T. Watanabe, "The K Computer: Japanese Next-Generation Supercomputer Development Project " in *International Symposium on Low Power Electronics and Design (ISLPED) 2011*, 2011, pp. 371–372.

[2] TOP500 Supercomputer Sites. Available: http://www.top500.org/ [retrieved: June, 2012]

[3] TSUBAME 2.0 Supercomputer. Available: http://www.gsic.titech.ac.jp/en/tsubame2 [retrieved: June, 2012]

[4] The NVIDIA website. Available: http://www.nvidia.com/object/cuda_home_new.html [retrieved: June, 2012]

[5] The KHRONOS website. Available: http://www.khronos.org/opencl/ [retrieved: June, 2012]

[6] T. Hamada, R. Yokota, K. Nitadori, T. Narumi, K. Yasuoka, M. Taiji, and K. Oguri, "42 TFlops Hierarchical N-body Simulations on GPUs with Applications in both Astrophysics and Turbulence" in *International Conference for High Performance Computing, Networking, Storage and Analysis (SC) 2009*, 2009, USB memory.

[7] Cray Inc. The Chapel Parallel Programming Language. Available: http://chapel.cray.com/ [retrieved: June, 2012]

[8] P. Charles, C. Grothoff, V. Saraswat, C. Donawa, A. Kielstra, K. Ebcioglu, C. von Praun, and V. Sarkar, "X10: an Object-Oriented Approach to Non-Uniform Cluster Computing" in *20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, OOPSLA '05*, 2005, pp. 519–538.

[9] J. Duato, A. J. Pẽna, F. Silla, R. Mayo, and E. S. Quintana-Ortí, "Performance of CUDA Virtualized Remote GPUs in High Performance Clusters" in *2011 IEEE International Conference on Parallel Processing*, 2011, pp. 365–374.

[10] R. M. Gual, "rCUDA:an approach to provide remote access to GPU computational power" in *HPC Advisory Council European Workshop 2011*, Available: http://www.hpcadvisorycouncil.com/events/2011/european_workshop/pdf/3_jaume.pdf [retrieved: June, 2012]

[11] L. Shi, H. Chen, J. Sun, and K. Li, "vCUDA: GPU-Accelerated High-Performance Computing in Virtual Machines" *IEEE Transactions on Computers*, vol. PP, 2011, pp. 1–13.

[12] G. Giunta, R. Montella, G. Agrillo, and G. Coviello, "A GPGPU transparent virtualization component for high performance computing clouds", in *Euro-Par 2010 - Parallel Processing, ser. LNCS*, vol. 6271, 2010, pp. 379–391.

[13] V. Gupta, A. Gavrilovska, K. Schwan, H. Kharche, N. Tolia, V. Talwar, and P. Ranganathan, "GViM: GPU-accelerated virtual machines", in *3rd Workshop on System-level Virtualization for High Performance Computing*, 2009, pp. 17–24.

[14] A. Barak, T. Ben-Nun, E. Levy, and A. Shiloh, "A package for OpenCL based heterogeneous computing on clusters with many GPU devices" in *Workshop on Parallel Programming and Applications on Accelerator Clusters*, 2010.

[15] The NVIDIA website. Available: http://developer.nvidia.com/gpudirect [retrieved: June, 2012]

[16] J. Barnes and P. Hut, "A hierarchical $O(N \log N)$ force-calculation algorithm" *Nature*, vol. 324, 1986, pp.446–449.

[17] L. Greengard and V. Rokhlin, "A Fast Algorithm for Particle Simulations" *J. Comput. Phys.*, vol 73, 1987, pp.325–348.

[18] M. P. Tosi, and F. G. Fumi, "Ionic sizes and born repulsive parameters in the NaCl-type alkali halides–II: The generalized Huggins-Mayer form" *J. Phys. Chem. Solids*, vol. 25, 1964, pp 45–52.

[19] R. W. Hockney, J. W. Eastwood, Computer Simulation Using Particles. New York, McGraw-Hill, 1981.

 ISBN: 978-1-61208-217-2

Parallel Approaches for Mining Fuzzy Orderings based Gradual Patterns

Malaquias Quintero Flores*, Federico Del Razo†, Anne Laurent*, Nicolas Sicard‡
* *University Montpellier 2 CNRS - LIRMM, Montpellier, France*
Email: {quinterofl, laurent}@lirmm.fr
† *Toluca Institute of Technology, Toluca, Mexico*
Email: delrazo@ittoluca.edu.mx
‡ *EFREI - AllianSTIC, Paris, France*
Email: nicolas.sicard@efrei.fr

***Abstract*—Mining gradual patterns invokes a number of iterations for generating, adjusting, measuring, and comparing gradual tendencies between numeric attributes of imprecise or uncertain databases. Gradual tendencies are complex correlations of the form {*The hight/lower X, the hight/lower Y*}. Automatic extraction of such gradual patterns involves huge amounts of processing time, load balance, and high memory consumption. When managing large databases, taking this into account is challenging. In this paper, we show a framework and an algorithm based on rank correlation and fuzzy orderings for mining gradual patterns from imprecise or uncertain data. We also present an approach to improve performance of the algorithm using the parallel programming model of OpenMP and the Yale Sparse Matrix Format to reduce memory consumption. Through an experimental study, we show the performance of our approach with respect to the number of attributes of the databases and the number of cores available.**

***Keywords*-Scaling fuzzy models; *Fuzzy orderings*; *Fuzzy gradual dependencies*; *OpenMP*; *Parallel programming*.**

I. INTRODUCTION

Data mining is often defined as the formulation, analysis, and implementation of an induction process proceeding from specific data to general patterns that facilitates the non-trivial extraction of implicit, unknown, and potentially useful information [20]. Data mining techniques are employed in traditional scientific discovery disciplines, such as biological, medical, biomedical, chemical, physical, social research, and other knowledge industries.

Mining gradual patterns (MGP) is an important task for knowledge discovery (KDD) and data mining (DM). In MGP, the goal is to find in numeric databases interesting and complex correlations of the form $\{X\{\geq|\leq\}, Y\{\geq|\leq\}\}$, interpreted as {*The hight/lower X, the hight/lower Y*} and named *gradual dependencies*.

Gradual dependencies express a relation (*tendency* or *correlation*) among the variation of the values of attributes of a *gradual pattern*. For example, in the database shown in Table I containing data about fruit characteristics, such as A_1*:Size*, A_2*:Weight* and A_3*:Sugar Rate*, the gradual pattern {*the higher the weight*, *the hight the sugar rate*}, means that as the *weight* of a fruit increases, its *sugar rate* tends to increases. Accordingly {*the lower the weight*, *the lower the sugar rate*}, means that as the *weight* of a fruit decreases, its *sugar rate* tends to decreases.

Table I
A SMALL DATABASE OF SOME FRUIT CHARACTERISTICS

Id	$A_1 : Size$	$A_2 : Weight$	$A_3 : SugarRate$
t_0	6	6	5.3
t_1	10	12	5.1
t_2	14	4	4.9
t_3	23	10	4.9
t_4	6	8	5.0
t_5	14	9	4.9
t_6	18	9	5.2
t_7	23	10	5.3
t_8	28	13	5.5

This new way to analyse the degree in which a pattern is present in a transaction, called *gradual pattern* or *gradual dependency*, was first proposed in [10], thereafter studied in [1][13][16], and more recently in [5][12][18].

There exist some algorithms for MGP that focus on finding gradual patterns from precise and certain data [12][22]. Unfortunately, this is not always so, there are situations where databases contain imprecise and uncertain data (e.g., meteorological data, river pollution data, indicator economic data, biological data, and so on) due to human errors, instrument errors, recording errors, noisy data, variables with imprecise and uncertain behaviour, and so on [2][6][22].

In this paper, we present a framework and an algorithm based on rank correlation and fuzzy orderings for MGP (*fuzzy*-MGP) [18] from imprecise or uncertain data. We also present an approach to improve performance of *fuzzy*-MGP algorithm using the parallel programming model of OpenMP [19][21] and a dedicated technique for handling sparse matrices to reduce memory consumption.

The outline of the paper is as follows: In Section II, we present the definitions of gradual dependency, fuzzy orderings, fuzzy γ rank correlation and related works. In

 ISBN: 978-1-61208-217-2

Section III, we explain our parallel fuzzy orderings for fuzzy gradual pattern mining algorithm (*parallel fuzzy*-MGP). Section IV presents experiments and main results. Lastly, in Section V, we present our conclusions and perspectives of research.

II. Definitions and Related Works

A. Gradual dependency: Some Background

In the context of fuzzy data mining, the concept of *gradual dependency* was first proposed to perform a regression analysis between the change of the presence of fuzzy items in a transaction [10].

In [1], a gradual dependence is defined as a rule of the form $(v_1, X, A) \rightarrow (v_2, Y, B)$ holds in D *iff* $\forall (x, y), (x^{'}, y^{'}) \in D$, meaning that $A(x)v_1A(x^{'})$ implies $B(y)v_2B(y^{'})$, where X, Y are two attributes of database D containing pairs of values $(x, y) \in X \times Y$, two fuzzy sets A, B defined on the domains of X, Y respectively, and $v_1, v_2 \in \{<, >\}$, represent two variations *the less* ($<$) and *the more* ($>$).

In [16], Molina et al. propose a particular definition of fuzzy gradual dependence from an specific approach to fuzzy association rules, where gradual dependencies represent tendencies in the variation of the degree of fulfilment of properties in a set of objects, and also introduce the notion of degree of variation associated to a pair of objects.

In [13], Laurent et al. present an approach for extracting gradual itemsets, where combines the interpretation of gradual dependency of Berzal in [1], with a rank correlation measure and the concept of binary matrices for representing the sets of concordant couples.

Koh and Hüllermeier [12] present a framework for mining fuzzy gradual dependencies, in which the strength of association between itemsets is measured in terms of a fuzzy rank correlation coefficient.

B. Gradual Pattern: Concepts and Definitions

Given a DB (database), constituted of *n* data record (transactions) **T**=$\{t_1, t_2, ..., t_n\}$, described by *m* numeric attributes **A**=$\{A_1, A_2, ..., A_m\}$, each record t_i is represented as a vector with *m* values, $t_i = \{A_1(t_i), A_2(t_i), ..., A_m(t_i)\}$, were $A_2(t_i)$ is the value of t_i for the attribute A_2, similarly for each record t_j.

A *GP*(*gradual pattern*) defines a relation of simultaneous variation between values of the attributes of two or more *gradual items*, different approaches have been defined according to the interpretation of the concept of gradual dependency, a *GP* is a combination of two or more *gradual items* of the form *GP*=$\{A_1 \geq A_2 \geq A_3 \leq\}$ interpreted as {*The hight* A_1, *the hight* A_2, *the lower* A_3}, where the size (*k*) of a *GP* is defined as the number of *gradual items* contained in the *GP*, such that *k*$\in\{2, 3, 4, ..., m\}$.

A *gradual item* is defined as the variation **v** associated to the values of a attribute $A_l \in DB$ denoted as A_l**v**, where **v** can be ascending ($\geq$|+) if the attribute values increase, descending ($\leq$|$-$) if the attribute values decrease, i.e., $\{A_l \geq\} \simeq \{A_l(t_i) < A_l(t_j)\}$ and $\{A_l \leq\} \simeq \{A_l(t_i) > A_l(t_j)\}$ for *i=1, 2, ..., n*, for *j=i+1, ..., n*, $i \neq j$ and $l \in \{1, 2, ..., k\}$.

A *concordant couple (cc)* is an index pair *cc(Ii,Ij)* defined in (1), where the records (t_i, t_j) satisfy all the variations **v** expressed by the involved *gradual items* in a given *GP* of size *k*, e.g., let *GP*=$\{A_1 \geq A_2 \geq A_3 \leq\}$ with size *k=3*, an index pair *cc(Ii,Ij)* is a *concordant couple* if $((A_1(t_i) < A_1(t_j)\ implies\ A_2(t_i) < A_2(t_j))\ implies\ A_3(t_i) > A_3(t_j))$, where *Ii* is defined in (2) and *Ij* in (3).

$$cc(Ii, Ij) = \begin{cases} 1 & \text{if } cc(Ii, Ij)\ is\ concordant\ couple \\ 0 & \text{in otherwise.} \end{cases} \tag{1}$$

$$\begin{cases} Ii = ((A_1(t_i), A_2(t_i)), (A_1(t_i), A_3(t_i)))\ or \\ \quad = ((A_1(t_i), A_2(t_i)), A_3(t_i)) \end{cases} \tag{2}$$

$$\begin{cases} Ij = ((A_1(t_j), A_2(t_j)), (A_1(t_j), A_3(t_j)))\ or \\ \quad = ((A_1(t_j), A_2(t_j)), A_3(t_j)) \end{cases} \tag{3}$$

In *MGP* a *minimal support* is used to choose interesting *GP* from *gradual items* which frequently occur together. A *GP* is an interesting pattern if *support(GP)* is greater than or equal to the user-predefined minimal support named *minimum threshold*.

In order to compute the support of a GP, different approaches have been defined according to the interpretation of the concept of gradual dependency, such as based on *fuzzy implication interpretation* [4][7][9], *regression analysis* [10], *induced rankings correlation* [1][13], *ranking-compliant data subsets* [15], *strengthening quality criteria* [5], and so on. An interesting analysis of such interpretations is considered in [5][13].

In the framework of the interpretation of gradual dependency based on *induced rankings correlation* and *concordant couple* concept, the *support* of a *GP* is computed as

$$support(GP) = \frac{\sum_{i=1}^{n} \sum_{j \neq i} cc\,(Ii, Ij)}{\frac{n(n-1)}{2}} \tag{4}$$

C. Fuzzy Orderings-Based $\tilde{\gamma}$ Rank Correlation Measure

Fuzzy orderings is a concept that have been introduced with the aim to model human-like decisions by taking the graduality of human thinking and reasoning into account. Within the framework of a process of making decisions *fuzzy orderings* allow express and evaluate preferences among a set of available alternatives [2][24].

A fuzzy relation $L : X^2 \rightarrow$[0,1] is called *fuzzy ordering* with respect to a t-norm *T* and a *T*-equivalence $E : X^2 \rightarrow$[0,1], for brevity *T-E-ordering*, if and only if the following three axioms are fulfilled for all *x, y, z* $\in$ *X*:

(i) *E*-Reflexivity: $E(x,y) \leq L(x,y)$
(ii) *T-E*-Antisymmetry: $T(L(x,y),L(y,x)) \leq E(x,y)$
(iii) *T*-Transitivity: $T(L(x,y),L(y,z)) \leq L(x,z)$.

The result of combining fuzzy orderings and *gamma rank correlation measure* is a robust rank correlation coefficient ideally suited for measuring rank correlation for numerical data perturbed by noise [3]. This innovative correlation coefficient is known as *fuzzy ordering-based rank correlation measure* $\tilde{\gamma}$. Its formal definition is:

$$\tilde{\gamma} = \frac{CT - DT}{CT + DT} \tag{5}$$

$$CT = \sum_{i=1}^{n} \sum_{j \neq i} \tilde{C}(i,j) \tag{6}$$

$$DT = \sum_{i=1}^{n} \sum_{j \neq i} \tilde{D}(i,j) \tag{7}$$

$$\tilde{C}(i,j) = \top(R_X(x_i,x_j), R_Y(y_i,y_j)) \tag{8}$$

$$\tilde{D}(i,j) = \top(R_X(x_i,x_j), R_Y(y_j,y_i)) \tag{9}$$

$$R_X(x_i,x_j) = 1 - L_X(x_j,x_i) \tag{10}$$

$$L_X(x1,x2) = min(1, max(0, 1 - \frac{(x1-x2)}{r})) \tag{11}$$

$$R_Y(y_i,y_j) = 1 - L_Y(y_j,y_i) \tag{12}$$

$$L_Y(y1,y2) = min(1, max(0, 1 - \frac{(y1-y2)}{r})) \tag{13}$$

$$\top(a,b) = max(1, a+b-1) \tag{14}$$

$$\{(x_i,y_i)_{i=1}^{n} | x_i \in X \text{ and } y_i \in Y\} \tag{15}$$

where $\tilde{C}$*(i, j)* is the degree to which *(i, j)* is a concordant pair and $\tilde{D}$*(i, j)* is the degree to which *(i, j)* is a discordant pair, for *i*=(x_i, y_i) and *j*=(x_j, y_j), such that $_{i=1,\ 2,\ldots,n}$, $_{j=1,\ 2,\ldots,n,\ i \neq j}$ and $n \neq CT + DT$. Given $n \geq 2$ pairs of numeric observations, defined in (15)

$R_X(x_i,x_j)$ is a strict $T_L - E_X - ordering$ on *X* defined in (10), $R_Y(y_i,y_j)$ is a strict $T_L - E_Y - ordering$ on *Y* defined in (12), $L_X(x_i,x_j)$ is a strongly complete $T_L - E_r - ordering$ on *X* defined in (11), and $L_Y(y_i,y_j)$ is a strongly complete $T_L - E_r - ordering$ on *Y* defined in (13) (assume *r>0*).

$\top$ is a $\top$*−equivalence relation* and denotes a Lukasiewicz *t-norm* defined in (14) for *a*=$R_X(x_i,x_j)$ and $b = R_Y(y_i,y_j)$. For more information we recommend consulting [2][3][12].

D. Problem statement

An important challenge in gradual pattern mining is to analyze the correlation between the variation of numerical attribute values perturbed by noise, and to consider when such a small difference between two values is meaningful. In this context, we present a fuzzy orderings-based framework for mining fuzzy gradual patterns, where we propose to compute the support of a *GP*, as in (16) based on compute of the degree to which *(Ii, Ij)* is a concordant pair as is defined in (8) and according to the definition of *concordant couple* given in (1), (2), and (3).

$$fuzzsupport(GP) = \frac{\sum_{i=1}^{n} \sum_{j \neq i} \tilde{C}(Ii,Ij)}{n(n-1)} \tag{16}$$

Huge amounts of processing time, load balance and high memory consumption are important problems observed in gradual pattern mining algorithms. We addressed these problems making use of a parallel programming model and an efficient technique to reduce memory consumption.

E. Related Work

Recently, in [14] and [15], Laurent et al. have presented PGP-mc a multicore parallel approach for mining gradual patterns where the evaluation of the correlation and support is based on conflict sets and precedence graph approaches. PGP-mc was implemented using the g++ 3.4.6 and 4.3.2 with POSIX threads, on two different workstations: i) COYOTE machine, with 8 AMD Opteron 852 processors (each with 4 cores), 64 GB of RAM with Linux Centos 5.1 And ii) IDKONN machine, with 4 Intel Xeon 7460 processors (each with 6 cores), 64 GB of RAM with Linux Debian 5.0.2. The experiments were led on synthetic databases automatically generated by a tool based on adapted version of IBM Synthetic Data Generation Code for Associations and Sequential Patterns.

 ISBN: 978-1-61208-217-2

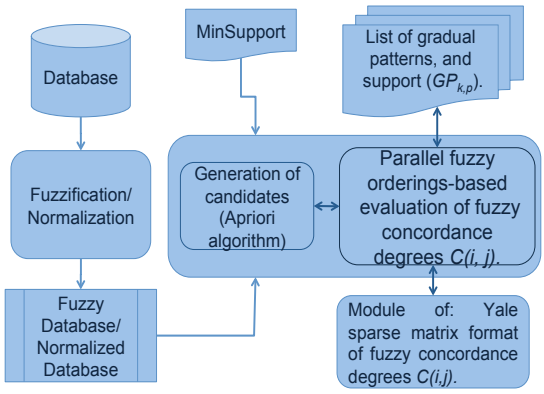


Figure 1. Parallel fuzzy orderings-based extraction of fuzzy frequent gradual dependencies: General structure.

An efficient parallel mining of closed frequent gradual patterns, named PGLCM, has been proposed by Do et al. [8]. This approach is based on the principle of the LCM algorithm for mining closed frequent patterns, an adaptation of LCM named GLCM in order to mine closed frequent gradual patterns, and parallelization of the GLCM algorithm named PGLCM based on the Melinda parallelism environment. The comparative experiment is based on synthetic databases produced with the same modified version of IBM Synthetic Data Generator for Association and Sequential Patterns. All the experiment have been conduced on a 4-socket server with 4 Intel Xeon 7460 with 6 cores each and 64 GB of RAM.

III. Parallel Fuzzy Orderings for Fuzzy Gradual Pattern Mining

In this section, we present an approach to improve the performance of our method of automatic extraction of frequent gradual patterns on the basis of fuzzy orderings [18] and the idea of storing the fuzzy concordance degrees $C(i,j)$ in sparse matrices.

Figure 1 shows the general structure of our optimization approach which has two purposes: A) Reduce memory consumption and B) Improve execution time via parallelization.

A. Memory consumption

In order to reduce memory consumption, we represented and stored each matrix of concordance degrees $C(i,j)$ according to the *Yale Sparse Matrix Format* [23], such as only non-zero coefficients are retained. Because we generate candidates from the frequent $k-$patterns, only matrices of the $(k\text{-}1)-$level frequent gradual patterns are kept in memory while being used to generate the matrices of the $(k)-$level gradual pattern candidates, e.g., Figure 2 illustrates the extraction of gradual patterns, from the data set of Table I, with a *minimum threshold*= 0.15, for level $k=2$ and $k=3$, where, if *support* of a *gradual pattern* (*GP*) is less than *minimum threshold* then the *GP* is pruned and its matrix of

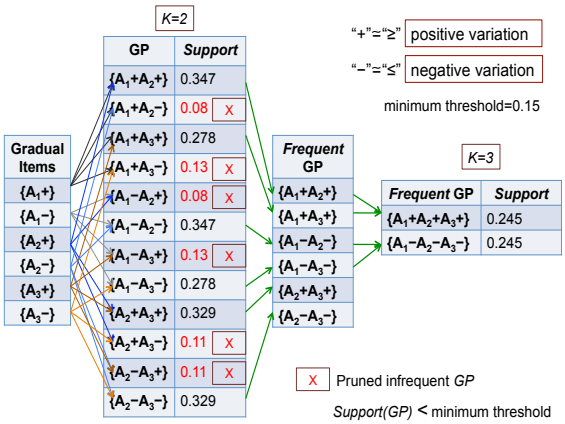


Figure 2. Extraction of gradual patterns from data set of Table I for level $k=2$ and $k=3$.

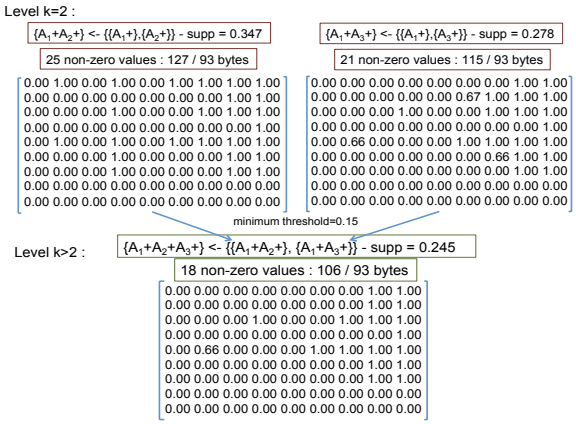


Figure 3. Examples of matrices of fuzzy concordance degrees $C(i,j)$ of three gradual patters of Figure 2 with ascending variation.

Compressed matrix of fuzzy concordance degrees *C(i,j)*:

Gradual pattern $GP_{3,1}$= {A1+A2+A3+} ⟵ {{A1+A2+},{A1+A3+}}

18 non-zero values : 106 / 93 bytes

	0	1	2	3	4	5	6	7	8
IA	0	2	4	8	9	14	16	null	null

	0	1	2	3	4	5	6	7	8	9	10
JA	7	8	7	8	3	6	7	8	8	1	5
A	1.0	1.0	1.0	1.0	1.0	1.0	0.72	1.0	1.0	0.66	1.0

	11	12	13	14	15	16	17
JA	6	7	8	7	8	7	8
A	1.0	1.0	1.0	1.0	1.0	1.0	1.0

Figure 4. Example of representation of a matrix of fuzzy concordance degrees *C(i,j)* in the *Yale Sparse Matrix Format*.

 ISBN: 978-1-61208-217-2

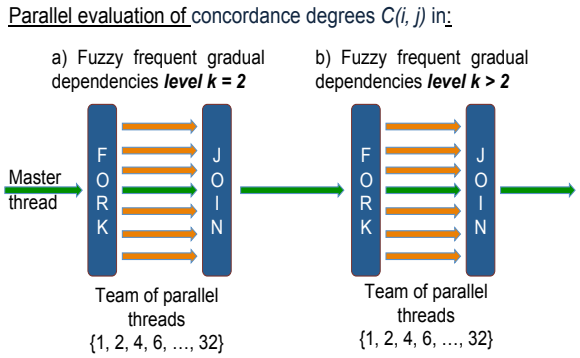


Figure 5. Parallel evaluation of fuzzy concordance degrees in: a) Gradual dependency of level $k = 2$, b) Gradual dependency of level $k > 2$.

fuzzy concordance degrees *C(i,j)* is removed.

Figure 3 shows the matrices of fuzzy concordance degrees *C(i,j)* of the gradual patters $GP_{2,1}$=$\{A_1+A_2+\}$, $GP_{2,3}$=$\{A_1+A_3+\}$ of level *k=2*, and $GP_{3,1}$=$\{A_1+A_2+A_3+\}$ of level *k=3*. Notation "+" represents an ascending variation "$\geq$" and notation "$-$" represents an descending variation "$\leq$". Figure 4 shows the representation of the matrix of fuzzy concordance degrees *C(i,j)* in the *Yale Sparse Matrix Format* for the gradual patter $GP_{3,1}$.

B. Parallelization of algorithm

Load balance, huge amounts of processing time and high memory consumption are important problems observed in gradual pattern mining algorithms. We addressed these problems using the shared memory architecture API of OpenMP, which is ideally suited for multi-core architectures [19][21].

Figure 5 gives an overall view of our approach to parallelize the automatic extraction of frequent gradual patterns of level *k=2* and the automatic extraction of frequent gradual patterns of level *k>2*.

Algorithm 1 shows the pseudocode of the master thread, where DB is a database, ***m*** is the number of attributes in DB, ***n*** is the number of records in DB, $\{A_{\mathbf{m}}\}$ represents the identifiers of the ***m*** attributes.

The variable named $minSupp$ is the *minimum threshold*, ***v***$\leftarrow\{+\}$ denotes a positive (ascending) variation in each index pair of an attribute, ***v***$\leftarrow\{-\}$ denotes a negative (descending) variation in each index pair of an attribute, *ListGIs* represents the list of *gradual items* level *k=1*, $\mathcal{F}_{k=2}$ is the set of gradual patterns/dependencies of level *k=2*, $\mathcal{F}_k$ is the set of gradual patterns/dependencies of level *k>2*, $\mathcal{F}_{FGD}$ are all fuzzy frequent gradual dependencies and *Nt* is the number of threads of each parallel region.

Algorithm 1: Main Thread for Fuzzy Gradual Dependency Mining

Data: Database(DB), # Attributes(**m**), # Records(**n**),
Data: Id. Attributes$\{A_{\mathbf{m}}\}$, $minSupp$, # Threads (Nt).
Result: Fuzzy Frequent Gradual Dependencies $\mathcal{F}_{FGD}$
$\mathcal{F}_{FGD} \leftarrow \emptyset$; $\mathbf{v} \leftarrow \{+,-\}$;
$ListGIs$ = $build\_gItems(\{A_{\mathbf{m}}\} \times \mathbf{v})$;
$k \leftarrow 2$;
/* Parallel extraction of fuzzy frequent gradual dependencies level $k = 2$*/;
$\mathcal{F}_{k=2}$ = $GradualPatterns(ListGIs, Ds, minSupp)$;
$\mathcal{F}_{FGD} \leftarrow \mathcal{F}_{FGD} \cup \{\mathcal{F}_{k=2}\}$;
$k++$;
repeat
  /* Parallel extraction of fuzzy frequent gradual dependencies level $k > 2$*/;
  $\mathcal{F}_k$=$GradualPatterns(\mathcal{F}_{k-1}, level(k), minSupp)$;
  $\mathcal{F}_{FGP} \leftarrow \mathcal{F}_{FGP} \cup \{\mathcal{F}_k\}$;
  $Delet(\mathcal{F}_{k-1}.M)$;
  k++;
until $\mathcal{F}_{FGP}$ *does not grow any more*;

The pseudo-code of the Algorithm 2 corresponds to the parallel extraction of frequent gradual dependencies of level $k = 2$, where $Sdc_{k=2}$ is the set of gradual patterns of size $k = 2$, Gdc is a pattern gradual candidate $\in Sdc_{k=2}$, $Gdc_q.M$ represents the matrix of fuzzy concordance degrees $C(i,j)$ *computed by fuzzy orderings*.

The pseudo-code of the Algorithm 3 corresponds to the parallel extraction of frequent gradual dependencies of level $k > 2$, where $\{\mathcal{F}_{k^1}\}$ is the set of frequent gradual patterns of level $(k-1)$, $C_{k,q}.M$ represents the matrix of fuzzy concordance degrees $C(i,j)$ of a gradual pattern candidate of level k (*computed* by $T-norm$) and $\mathcal{F}_k$ is the set of frequent gradual patterns of level k.

IV. Experiments and Main Results

A. Experiments

We present an experimental study of the scaling capacities of our approach on several cores, for the database C500A100 with 500 records and 100 attributes, and database C500A150 with 500 records and 150 attributes, which were used in [8][14] and produced with the IBM Synthetic Data Generator for Association and Sequential Patterns.
Our experiments were performed on a workstation with up to 32 processing cores, named COYOTE, with 8 AMD Opteron 852 processors (each with 4 cores), 64 GB of RAM with Linux Centos 5.1, GCC OpenMP 3.1.

 ISBN: 978-1-61208-217-2

Algorithm 2: Parallel Fuzzy Orderings for Gradual Pattern Mining: Level $k = 2$

Data: List_of_gradual_Items($List\_gIs$), minSupp
Data: Attribute values (aValues), # Threads (Nt)
Result: Fuzzy Frequent Gradual Dependencies $(\mathcal{F}_{k=2})$
$\mathcal{F}_{k=2} \leftarrow \emptyset$; $q \leftarrow 1$;
/*Each thread computes the ***support*** of its fuzzy frequent gradual dependency of level $k = 2$*/;
for ***all***$(thread\ in(1, 2, 4, 6, 8, ..., Nt = 32))$ **do**
  $Sdc_{k=2} \leftarrow GenCand(\{List\_gIs\}, level(k = 2))$;
  $Gdc \leftarrow FirstxCandidate \in Sdc_{k=2}$;
  foreach $Gdc \in Sdc_{k=2}$ **do**
    $Gdc_q.M = FuzzOrderings(Gdc, aValues)$;
    $Support(Gdc_q) = EvalSupport(Gdc_q.M)$;
    /* *minSupp stands for a user-specified minimum support value* */;
    if $Support(Gdc_q) \geq minSupp$ **then**
      $Critical\ section:$
      $>>>$ $\mathcal{F}_{k=2} \leftarrow \mathcal{F}_{k=2} \cup \{Gdc_q\}$;
    $q++$;
    $Gdc_q \leftarrow NextCandidate \in Sdc_{k=2}$;

Algorithm 3: Parallel Fuzzy Orderings for Gradual Pattern Mining: Level $k > 2$

Data: Fuzzy Frequent Gradual Dependency$\mathcal{F}_{k-1}$,
Data: minSupp, Level*(k>2)*, # Threads (Nt).
Result: Fuzzy Frequent Gradual Dependencies $(\mathcal{F}_k)$
$\mathcal{F}_k \leftarrow \emptyset$; $q \leftarrow 1$; $k^1 \leftarrow k - 1$;
/*Each thread computes the ***support*** of its fuzzy frequent gradual dependency of level $k > 2$*/;
for ***all***$(thread\ in(1, 2, 4, 6, 8, ..., Nt = 32))$ **do**
  $F_{k^1,a} \leftarrow GenFirstFather1(\{\mathcal{F}_{k^1}\})$;
  $F_{k^1,b} \leftarrow GenFirstFather2(\{\mathcal{F}_{k^1}\})$;
  foreach $\{F_{k^1,a}, F_{k^1,b}\} \in \{\mathcal{F}_{k^1}\}$ **do**
    $C_{k,q}.M = T-norm(\{F_{k^1,a}.M\}, \{F_{k^1,b}.M\})$;
    $Support(C_{k,q}.M) = EvalSupport(C_{k,q}.M)$;
    /* *minSupp stands for a user-specified minimum support value* */;
    if $Support(C_{k,q}.M) \geq minSupp$ **then**
      $Critical\ section:$
      $>>>$ $\mathcal{F}_k \leftarrow \mathcal{F}_k \cup \{C_{k,q}\}$;
    $F_{k^1,a} \leftarrow GenNextFather1(\{\mathcal{F}_{k^1}\})$;
    $F_{k^1,b} \leftarrow GenNextFather2(\{\mathcal{F}_{k^1}\})$;
    $q++$;

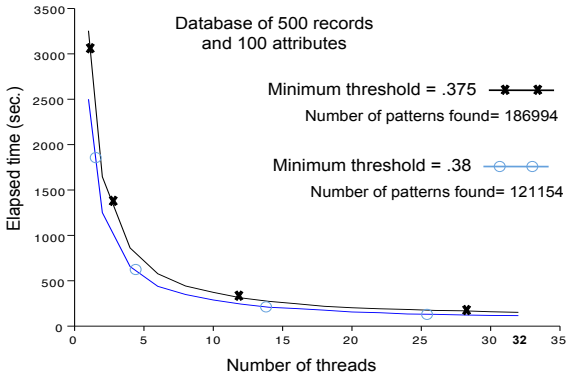


Figure 6. Threads vs. elapsed time with a database of 500x100 and minSupp=.375 and .38, using uncompressed binary matrices of concordance degrees.

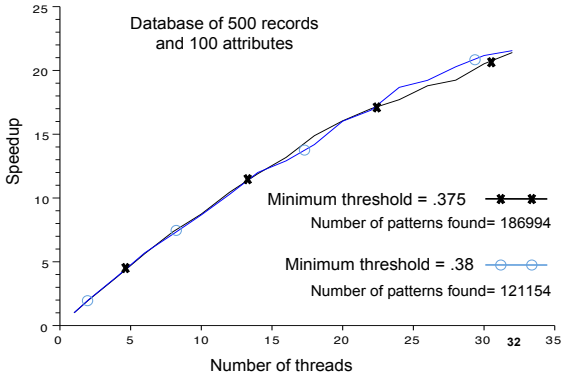


Figure 7. Speedup with a database of 500x100 and minSupp=.375 and .38, using uncompressed binary matrices of concordance degrees.

B. Main Results

The first experiment involves a database with 500 lines and 100 attributes, Figures 6 and 7 depict the execution time and speed-up related to: 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, and 32 threads, on the test database, from which were found 186994 frequent gradual patterns for a minimum threshold of 0.375, and 121154 frequent gradual patterns for a minimum threshold of 0.38. In the first case we reach a memory consumption of 36.2% using uncompressed binary matrices of concordance degrees and of 14.4% using compressed matrices of concordance degrees (Yale Sparse Matrix Format). While for the second case we obtained a memory consumption of 24.7% with uncompressed binary matrices of concordance degrees and of 10.3% with compressed matrices of concordance degrees. Within this experimental framework, Figures 8 and 9 illustrate the execution time and speed-up of our approach using compressed matrices of concordance degrees.

The second experiment involves a database with 500 lines and 150 attributes, from which were found 100834 frequent

 ISBN: 978-1-61208-217-2

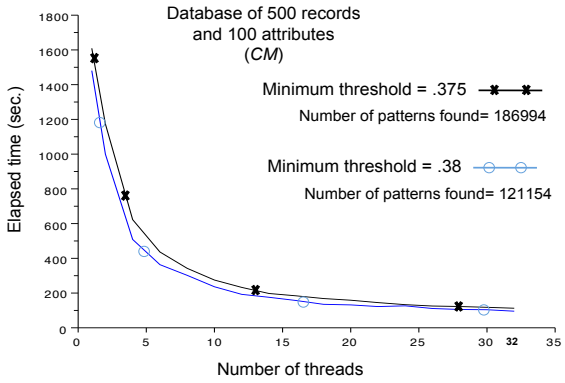


Figure 8. Threads vs. elapsed time with a database of 500x100 and minSupp=.375 and .38, using compressed matrices of concordance degrees.

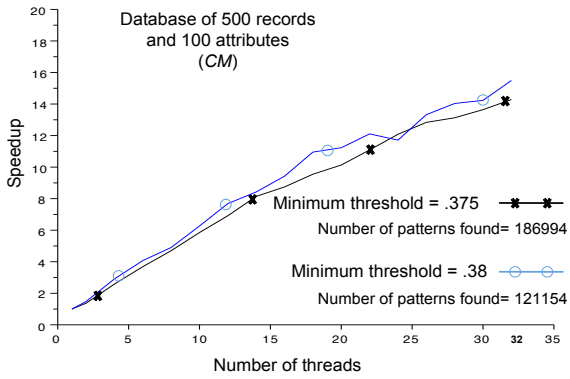


Figure 9. Speedup with a database of 500x100 and minSupp=.375 and .38, using compressed matrices of concordance degrees.

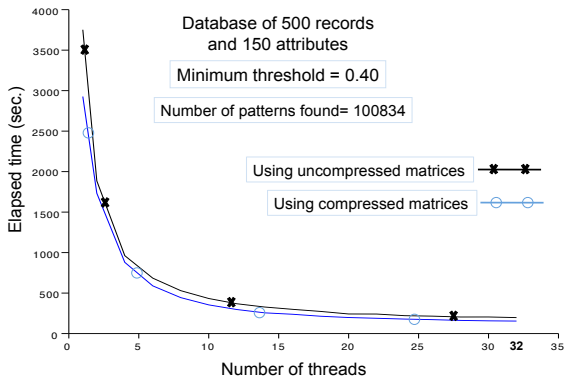


Figure 10. Threads vs. elapsed time with a database of 500x150 and minSupp=0.40, using uncompressed and compressed matrices of concordance degrees.

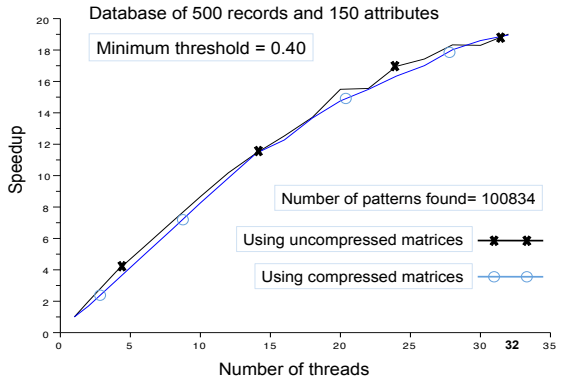


Figure 11. Speedup with a database of 500x150 and minSupp=0.40, using uncompressed and compressed matrices of concordance degrees.

gradual patterns for a minimum threshold of 0.40, where we reach a memory consumption of 24.5% using uncompressed binary matrices of concordance degrees and of 10.8% using compressed matrices of concordance degrees (Yale Sparse Matrix Format). Within this experimental framework, Figures 10 and 11 depict the execution time and speed-up of our approach, related to: 1, 2, 4, 6, . . . , to 32 threads.

V. Conclusion and Future Work

In this paper, we presented the parallelization of our fuzzy gradual pattern mining named *fuzzy orderings for fuzzy gradual pattern mining*. Our approach consists in parallelizing the extraction of gradual dependencies of level $k = 2$ and in parallelizing the extraction of gradual dependencies of level $k > 2$. The programming parallel model used was the shared memory with the API of OpenMP - C++.

In order to reduce memory consumption, each matrix of concordance degrees $C(i, j)$ is represented and stored according to the *Yale Sparse Matrix Format*, such as only non-zero coefficients are retained. Because we generate candidates from the frequent $k-$itemsets, only matrices of the $(k-1)-$level frequent gradual patterns are kept in memory while being used to generate the matrices of the $(k)-$level gradual pattern candidates.

The experimental work reported in this document was conducted on two databases, the first with 500 records and 100 attributes and the second with 500 records and 150 attributes.

In general, the performance of our *Parallel Fuzzy Gradual Dependency Mining* is significantly improved with the use of compressed matrices, mainly with databases containing a large number of attributes. We recommend testing our algorithm with databases with a larger number of records and real world databases.

References

[1] F. Berzal, J.C. Cubero, D. Sanchez, J. M. Serrano, and M. A. Vila, *An alternative approach to discover gradual dependencies,* Int. Journal of Uncertainty, Fuzziness and Knowledge-based Systems, vol. 15, no. 5, pp. 559-570, (2007).

 ISBN: 978-1-61208-217-2

[2] U. Bodenhofer and F. Klawonn, *Towards robust rank correlation measures for numerical observations on the basis of fuzzy orderings,* in 5th Conference of the European Society for Fuzzy Logic and Technology, pp. 321 - 327, Ostrava, Czech Republic, University of Ostrava, Institute for Research and Applications of Fuzzy Modeling, (Septembre, 2007).

[3] U. Bodenhofer and F. Klawonn, *Roboust rank correlation coefficients on the basis of fuzzy orderings: Initial steps,* in Mathware & Soft Computing 15, 5-20 (2008).

[4] B. Bouchon-Meunier and S. Desprès. *Acquisition numérique / sysmbolique de connaissance graduelles,* in 3èmes Journées Nationales du PRC Intelligence Artificielle. Hermès, pp. 127-138 (1990).

[5] B. Bouchon-Meunier, A. Laurent, M.-J. Lesot, and M. Rifqi, *Strengthening fuzzy gradual rules through "all the more" clauses,* in Proc. WCCI 2010 IEEE World Congress on Computatinal Inteligence, FUZZ-IEEE 2010, pp. 2940-2946 (2010).

[6] T. Calders, B. Goethals, and S. Jaroszewicz, *Mining Rank-Correlated Sets of Numerical Attributes,* in Proc. KDD'06 ACM, pp. 96-105, (2006).

[7] D. Dubois and H. Prade, *Gradual inference rules in approximate reasoning,* Information Sciences, vol. 61, no. 1-2, pp. 103-122 (1992).

[8] T. D. T. Do, A. Laurent, and A. Termier, *PGLCM: Efficient Parallel Mining of Closed Frequent Gradual Itemsets,* in Proc. International Conference on Data Mining (ICDM), pp. 138-147 (2010).

[9] E. Hüllermeier, *Implication-based fuzzy association rules,* in Proceedings PKDD'01, pp. 241-252, (2001).

[10] E. Hüllermeier, *Association rules for expressing gradual dependencies,* in Proceedings PKDD 2002 Lecture Notes in Computer Science 2431, pp. 200-211 (2002).

[11] E. Hüllermeier, *Fuzzy Sets in Machine Learning and Data Mining,* in Applied Soft Computing Journal, vol. 11, pp. 1493-1505, (2011).

[12] H-W. Koh and E. Hullermeier, *Mining gradual dependencies based on fuzzy rank correlation,* in Combining Soft Computing and Statistical Methods in Data Analysis. Advances in Intelligent and Soft Computing, Vol. 77, pp. 379-386. Springer Heidelberg (2010).

[13] A. Laurent, M.-J. Lesot, and M. Rifqi, *GRAANK: Exploiting rank correlations for extracting gradual itemsets,* in FQAS'09, LNAI 5822, Springer Berlin Heidelberg, pp. 382-393, (2009).

[14] A. Laurent, B. Negrevergne, N. Sicard, A. Termier, *PGP-mc: Towards a multicore parallel approach for mining gradual patterns,* in: Proc. DASFAA, pp. 78-84, (2010).

[15] A. Laurent, B. Negrevergne, N. Sicard, A. Termier, *Efficient parallel mining of gradual patterns on multi-core processors,* in AKDM-2, Advances in Knowledge Discovery and Management. Vol. 2. Springer, (2010).

[16] C. Molina, J. M. Serrano, D. Sanchez, and M. A. Vila, *Mining gradual dependencies with variation strength,* in Mathware & Soft Computing, Volume 15, pp. 75 - 93 (2008).

[17] P. L. Nancy, and C. Hao-en, *Fuzzy Correlation Rules Mining,* in Proceedings of the 6th WSEAS International Conference on Applied Computer Science, Hangzhou, China, pp 13-18, April 15-17 (2007).

[18] M. Quintero, A. Laurent, P. Poncelet, *Fuzzy Ordering for Fuzzy Gradual Patterns,* in FQAS 2011, LNAI 7022, Springer-Verlag Berlin Heidelberg, pp. 330-341, (2011).

[19] T. Rauber, G. Rnger, *Parallel Programming: for Multicore and Cluster Systems,* Springer-Verlag Berlin Heidelberg (2010)

[20] V. Stankovski, *Special section: Data mining in grid computing environments,* in CienceDirect Elsevier, Future Generation Computer Systems, vol. 23, pp. 31-33, (2007).

[21] R. Van der Pas, *An Overview of OpenMP,* In the OpenMP API specification for parallel programming, http://openmp.org/wp/resources/#Tutorials, [retrieved: May, 2012].

[22] D.-H. Weng, and Y.-L. Chen, *Mining fuzzy association rules from uncertain data,* in Knowl Inf Syst, vol. 23, Springer, pp. 129-152, (2010).

[23] D. Wilhelm, *Sparse Matrix Formats,* in ECE 250 Algorithms and Data Structures, Department of Electrical and Computer Engineering, University of Waterloo, Ontario, Canada, [retrieved: May, 2012].

[24] L. A. Zadeh, *Similarity Relations and Fuzzy Orderings,* in Information Sciences. Volume 3, Issue 2, pp. 177 - 200, (April, 1971).

 ISBN: 978-1-61208-217-2

Paper Recommendation System: A Global and Soft Approach

Siwipa Pruitikanee, Lisa Di Jorio
LIRMM - Univ. Montpellier 2 CNRS
Montpellier, France
Email: Siwipa.Pruitikanee@lirmm.fr, Lisa.Dijorio@gmail.com

Anne Laurent, Michel Sala
LIRMM - Univ. Montpellier 2 CNRS
Montpellier, France
Email: Anne.Laurent@lirmm.fr, Michel.Sala@lirmm.fr

***Abstract*—This Paper recommendation to researchers has been extensively studied in the last years, and many methods have been investigated for this purpose. In this paper, we propose a novel approach embedding the whole process for selecting papers of interest given some keywords. Our approach is based on a workflow integrating fuzzy clustering of the papers, the computation of a representative summary paper per cluster using Ordered Weighted Average (OWA) operators, and ranking, in order to answer user queries adequately. The originality of our method relies in the introduction of fuzziness for more flexibility in the approach. The use of representative papers allows us to summarize sets of papers into a single representative one, thus simplifying the user's interactions with the huge number of papers from the literature.**

Keywords-Paper Recommendation; Soft Computing; Fuzzy Clustering; Ordered Weighted Average (OWA)

I. INTRODUCTION

As the scientific literature is growing dramatically, selecting and reading papers has become a hard task, especially in the case of literature review. Digital libraries provide tools to help the user navigate through the resources and query the datasets. We discern many reasons for choosing and reading a paper; among them are the need to be aware of every new potential discovery in very specific domains, or the paper selection in a literature review process, as for example when writing an academic paper. In this context, recommending papers meeting some criteria such as the conference or author ranking is of great importance in order to avoid the time consuming step of reading many papers that are not so relevant to the subject.

Most of Digital libraries propose navigation tools, most of them based on multicriteria filtering and/or collaborative filtering [1], [2]. For this purpose, paper recommendation systems have been extensively studied in the last years Gipp et al. [3], Gori and Pucci [4] or Liang, Li and Qian [5]. Some tools have been created to group very similar papers using clustering methods, to provide organised information to the user. However, none of these tools is able to point out representative papers. Thus, the reader has no idea of the main methods described in these groups of papers and of the most representative of these methods.

In this paper, we propose a novel approach embedding the whole process. Our approach is based on a workflow composed of four steps.

The first step consists in selecting papers that are related to the user query. In this step, all papers containing at least one keyword among those included in the user query are selected.

The second step consists in grouping papers based on their similarity. For this purpose, we consider that papers are similar if they deal with the same topics. As we consider that it is not relevant to split objects in a crisp manner, we consider here fuzzy clustering.

The third step consists in computing representative papers, allowing us to resume sets of papers into one, thus simplifying the user interactions with the huge number of papers from the literature. We propose this representative paper to get enriched by a small number of other papers from the group, in order to cover all the topics of the user query.

The fourth step consists in ranking the representative papers so as to present the papers to the user in decreasing order of interest. In this step, we consider classical methods, such as PageRank.

The originality of our approach is twofold: in one hand we consider the whole steps of workflow whereas common approaches consider only specific steps of the process and on the other hand we introduce fuzziness in order to soften the approach.

The paper is organized as follows. Section II presents the existing work related to our approach. Sections III and IV introduce the running example and the formal framework we rely on in the proposition Section V details our proposition. Finally, Section VI draws some conclusions on our work and proposes research directions.

II. RELATED WORK

Paper recommendation systems lie at the intersection of different fields of data analysis: recommendation systems, text ranking and scientometry. In this section, we discuss of the main advances in each domain, and of the main drawbacks.

A. Paper Ranking Methods

Large efforts have been provided regarding the ranking of papers. Papers can be evaluated and compared using different criteria: the authors reputation, the date of

 ISBN: 978-1-61208-217-2

publication, the conference or journal ranking and the number of times it has been cited, are among the most often used information.

Citation count is one of the most used information. For example, the Thomson Scientific ISI impact factor (ISI IF) [6] is based on citation counts. It combines citation counts with a moving window to favor the most recent papers, and also include the impact of some journals in the calculation. However, methods based on the citation count suffer from the fact that paper impact is not taken into account. Thus, recent works have proposed a modified version of the ISI IF to integrate the "popularity factor", which is defined by the citation analysis of publication venues and the PageRank algorithm.

Krapivin and Marchese [7] modified the PageRank algorithm in order to apply it on academic papers. As in PageRank, the quality of a paper is based on the number of papers pointing to it, and its quality decrease if there are too much outgoing links (citations) from it. However, this approach suffers from the following drawback: some good papers (especially survey papers) need a lot of citation in order to contextualize their work. Moreover, as for PageRank, the algorithm has some difficulties to take very recent papers into account, no matter their quality.

B. Recommender Systems

Recommender systems are an important research field since the 90's, mainly because of their generic and industrial application. Roughly speaking, a recommender system takes some user interest or profile as an input, and searches among massive database information for items that the user has not seen and which he may be interested in. Recommender systems differ from classical data mining as it has to deal with specific user profile and result ranking.

There are two main paradigms in Recommender Systems: Collaborative Filtering (CF) and Content-Based Filtering (CBF). Note that some works propose "hybrid" approaches by mixing the two paradigms. The interested reader may refer to [8] for a detailed survey about these different approaches.

In Collaborative Filtering (D. Goldberg et al. [9], Ekstrand et al. [1]), the systems propose items to a user, by considering the items that similar users liked in the past. Thus, CF systems rely on rating and profiling. Such systems are quite mature and currently used in e-commerce websites. Among the weakness of such systems are the "cold start problem": when starting or adding new items, the system needs some elements to be initialised before being able to predict. When a new user is added, the system needs to profile him in order to make efficient recommendation. Finally, there is a sparsity problem, as there are only a small set of rates compared to the set of recommendation that has to be predicted.

In Content Based Filtering ([10], [11]), items that a user already pointed out as being of interest are used to recommend new items. Thus, the process can be seen as a classification task, where the training set is the user preferences. As it has been widely used in text-based context (internet, news,...) CBF systems mainly use information retrieval and information filtering methods. However, such systems can be limited by the problem of content analysis, because of the format of input items; while research reached a mature point concerning text-based documents, feature extraction from stream or video based document is much harder. Also, CBF systems are limited to what the user feeds them: they will never recommend items from another domain than those already rated by the user.

More recently, recommender systems have been extended to the paper recommendation context.

Torres et al. [12] proposes a hybrid approach that mixes CF and CBF. The authors detail a set of tools ranging from the simple CF system using k-nn algorithm and enriching data by adding cited papers to CBF using TF-Idf measure. Here, hybridation occurs by merging the results of CF and CBF. The author concludes that the hybrid system performs better than only CF or only CBF. Huang et al. [2] also proposes a hybrid approach based on graphs. It allows both for users and items integration in the system. The authors are then able to use classical graph search for extracting and recommending useful information. As Tores et al. [12], the authors show that hybrid approaches outperform CBF methods.

Gori and Pucci [4] propose a system based on a new random walk process and the citation graph, called Item-Rank. It is based on PageRank through its propagation and attenuation properties. In Agarwal et al. [13], a CF approach is done by clustering a subspace of papers. In this paper, the main goal is to apply the system to researchers working in the same laboratory. The originality of the method is the clustering algorithm that efficiently traverses the search space by subspace intersection. Yang et al. [14] describes a ranking-oriented CF system which extracts user's access logs as the training set. The system overcomes the cold start problem, however, weblogs stay noisy and not reliable data, Shahabi and Chen [15].

He et al. [16] uses different informations such as the title, the abstract, the sentences around a citation in order to build a citation recommender system. The main novelty is the user query form; it does not have to be a bibliography, it can also only be a document or some specific sentences.

In Gibb et al. [3], a user can give as an input an entire document. The process then uses every contextual information such as the citation analysis, authors, sources, implicit and explicit ratings. Moreover, the authors propose to use the Distance Similarity Index (DSI) and the In-text Impact Factor (ItIF). The authors build a system combining all user-given information parameters (for example an h-index range for author reputation) and provide a graphical user interface.

Liang et al. [5] only use the citation graph in order to output a small-sized set of relevant papers. They define measures working at two granularity levels: the Local Relation Strength measures the dependency between cited and citing papers, and then the Global Relation Strength captures the relevance between two papers in the whole citation graph. The Local Relation Strength relies on weighted parameters such as the number of times a paper is cited, and the number of times two papers are cited together, or the age of a publication. Then, the Global Relation Strength combines the Kratz measure [17] with the dependency in a citation link.

 ISBN: 978-1-61208-217-2

Sugiyama and Kan [18] use the user's recent research interests in order to recommend new papers. The work focuses more on the user profile: the author discriminates between junior researchers and senior researchers. The authors hypothesis is that contextual information about the user can provide evidence for recommendation. In this context, the information is provided by the user historical search. Then, the paper selection is driven by the prebuilt profile.

C. Ordered Weighted Average (OWA)

When aggregating information, many operators are available, Torra and Narukawa [19], as weighted average. The idea here is to combine N values into a single result. Yager [20] and Yager [21] propose the OWA operator, defined as below.

Definition 1: A vector $v = (v_1, \ldots, v_N)$ is a weighting vector of dimension N if $v_i \in [0, 1]$ and $\sum_{i=1}^{N} v_i = 1$.

Definition 2: A mapping AM: $R^N \rightarrow R$ is an arithmetic mean of dimension N if $AM(a_1, \ldots, a_N) = (1/N) \sum_{i=1}^{N} a_i$.

Definition 3: Let p be a weighting vector of dimension N. A mapping WM: $R^N \rightarrow R$ is a weighted mean of dimension N if $WM(a_1,\ldots,a_N) = \sum_{i=1}^{N} p_i a_i$.

Definition 4: Let w be a weighted vector of dimension N which correspondent with vector a. A mapping OWA: $R^N \rightarrow R$ in an ordered weighting average of dimension N if $OWA(a_1,\ldots,a_N) = \sum_{i=1}^{N} w_i a_{\sigma i}$, where σ is a permutation such that $\forall i \in [1,N-1]$, $a_{\sigma(i)} > a_{\sigma(i+1)}$

D. Fuzzy Clustering

Clustering consists in grouping together observations sharing the same characteristics, but without the help of predefined classes. Clustering method appeared in the 70's, and if some specific context still need to be explored, there exist several mature methods to compute this result, such as hierarchical clustering, K-means, C-means, etc. Some methods consider that clusters can overlap. These last solutions are known as *fuzzy clustering* Bezdek [22]. Every object then belongs to every cluster with a membership degree ranging from 0 to 1. (Fuzzy) Clustering is based on a distance measure which is used for describing to which extent two objects are similar.

Fuzzy C-means is one of the most often used method. Let us consider n objects $x_1, \ldots, x_n$ described over d attributes. The objective is to group these objects into k clusters, each cluster c_i $(i = 1, \ldots, k)$ being represented by its center v_i. Let $u_{i,j}$ be the degree of membership of the object x_i in the cluster c_j.

Let $\| * \|$ be any norm expressing the similarity.

$u_{i,j}$ is computed as:

$$u_{i,j} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|x_i - v_j\|}{\|x_i - v_k\|} \right)^{\frac{2}{m-1}}}$$

The algorithm relies on a iterative process that computes, for every object, the membership degree to every cluster and recomputing the center of the clusters. The degree of fuzziness of the process, impacting the overlapping rate of the clusters, is tuned using the m parameter.

III. Running Example

We consider the example detailed in Tables I, II and IV. In this example, we consider several papers that have been published on topics identified by keywords. These keywords can belong to more than one paper. These papers have been written by authors and cite some other ones.

The abstracts and titles allow us to identify keywords. For instance, let us consider the paper p1 published in 1996, it is related to data mining, with the following abstract:

"The mining of large databases is a very hot topic in database systems and machine learning. Companies have used some data mining techniques for understanding customer behavior on their data warehouse. This article provides a survey on the data mining techniques, classification and comparing of data mining techniques."

As a classical data mining process can not proceed such information, we need to make some technical steps. First of all, we remove stopwords and special characters from each title and abstract. Then, cleaned abstract texts can be mapped onto a word vector. Citations are considered as useful information in our approach. Informaly, citations can be viewed as a directed graph, with papers being vertices

TABLE I. Example Dataset - Papers

Paper Id	Title	Abstract	Conf/ Jal	Year
p1	A survey of Data mining techniques	... data mining ...	s1	1996
p2	Data Streams Mining with a Classifiers	...data mining... machine learning...	s2	2003
p3	Summarization k representative rules of frequent pattern	... mining... data ...	s2	2005
p4	Data mining in money laundering crimes	… mining... data ...	s2	2006
p5	Selection of relevant features and examples in machine learning	... machine learning ...	s3	1997
p6	Machine learning for automatic text classification	...learning ... mechine..	s4	2002
p7	Using Data Mining to Develope The Expert System	... machine learning ... data mining ...	s5	2004

TABLE II. EXAMPLE DATASET – PAPERS AND AUTHOR

Paper Id	Authors	Year	Number of Citation
p1	J. Martin, J. Smith	1996	3
p2	J. Martin, J. Smith	2003	1
p3	J. Martin, J. Jibb	2005	0
p4	M. Clark, L. Martinez	2006	0
p5	L. Davis, P. Green	1997	2
p6	L. Davis	2002	0
p7	F. Lee, H. Sweet	2004	0

TABLE III. EXAMPLE DATASET – CITATIONS

Paper Id	Cite to
p2	p1
p3	p1
p4	p1
p6	p5
p7	p5

TABLE IV. EXAMPLE DATASET – AUTHOR

Author Id	Author Name	H-index
A1	John Matin	88
A2	John Smith	78
A3	Jack Jibb	17
A4	Mark Clark	7
A5	Luis Martinez	5
A6	Lora Davis	57
A7	Pen Green	25
A8	Frank Lee	0
A9	Home Sweet	8

TABLE V. EXAMPLE DATASET-CONFERENCE/JOURNAL RANKING

Conf/Jal	Ranking
A1	88
A2	78
A3	17
A4	7
A5	5
A6	57
A7	25
A8	0
A9	8

TABLE VI. EXAMPLE DATASET-CITATION MATRIX

Paper Id	p1	p2	p3	p4	p5	p6	p7
p1	0	0	0	0	0	0	0
p2	1	0	0	0	0	0	0
p3	1	0	0	0	0	0	0
p4	1	0	0	0	0	0	0
p5	0	0	0	0	0	0	0
p6	0	0	0	0	1	0	0
p7	0	0	0	0	1	0	0

and a paper citation being a directed edge. We represented such a graph by the mean of a binary matrix, as showed in Table VI.

Moreover, we also assume that users would like to start with some strong references before going deeper in the search process. The interestingnesses of paper such as conference ranking, h-index of authors are considered into our process. Our idea is that the better the conference and h-index is, the higher the quality of the paper is. Thus, we propose to select the conference ranking using commonly agreed ranks on the main ranking websites and h-index; see the example in Table V and h-index in Table IV, respectively.

Our process will first select publications based on keywords, and then group similar papers by the mean of OWA operators. We consider that similarity can be measured with three attributes: title, abstract and citation or bibliography list. The OWA operator aggregates three similarities between papers in dataset, resulting in a matrix of aggregated similarity and fuzzy clustering are applied. Thus, two clusters will be created :

- cluster 1: {*p1, p2, p3, p4, p7*}
- *cluster* 2: {*p3, p5, p6*}

For each cluster, we select the representative papers by mean of the membership degree and interestingness measures of a paper.

- *cluster* 1: {p2, p3, p1}
- cluster 2: {*p6*, p3}

Then, to show the final output to user, each centroid of cluster is compared with the keywords. The papers in the cluster are ranked by interestingness as mentioned above; see the result in Table VII.

TABLE VII. EXAMPLE DATASET-RANKING RESULT

Cluster No.	Paper Id
1	p2
1	p3
1	p1
2	p6

IV. FORMAL FRAMEWORK

In this section, we present the seminal definitions for describing the data we are dealing with.

Let:

- $D = \{p_1, p_2, ..., p_m\}$ be a set of research papers
- $K = \{k_1, k_2, ..., k_n\}$ be a set of keywords
- $A = \{a_1, a_2, ..., a_q\}$ be a set of distinct authors

These sets are mapped using the following functions:

- *W:* $D \rightarrow P(A)$, where *W(p)* returns the set of authors of paper $p \in D$
- *T:* $D \rightarrow P(K)$, where *T (p)* returns the set of keywords embedded in the title of paper *p*
- *Ab:* $D \rightarrow P(K)$, where *Ab(p)* returns the set of keywords embedded in the abstract of paper *p*
- *C:* $D \rightarrow P(D)$, where *C (p)* returns the set of papers cited by paper *p*

V. PROPOSITION

Our proposition relies on a four-step process, starting from papers from several sources (e.g, Web of Science,

DBLP,local databases) and arriving to representative papers ranked regarding their interestingness, as shown in Figure 1.

The data pre-process has been done by collecting publication or academic paper data from multi-sources into one data structure. Our structure focuses on common attributes, being composed of title, authors, published date, source (e.g., journal) and citations or reference list.

A. Step 1: Selecting Papers

The process starts with publication selection and is based on keywords provided by the user in her/his query. This step returns the papers that match at least one of these given keywords. We thus obtain the preliminary related publications dataset. For instance, let us consider a user choosing the following two keywords: data mining and machine learning. Both of them are separated into four given words: data, mining, machine and learning, and use them for finding the publications from database; assume the result is Table I, which contains detail of each publication and Table VI that contains the list of citations from one paper to other ones.

B. Step 2: Grouping Papers

The second step consists of grouping the selected papers into clusters by creating similarity matrix among papers and using fuzzy clustering technique.

The Similarity σ between papers is computed by considering the titles, abstracts and common citations. We indeed assume that titles contain keywords, leading to the fact that if two titles share many common words, then this means they are similar. Moreover, we rely on the abstract as an indication of the content, thus assuming that common keywords lead to similar topics and interest.

Finally, as our approach aims at grouping papers that share common interest, we thus consider the co-citations.

These three criteria are aggregated using OWA so that it is possible to decide whether a representative paper is a paper being representative on all criteria or not.

Given two papers $d_1, d_2 \in D$, we thus have

$$\sigma(d1,d2) = \Theta(\sigma_K(T(d1),T(d2)), \sigma_K(Ab(d1),Ab(d2)), \sigma_C(C(d1),C(d2)))$$

where:

- Θ: $[0, 1]^n \rightarrow [0, 1]$ is an aggregation operator for fusing the three similarity degrees, e.g., Θ = OWA = average, min, max, …
- $\sigma_K : P(K)^2 \rightarrow [0, 1]$ is a function comparing two sets of keywords and returning a number ranging from 0 to 1 which estimates to which extent the sets of keywords are similar;
- $\sigma_C : P(D)^2 \rightarrow [0, 1]$ is a function comparing two sets of cited papers and returning a number ranging from 0 to 1 estimating the similarity extent of the set.

As it is not relevant to consider that papers can be split into several groups in a crisp manner, we use fuzzy clustering, thus outputting overlapping groups. In this framework, we compute the membership degree of every paper p_i to every cluster c_j using the following equation:

$$u_{i,j} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\sigma(p_i - v_j)}{\sigma(p_i - v_k)^{\frac{2}{m-1}}} \right)}$$

C. Step 3: Electing Representative Papers

This step aims at proposing a representative paper of every group.

A paper is considered as being representative if the topics are the ones that are shared in the group and if it has some criteria making it more interesting than other ones. For this purpose, the papers taken from a famous conference will be preferred to papers from non significant conferences. select the most nearest of center or ranking by interest.

Let c be a cluster containing the set of D_c papers, the representative paper $rep(c) \in D_c$ is computed as:

$$rep(c) = \arg\max_{p \in D_c \text{ and } p' \in D_c \setminus \{p\}} \sigma(p,p'),$$

As we assume that it is not possible to find out only one paper being representative enough, we associate every representative paper to some other ones to complete the keywords that are not covered by the representative, as shown in Figure 2.

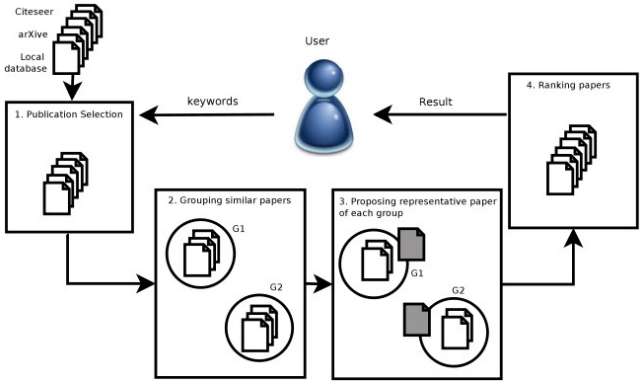


Figure 1. Method Overview

D. Step 4: Ranking Representative Papers

The last step aims to organize the final output to users. We propose two step ranking: external and internal ranking. The external ranking means to rank clusters comparing with query keywords while internal ranking is to rank papers in cluster by interestingness measures. Firstly, we rank clusters by similarity measures which are calculated from the distances of cluster centroid and query. Finally,

 ISBN: 978-1-61208-217-2

interestingness measures such as conference ranking and h-index are taken into account to rank papers in the cluster.

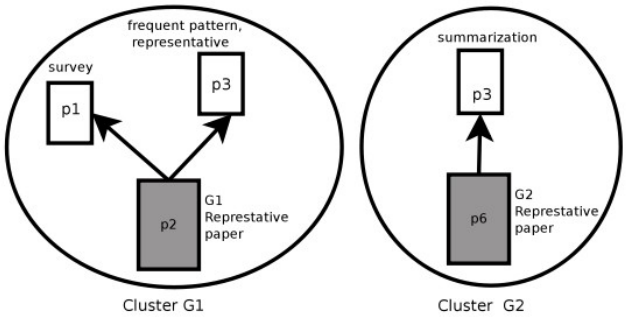


Figure 2. Representative Papers

VI. Conclusion and Future Work

In this paper, we presented our approach for paper recommendation. It relies on a workflow including soft approaches, thus allowing to take into account real dataset. It is indeed not relevant to consider crisp cuttings between papers and paper attributes. Current and future works include the deep study of the measures used in our approach, for exploring efficiency for both semantic and computational (memory and time) criteria, together with a study of the evaluation process, for enhancing precision/recall criteria that are often used to assess the methods.

We are also planning to further investigate the concept of representative paper, to determine if a single paper should represent a whole cluster. This question leads us to different approaches: on one hand, a paper could be representative only for one criterium, while on the other hand we could consider the generation of a cluster summary, for example by using text summary methods.

References

[1] Ekstrand, J. Riedl, and J. Konstan, Collaborative Filtering Recommender Systems. Now Publishers, 2011.

[2] Z. Huang, W. Chung, T.-H. Ong, and H. Chen, “A Graph-based Recommender System for Digital Library,” JCDL ’02, pp. 65–73, 2002.

[3] B. Gipp, J. Beel, and C. Hentschel, “Scienstein: A research paper recommender system,” in International Conference on Emerging Trends in Computing, 2009, pp. 309–315.

[4] M. Gori and A. Pucci, “Research paper recommender systems: A random-walk based approach,” in Web Intelligence,2006. WI 2006. IEEE/WIC/ACM International Conference on. IEEE, 2006, pp. 778–781.

[5] Y. Liang, Q. Li, and T. Qian, “Finding relevant papers based on citation relations,” in Proceedings of the 12th international conference on Web-age information management, ser. WAIM’11. Berlin, Heidelberg: Springer-Verlag, 2011, pp.403–414.

[6] E. Garfield and R. Merton, Citation indexing: Its theory and application in science, technology, and humanities. Wiley New York, 1979, vol. 8.

[7] M. Krapivin and M. Marchese, “Focused Page Rank in Scientific Papers Ranking.” in ICADL, ser. Lecture Notes in Computer Science, G. Buchanan, M. Masoodian, and S. J. Cunningham, Eds., vol. 5362. Springer, 2008, pp. 144–153.

[8] G. Adomavicius and A. Tuzhilin, “Towards the Next Generation of Recommender Systems: A Survey of the State of-the-Art and Possible Extensions,” IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, pp. 734–749, 2005.

[9] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, “Using collaborative filtering to weave an information tapestry,” Communications of the ACM, vol. 35, pp. 61–70, December 1992.

[10] R. Van Meteren and M. Van Someren, “Using content-based filtering for recommendation,” in Proceedings of the Machine Learning in the New Information Age: Mlnet/ECML2000 Workshop, 2000.

[11] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Analysis of recommendation algorithms for e-commerce,” in Proceedings of the 2nd ACM Conference on Electronic Commerce. New York, NY, USA: ACM, 2000, pp. 158–167.

[12] R. Torres, S. M. Mcnee, M. Abel, J. A. Konstan, and J. Riedl, “Enhancing digital libraries with TechLens+,” in JCDL ’04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries. New York, NY, USA: ACM Press, 2004, pp. 228–236.

[13] N. Agarwal, E. Haque, H. Liu, and L. Parsons, “Research paper recommender systems: A subspace clustering approach,” Advances in Web-Age Information Management, pp. 475–491, 2005.

[14] C. Yang, B. Wei, J. Wu, Y. Zhang, and L. Zhang, “CARES: a ranking-oriented CADAL recommender system,” in Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries. ACM, 2009, pp. 203–212.

[15] C. Shahabi and Y. Chen, “An adaptive recommendation system without explicit acquisition of user relevance feedback,” Distributed and Parallel Databases, vol. 14, no. 2, pp. 173–192, 2003.

[16] Q. He, D. Kifer, J. Pei, P. Mitra, and C. L. Giles, “Citation recommendation without author supervision,” in Proceedings of the fourth ACM international conference on Web search and data mining. New York, NY, USA: ACM, 2011, pp.755–764.

[17] D. Liben-Nowell and J. Kleinberg, “The link-prediction problem for social networks,” Journal of the American society for information science and technology, vol. 58, no. 7, pp. 1019–1031, 2007.

[18] K. Sugiyama and M.-Y. Kan, “Scholarly paper recommendation via user’s recent research interests.” in JCDL, J. Hunter, C. Lagoze, C. L. Giles, and Y.-F. Li, Eds. ACM, 2010, pp. 29–38.

[19] V. Torra and Y. Narukawa, Modeling decisions: information fusion and aggregation operators. Springer-Verlag New York Inc, 2007.

[20] R. Yager, “On ordered weighted averaging aggregation operators in multicriteria decisionmaking,” Systems, Man and Cybernetics, IEEE Transactions on, vol. 18, no. 1, pp. 183–190, 1988.

[21] R. R. Yager, “Families of OWA operators,” Fuzzy Sets Syst., vol. 59, no. 2, pp. 125–148, Oct. 1993. [Online]. Available: http://dx.doi.org/10.1016/0165-0114(93)90194-M

[22] J. C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algoritms. Plenum Press, New York, 1981.

 ISBN: 978-1-61208-217-2

 ISBN: 978-1-61208-217-2

Balancing Communication and Execution Technique for Parallelized Sparse Matrix-Vector Multiplication

Seiji Fujino and Takeshi Nanri
Research Institute for Infoemation Technology
Kyushu University
Fukuoka, Japan
Email: {fujino, nanri}@cc.kyushu-u.ac.jp

Ken'itiro Kusaba
Mitsubishi Electric Co. Ldt.
Chiyoda-ku, Marunouti
Tokyo, Japan

***Abstract*—This paper proposes a runtime optimization technique for load balancing parallelized SpMxV (sparse matrix-vector multiplication) with consideration of cost for both communication and execution time. In state-of-the-art iterative methods that call SpMxV repeatedly during iterations, this optimization utilizes the estimated time of communication and the measured time of execution for adjustment of balance of load among processes. Through numerical experiments of an iterative solver for linear systems, it will make it clear that the proposed optimization technique outperforms compared with the conventional technique.**

Keywords*-Sparse Matrix-Vector Multiplication; Load-Balancing; Bi_IDR(s) method.*

I. INTRODUCTION

In many scientific computations, SpMxV (sparse matrix vector multiplication) [5] [8] plays a crucial role. For example, for the purpose of solution of linear systems of equations which stem from FEM (Finite Element Method) analysis, SpMxV is often repeatedly executed and shares major part of the entire executing time. As a strategy for accelerating this operation, parallelization of SpMxV has been studied in many researches. The most popular way for parallelizing SpMxV is to attach calculation to processes by row of the target matrix. Since most of the entries in sparse matrices are zero, they are compressed so that only nonzero entries need to be stored and calculated. The execution time of each row depends heavily on the pattern of nonzero entries of matrix. This causes difficulty in balancing loads among processes in parallelized SpMxV.

This paper proposes a technique for load-balancing MPI-based SpMxV operation. There are a lot of researches for parallelizing SpMxV on distributed memory systems. Graph-partitioning based decomposition algorithms [1] are major approach among them. Due to the complexity of these algorithms, they are generally applied statically. On the other hand, to achieve better load-balance in MPI-based SpMxV, Lee *et al*. proposed a runtime technique called NRET (Normalized Row-Execution-Time-based iteration-to-process mapping) to adjust the mapping of rows to processes repeatedly [4]. In NRET, averages of measured time for calculation on each process are used to estimate the time of computing each row. Then, the mapping is adjusted according to this estimation so that it achieves better load-balance. However, since the technique does not consider the time for communication, the total loads among processes still remains to be unbalanced.

To solve this issue of unbalancing, we proposes Balancing Communication and Execution Technique (abbreviated as Balancing-CET). In a program that invokes SpMxV repeatedly, the technique adjusts mappings of rows to processes at each invocations in the beginning several iterations. The adjustments are done according to the estimation of the time for computation and communication on each process. The execution time is estimated from the measurement in the previous iteration, while the communication time is estimated by a linear performance model with the amount of data to be sent and received by each process. By repeating adjustment, this technique becomes close to the optimal distribution. This paper verifies effectiveness of Balancing-CET through numerical experiments of an iterative solver for linear systems.

This paper is organized as follows. In section 2 we make an overview of runtime load-balancing technology. In section 3, we describe costs for communication and execution in SpMxV. In section 4, through numerical experiments, we verify effectiveness of the proposed Balancing-CET. In section 5, we will draw some conclusions.

II. OVERVIEW OF RUNTIME LOAD-BALANCING TECHNOLOGY

The proposed Balancing-CET assumes a kind of program in which SpMxV is invoked many times with the same sparse matrix. In addition, it assumes that the program uses the result vector of a SpMxV as the vector to be multiplied to the matrix at the next SpMxV. These assumptions can apply to many scientific programs. Figure 1 presents the parallelized version of SpMxV program. Since this program is parallelized in a row-based manner, the result vector of SpMxV is distributed among processes.

 ISBN: 978-1-61208-217-2

Therefore, to use the result vector as the vector to be multiplied to the matrix in the next SpMxV, communication among processes must be done. The simplest but inefficient way of such communication is to invoke `MPI_Allgather`. Instead, this paper assumes that the program uses more efficient way proposed by Lee *et* a*l*. [4]. It reduces the amount of data to be transferred by letting each process send a minimum block that contains necessary data for each target process to calculate its part of SpMxV. Figure 2 sketches the flow of the communications.

After each execution of SpMxV, Balancing-CET checks if the load-balance is fine enough by comparing the execution time of each process. If the ratio of the difference between the maximum time and the minimum time among all processes is larger than a specified threshold, the algorithm invokes adjustment of distribution.

In the adjustment, Balancing-CET first calculates the average time of the total cost of previous SpMxV among all processes. Then it uses this time tCT as the target time for mapping rows to processes. Mappings are done row by row. As mapping one row to a process, estimated cost of communication and execution is added to the predicted execution time of the process. If the predicted execution time exceeds tCT, Balancing-CET changes the target process to the next one.

III. Costs for Communication and Execution in SpMxV

To reduce the overhead of adjustment, estimation of costs for communication and execution on each row are done in a simple way. Execution cost of each row is estimated by using the same method as NRET. In this method, the computation cost of *i*th row is estimated by the average time of calculating rows at the process which calculated *i*th row in the previous SpMxV.

Communication cost, on the other hand, is estimated by using a linear performance model of communication, $\alpha \times m + \beta$. In the model, α, β and m stands for the time for transferring one unit data, the latency of communication and the amount of data to be transferred, respectively. α and β are calculated at the beginning of a program by using the results of measuring time for some pingpong communication between processes. On the other hand, m is calculated for each row at the time the row is applied to a process. In applying a row to a process, the amount of data that should be sent and received is determined for each of other processes by using location of nonzero entries of the matrix. Figure 3 depicts how to derive the receiver processes of an entry. From these information, the additional costs for sending and receiving with this new mapping are calculated.

```
for i = start(myrank) to start(myrank + 1) - 1
   temp = 0.0
   for j = rowptr(i) to rowptr(i+1) - 1
      temp = temp + val(j) * x(colind(j))
end for
```

Figure 1. An Example of Parallelized SpMxV program.

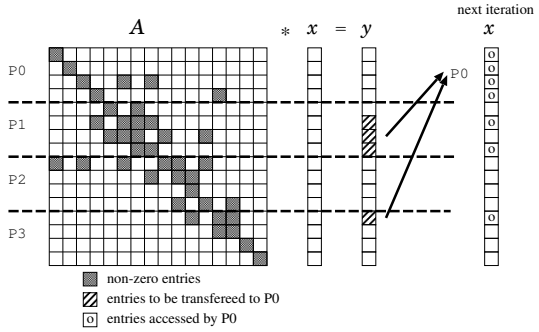


Figure 2. A Sketch of Communications for Transferring Minimum Block of Data Necessary for the Next SpMxV.

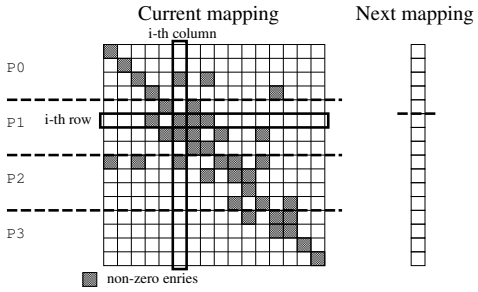


Figure 3. A Diagram of Determining the Target Process of Communications for an Entry.

IV. Numerical Experiments

To examine the effect of Balancing-CET, some numerical experiments have been done. The threshold for determining load-balance in Balancing-CET is set to 5%. In addition, the number that this technology is invoked is limited by 20 times. The experimental platform is a cluster of 16 PCs connected by InfiniBand. Each PC consists of 4 cores of Intel Xeon 3.0GHz and 8GB of RAM and runs with RedHat Enterprise Linux 3. The program is written by Fortran90 and MPI. The compiler and the MPI library are products of Fujitsu Corp. Each process of MPI programs is attached to one CPU core.

A. *Effect of Balancing-CET on SpMxV*

The performance of a program that repeats 1000 times of SpMxV is measured to examine the effects of NRET and Balancing-CET. After each SpMxV, communications are performed to use the result vector of it in the next SpMxV. As for test matrices, three matrices selected from Florida Sparse Matrix Collection [2] are used. Table I lists

 ISBN: 978-1-61208-217-2

specifications of test matrices with images that show their patterns of nonzero entries. As the images show, each matrix has different characteristics. In matrix matrix_9, nonzero entries are mainly located on a diagonal band with some exceptions on the right-most column. On the other hand, they are distributed throughout in matrix ecl32, and they are confined to a diagonal band in matrix xenon1.

Table I
INPUT MATRICES.

name	dimension	total nonzero entries	ave. nonzero entries
matrix9	103,430	2,121,550	20.51
ecl32	51,993	380,415	7.32
xenon1	48,600	1,181,120	24.30

Table II
SPECIFICATION OF TEST MATRICES.

pattern of nonzero entries	name	dimension	total nonzero entries	ave. nonzero entries
	matrix_9	103,430	2,121,550	20.51
	ecl32	51,993	380,415	7.32
	xenon1	48,600	1,181,120	24.30

Tables III - V show communication time and total execution time for matrices of matrix_9, ecl32 and xenon1. In these Tables, “No Opt.”, “NRET” and “Balancing-CET” mean the results without any load-balancing, with NRET and with Balancing-CET, respectively. “Procs” is the number of processes. “Comm” is the time for communication and “Total” is the total execution time. Hereafter time in Tables is measured in seconds.

Table III
COMMUNICATION TIME AND TOTAL EXECUTION TIME FOR MATRIX MATRIX_9.

Procs	No Opt.		NRET		Balancing-CET	
	Comm	Total	Comm	Total	Comm	Total
1	0.725	11.037	0.002	9.741	0.001	9.708
2	1.483	9.377	0.758	8.947	0.756	8.928
4	2.265	7.294	0.746	5.370	0.611	5.297
8	2.701	5.294	0.945	2.982	0.678	2.669
16	2.960	4.337	1.171	1.761	0.412	1.190
32	3.187	3.962	0.764	1.033	0.388	0.796

As shown in these Tables, NRET and Balancing-CET reduced the execution time significantly. With matrices matrix_9 and ecl32, the effect of Balancing-CET is better than

Table IV
COMMUNICATION TIME AND TOTAL EXECUTION TIME FOR MATRIX ECL32.

Procs	No Opt.		NRET		Balancing-CET	
	Comm	Total	Comm	Total	Comm	Total
1	0.320	2.423	0.001	1.912	0.001	1.934
2	0.726	2.001	0.300	1.380	0.262	1.351
4	1.036	1.843	0.474	1.111	0.467	1.110
8	1.181	1.554	0.812	1.109	0.800	1.067
16	1.135	1.334	1.074	1.231	0.956	1.143
32	1.185	1.332	1.143	1.235	0.960	1.101

Table V
COMMUNICATION TIME AND TOTAL EXECUTION TIME FOR MATRIX XENON1.

Procs	No Opt.		NRET		Balancing-CET	
	Comm	Total	Comm	Total	Comm	Total
1	0.347	7.436	0.002	7.013	0.002	7.016
2	0.719	5.237	0.177	4.573	0.156	4.555
4	0.969	3.573	0.262	2.620	0.260	2.628
8	1.089	2.220	0.298	0.970	0.301	1.024
16	1.091	1.532	0.197	0.412	0.179	0.441
32	1.126	1.345	0.139	0.274	0.156	0.355

NRET in most of the cases. On these matrices, it can be concluded that the considerations of communication costs in load-balancing worked well. With matrix xenon1, on the other hand, NRET showed better effects than Balancing-CET. Since nonzero entries of matrix xenon1 are confined in a diagonal band, imbalance of load for communication and execution is not significant. Therefore, both techniques could achieve sufficiently fine load-balance. In this case, the overhead for estimating costs of communications caused longer execution time than NRET.

To examine the detailed behavior of NRET and Balancing-CET, the number of rows attached to each process is studied. Figure 4 plots the case of matrix matrix_9 with 16 processes. “Opt.” is the number of rows attached to each process with Balancing-CET. Since this matrix has nonzero entries on the right-most column, the process P15 needs to invoke larger number of send operations than others. Therefore, Balancing-CET attached fewer number of rows to P15. In addition to that, it also reduced the number of rows on P3, P4, P7, P8 and P11 because they performed inter-node communications that are slower than intra-node communications.

B. Effect on an Iterative Solver for Linear Systems

To see the effects of Balancing-CET on more realistic situations, parallelized Bi_IDR(s) method [3] [6] [7] for solving linear systems has been examined. This solver is an enhanced version of IDR(s) method to achieve better stability with lower computation cost. In this program, not only SpMxV but some other vector operations, such as summation and dot-product, are performed repeatedly. Those operations are also parallelized into processes. In parallelized dot-product, MPI_Allreduce is invoked to calculate

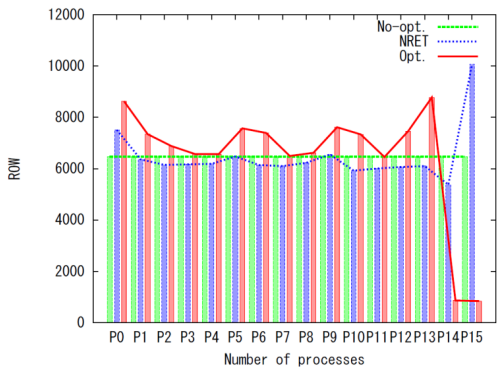


Figure 4. Fluctuation of Number of Rows Attached on each Process for matrix matrix_9.

the total sum among all processes.

Tables VI-VII tabulate total time in seconds, ratio and average of Bi_IDR(s) method with parameter s= 4 and 8 for matrix matrix_9. This program stops if the ratio of the 2-norm of the residual and of the initial residual is less than 10^{-12}. "Iter." and "Total" mean the number of iterations and the total time to satisfy this convergence test, respectively. "Ratio" as shown in Balancing-CET column means the time ratio of Balancing-CET to that of NRET. "TRR" means True Relative Residual for the approximate solution $\boldsymbol{x}_{k+1}$ as $||\boldsymbol{b}-A\boldsymbol{x}_{k+1}||_2/||\boldsymbol{b}-A\boldsymbol{x}_0||_2$.

As shown in Table VI, a mark "$_{**}$" indicates that measurement of time did not succeed. In the case with s=4 and number of processes is 8 of NRET in Table VI, TRR is very poor as "(-9.43)". However TRR of results of Balancing-CET is always sound. Moreover, in Table VII, the performance with Balancing-CET is more efficient than that with NRET as number of processes becomes larger. The performance degradation of Balancing-CET in the smaller number of processes is caused by the overhead for estimating communication cost. TRR of results of Balancing-CET is fairly good compared with that of results of NRET. Accordingly we may expect higher performance of Balancing-CET when we will solve large-scale problems.

Table VI
TOTAL TIME, RATIO AND AVERAGE OF BI_IDR(s) METHOD WITH $s = 4$ FOR MATRIX_9.

procs	No Opt.		NRET			Balancing-CET			
	Iter.	Total	Iter.	Total	TRR	Iter.	Total	Ratio	TRR
serial	5248	73.975	-	-	-	-	-	-	-
1	5248	146.126	5248	132.453	-11.80	5248	118.487	0.895	-11.80
2	5134	88.591	5192	80.878	-11.88	5593	79.072	0.978	-11.57
4	5421	72.605	5654	44.107	-11.55	5367	42.907	0.973	-11.88
8	5467	82.283	5334	**	-11.84	5417	23.175	-	-11.95
16	5887	91.571	5366	15.006	-11.37	5370	12.255	**0.817**	-11.71
32	5493	111.989	5305	12.723	(-9.43)	5308	11.240	0.883	-11.94
ave.	5442	-	5350	-	-11.31	5384	-	-	**-11.81**

Table VII
TOTAL TIME, RATIO AND AVERAGE OF BI_IDR(s) METHOD WITH $s = 8$ FOR MATRIX_9.

procs	No Opt.		NRET			Balancing-CET			
	Iter.	Total	Iter.	Total	TRR	Iter.	Total	Ratio	TRR
serial	4166	73.975	-	-	-	-	-	-	-
1	4166	125.880	4166	116.508	-10.31	4166	144.246	1.238	-10.31
2	4261	90.098	4350	80.769	-10.14	4192	83.897	1.039	-11.49
4	4134	62.663	4441	41.725	-10.36	4142	41.507	0.995	-10.50
8	4087	65.310	4285	23.563	-10.63	4249	22.364	0.949	-10.88
16	4098	65.551	4280	16.970	-10.85	4272	15.844	0.934	-10.49
32	4195	85.406	4256	15.389	-11.38	4145	12.299	**0.799**	-11.25
ave.	4157	-	4296	-	-10.61	**4194**	-	-	**-10.82**

V. CONCLUSIONS

This paper proposed Balancing-CET for load-balancing in SpMxV. It estimates costs of communication and execution time for achieving better load-balance. Through numerical experiments are simple, numerical results demonstrated effects of Balancing-CET.

REFERENCES

[1] UV. Catalyurek and C. Aykanat: Hypergraph-partitioning-based decomposition for parallel sparse-matrix vector multiplication, IEEE Transactions on Parallel and Distributed systems, vol. 10, no. 7, pp. 673-693, 1999.

[2] Sparse Matrix Collection: http://www.cise.ufl.edu/research /sparse/matrices/index.html

[3] S. Fujino, P. Sonneveld, Y. Onoue and M.B. van Gijzen: A Proposal of IDR(s)-SOR Method, Transaction of JSIAM, vol. 20, no. 4, pp. 289-308, 2010.

[4] S. Lee and R. Eigenmann: Adaptive runtime tuning of parallel sparse matrix-vector multiplication on distributed memory systems, Proceedings of the 22nd annual International Conference on Supercomputing 2008, pp. 195-204, June, 2008.

[5] Y. Saad: Iterative Methods for Sparse Linear Systems, 2nd ed., SIAM, Philadelphia, 2003.

[6] P. Sonneveld, M.B. van Gijzen: IDR(s): a family of simple and fast algorithms for solving large nonsymmetric linear systems, SIAM J. Sci. Comput., vol. 31, pp. 1035-1062, 2008.

[7] P. Sonneveld, M.B. van Gijzen: An elegant IDR(s) variant that efficiently exploits bi-orthogonality properties, Depart. of Applied Math. Anal., TR08-21(2008), Delft University of Technology.

[8] H.A. van der Vorst: Iterative Krylov Preconditionings for Large Linear Systems, Cambridge Univ. Press, Cambridge, 2003.

 ISBN: 978-1-61208-217-2

Implementation and Evaluation of Recurrence Equation Solvers on GPGPU systems using Rearrangement of Array Configurations

Akiyoshi Wakatani*
* *Faculty of Intelligence and Informatics*
Konan University
Higashinada, Kobe, 658-8501, Japan
wakatani@konan-u.ac.jp

***Abstract*—The recurrence equation solver is used in many numerical applications and other general-purpose applications, but it is inherently a sequential algorithm, so it is difficult to implement the parallel program for it. Recently, GPGPU (General Purpose computing on Graphic Processing Unit) attracts a great deal of attention, which is used for general-purpose computations like numerical calculations as well as graphic processing. In this paper, we implement a parallel and scalable algorithm for solving recurrence equations on GPUs by using CUDA (Compute Unified Device Architecture) and evaluate its effectiveness. The algorithm was originally implemented for MIMD parallel computers like a PC cluster and an SMP system by the authors and we modify the algorithm suitable for the GPGPU system by rearranging arrays configurations.**

***Keywords*-multithreading; tridiagonal solver; GPU; multi-core; CUDA**

I. INTRODUCTION

Recently, the peak performance of GPU (Graphic Processing Unit) has increased very much and outperforms that of general-purpose processors. Since past GPUs consisted of special-purpose hardware, they were used only for graphic processing and image processing. However, recent GPUs like GeForce 8 type of NVIDIA are composed of general-purpose unified shaders, so by using CUDA (Compute Unified Device Architecture) [1], they are used for general-purpose processing like numerical calculations as well as graphic processing.

Parallel applications having less data dependencies can be easily implemented on GPGPU systems, but complicated data dependencies prevent an optimal implementation of applications on GPGPU systems because we must carefully select which data should be kept in a small but fast memory. Linear first-order recurrence equations are expressed as $w_i = s_i \times w_{i-1} + t_i$, but these cannot be parallelized straightforwardly by dividing domains because the value of w_i is determined by using w_{i-1}. The recurrence equations are used frequently on many applications like Gauss elimination, the tridiagonal matrix solver and DPCM (Differential Pulse-Code Modulation) codec, so it is very important to implement the recurrence equation solver on GPGPU systems in order to achieve a high performance [11], [12].

In this paper, we modify the parallel algorithm of recurrence equations "P-scheme" suitable for GPGPU system and we evaluate the performance comparison of our methods on GPU and CPU. Note that P-scheme has been developed for distributed memory computers by the authors [2].

The rest of this paper is organized as follows: Section 2 presents the P-scheme algorithm and Section 3 summarizes the prior arts related to our method. Section 4 presents the experimental method and discusses the results and Section 5 concludes this paper with a summary.

II. RECURRENCE EQUATIONS

A. Tridiagonal system of equations

P-scheme is an algorithm that solves a recurrence equation in parallel. Our purpose is to parallelize a solver for the following tridiagonal system of equations of $A \times x = c$ where A is a tridiagonal matrix with $N \times N$ elements and x and c are vectors with N elements. The system is given by

$$x_0 = c_0 \quad (1)$$

$$-b_i \cdot x_{i-1} + a_i \cdot x_i - b_i \cdot x_{i+1} = c_i \, (1 \le i \le N-2) \quad (2)$$

$$x_{N-1} = c_{N-1} \quad (3)$$

where arrays a and b are elements of matrix A and array c is given in advance, array x is an unknown variable and N is the number of elements of the arrays.

B. P-scheme

It is known that the system given by Equations (1), (2) and (3) can be deterministically solved by Gaussian elimination, which utilizes two auxiliary arrays p and q.

$$p_0 = 0, \; q_0 = c_0 \quad (4)$$

$$p_i = \frac{b_i}{a_i - b_i \cdot p_{i-1}}, \; q_i = \frac{c_i + b_i \cdot q_{i-1}}{a_i - b_i \cdot p_{i-1}} \quad (5)$$

$$x_i = x_{i+1} \cdot p_i + q_i \quad (6)$$

The above procedure consists of a forward substitution (Equation (5)) and a backward substitution (Equation (6)). However, on parallel computers, the procedure cannot be straightforwardly parallelized due to the data dependency

 ISBN: 978-1-61208-217-2

that resides on both the forward and backward substitutions. Suppose that the number of processor is P, $N = P*M+2$ and arrays are block-distributed. Processor k $(0 \le k \le P-1)$ is in charge of M elements of arrays from $k*M+1$ to $k*M+M$, thus p_{k*M+1} can be calculated on processor k only after $p_{(k-1)*M+M}$ is calculated on processor $k-1$. Meanwhile, x_{k*M+M} can be calculated on processor k only after $x_{(k+1)*M+1}$ is calculated on processor $k+1$. These data-dependencies completely diminish the possibility of parallel computing.

We have proposed a parallel and scalable algorithm, called "*P-scheme*" [2], [3], [4]. We focus on array p on Equation (5) and explain how P-scheme works for it. Note that the equation for array p is a non-linear recurrence equation and the equations for arrays q and x are linear recurrence equations. So, our method can be easily extended to the equations q and x. Then we assume that p_{i-j} and p_i can be expressed by the following equation:

$$\frac{\beta_j + \gamma_j \cdot p_i}{\delta_j + p_i} = (-1)^j \cdot p_{i-j} \ (j > 0), \tag{7}$$

where β_j, γ_j and δ_j are auxiliary arrays that are defined below. By substituting Equation (5) to Equation (7), the following relation can be found.

$$\begin{aligned} & \frac{\frac{\delta_j + (-1)^{j+1} \cdot \frac{a_{i-j}}{b_{i-j}} \cdot \beta_j}{\gamma_j} + \frac{1 + (-1)^{j+1} \cdot \frac{a_{i-j}}{b_{i-j}} \cdot \gamma_j}{\gamma_j} \cdot p_i}{\frac{\beta_j}{\gamma_j} + p_i} \\ = \ & (-1)^{j+1} \cdot p_{i-(j+1)} \end{aligned} \tag{8}$$

Thus, β_j, γ_j and δ_j can be determined by the following system of equations:

$$\beta_1 = 1, \ \gamma_1 = -\frac{a_i}{b_i}, \ \delta_1 = 0 \tag{9}$$

$$\beta_{j+1} = \frac{1}{\gamma_j}(\delta_j + (-1)^{j+1} \cdot \frac{a_{i-j}}{b_{i-j}} \cdot \beta_j) \tag{10}$$

$$\gamma_{j+1} = \frac{1}{\gamma_j}(1 + (-1)^{j+1} \cdot \frac{a_{i-j}}{b_{i-j}} \cdot \gamma_j) \tag{11}$$

$$\delta_{j+1} = \frac{\beta_j}{\gamma_j} \tag{12}$$

It should be noted that β_j, γ_j and δ_j are independent of p_j. Hence p_i can be determined by using only p_0 as follows:

$$p_i = \frac{-\beta_i + (-1)^i \cdot p_0 \cdot \delta_i}{\gamma_i - (-1)^i \cdot p_0} \tag{13}$$

Therefore, if β_i, γ_i and δ_i are calculated in advance, p_i can be directly determined just after p_0 is determined.

By using the above relation, we proposed a scheme called *P-scheme(Pre-Propagation scheme)*, which consists of three phases. First of all, every processor simultaneously starts its calculation of β_i, γ_i and δ_i. This is called *pre-computation* phase. After that, processor 0 can directly determine p_M from p_0, β_M, γ_M and δ_M and sends p_M to processor 1. After receiving it, processor 1 can directly determine p_{2*M} from p_M, β_M, γ_M and δ_M and sends p_{2*M} to processor 2 and then processor 2 can directly determine p_{3*M} from received p_{2*M} and its auxiliary arrays and sends p_{3*M} to processor 3 and so on. This is called *propagation* phase. It should be noted that processor k can determine $p_{(k+1)*M}$ without calculating p_{k*M+i} $(1 \le i \le M-1)$. Finally, all processors can determine p_{k*M+i} $(1 \le i \le M-1)$ by using received data p_{k*M}. This is called *determination* phase.

The pre-computation and determination phases can be completely parallelized. Meanwhile the propagation phase is still sequential but the data to be exchanged is very slight, like just one data, so the communication cost is also expected to be very slight. The cost of the propagation phase is in proportion to the number of processors. Thus the total execution time is estimated by $O(\frac{N}{P}) + O(P) + O(\frac{N}{P})$. It is general that $O(P)$ is absolutely less than $O(N)$, because P is supposed to be much less than N.

Although the pre-computation and determination phases can be parallelized, the computational complexity is larger than the original substitution given by Equation (5). Since the original contains 1 multiplication, 1 division, 1 addition, 3 loads and 1 store and P-scheme contains 4 multiplications, 3 divisions, 3 additions, 6 loads and 4 stores, P-scheme must carry out over twice more computation than the original substitution. Execution times of the original substitution and P-scheme on PC (Pentium III (1 GHz), 1GB memory, GCC4.1.1 with O3) are shown in Figure 1. The graph shows that the execution time of P-scheme is about twice slower than that of the original substitution.

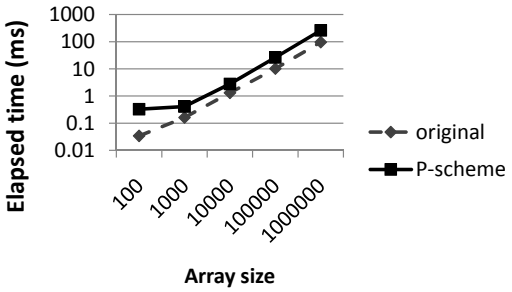


Figure 1. Comparison of execution times

III. RELATED WORKS

It is known that the first-order recurrence equation cannot be parallelized straightforwardly since the i-th element can be determined by using the $(i-1)$-th element. CR (Cyclic Reduction) and RD (Recursive Doubling) are recurrence equation solvers which can be directly applied on parallel computers [5], [6], [7], [8], [9].

 ISBN: 978-1-61208-217-2

A tridiagonal matrix can be solved by using recurrence equations and several tridiagonal matrix solvers have been implemented on GPUs. Kass et al. used ADI method for an approximate depth-of-view computation and solved the tridiagonal matrix by using CR method on a GPU [10]. Zhang et al. applied four methods (CR, parallel CR, RD and hybrid) to the tridiagonal matrix solver on the GPU and evaluated the performances to find that the hybrid method achieved the best performance [11]. Goddeke and Strzodka proposed mixed precision iterative solvers using CR method and implemented it on the GPU. They found that the resulting mixed precision schemes are always faster than double precision alone schemes, and outperform tuned CPU solvers [12].

When the size of the matrix is $N \times N$, the computational complexity of CR is $O(N)$ but it requires $2\log_2 N$ synchronizations between processors. Meanwhile, parallel CR requires only $\log_2 N$ synchronizations but its total computational complexity is $O(N \times \log_2 N)$. On the other hand, since the sequential algorithm (Gaussian elimination) consists of two recurrence equations (a forward substitution and a backward substitution) and both of the recurrence equations can be parallelized by using our method, those computational complexities are $O(N/P)$, $O(P)$ and $O(N/P)$, respectively. Our method also requires only two synchronizations for each recurrence equation. Then, we implement our method for recurrence equations on GPUs and evaluate the parallelism and the effectiveness of the rearrangement of array configurations in order to utilize the coalesced communication.

IV. Rearrangement of array configurations

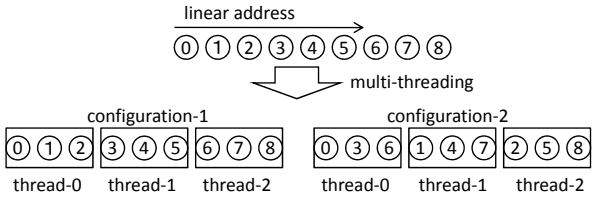


Figure 2. Rearrangement of array configurations

As mentioned before, the pre-computation and determination phases can be completely parallelized between threads, but the global memory accesses are done in either the coalesced communication or the non-coalesced communication, which depends on the array assignment. On the P-scheme algorithm for distributed memory computers, the i-th thread is in charge of the computations between $w_{i\times(N/P)}$ and $w_{(i+1)\times(N/P)-1}$ when w_i is distributed into P threads. On GPGPU systems, w_{0+k}, $w_{(N/P)+k}$, $w_{2*(N/P)+k}$, $w_{3*(N/P)+k}\cdots$ are concurrently accessed at the k-th step since calculations on GPUs are in principle SIMD calculations. However, these are accessed using the non-coalesced communication, so the access cost is very large.

In order to cope with this difficulty, array elements that are accessed simultaneously should be rearranged so that they are adjacent to each other.

$$w'_{i*P+j} = w_{j*s+i}\ (0 \le i \le s-1, 0 \le j \le P-1)$$

where $s = \frac{N}{P}$. Namely, this rearrangement is equal to the transposition of a $P \times s$ two-dimensional array into a $s \times P$ two-dimensional array.

Figure 2 shows an example of the rearrangement of array configurations. Suppose that the size of an array is 9 and the array should be divided into three parts. In an ordinary parallel computer like a PC cluster or an SMP system, the array should be just divided simply, so thread 0 is in charge of array elements 0, 1 and 2, thread 1 is in charge of array elements 3, 4 and 5, and thread 2 is in charge of array elements 6, 7 and 8, because this configuration (configuration 1) can enhance the locality of memory accesses and the efficiency of the cache memory. On the other hand, in a GPGPU system having NVIDIA's GPUs, the array should be rearranged (configuration 2) in order to utilize the coalesced communication, that is, thread 0 is in charge of array elements 0, 3 and 6, thread 1 is in charge of array elements 1, 4 and 7, and thread 2 is in charge of array elements 2, 5 and 8. So, at the first step, the threads access array elements 0, 1 and 2, and at the second step, the threads access array elements 3, 4 and 5, and so on. Thus threads can always coalesce their memory accesses into one memory request.

In the following subsections, we will evaluate the effectiveness of the rearrangement of arrays empirically by using the experiments.

V. Experiment and discussion

A. Experimental environment

Our experiments are carried out on the GPGPU system that consists of AMD Phenom II X4 945 (3.0 GHz), 4.0 GB memory and Tesla C1060 GPU (30 MPs and compute capability 1.3) under Windows 7 Ultimate and CUDA 3.0.

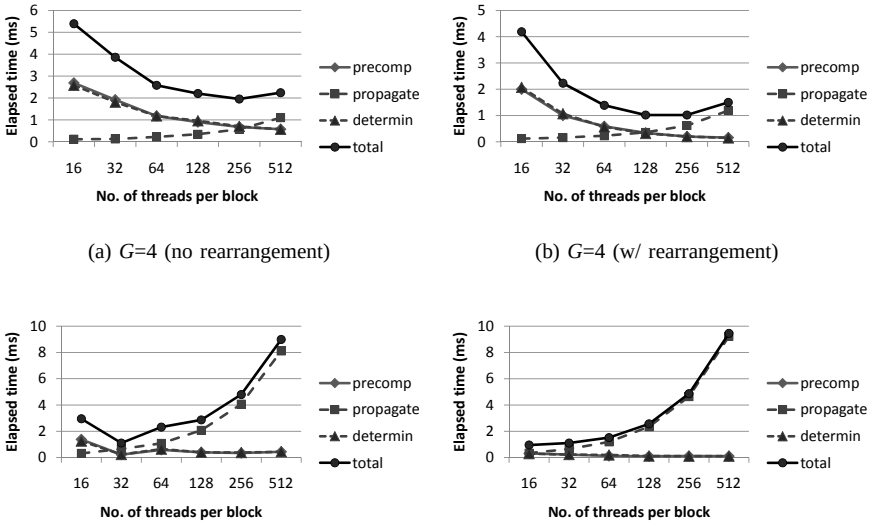
In order to evaluate our approach on the GPGPU system, we focus on the following linear recurrence equation:

$$\begin{aligned} w_0 &= C \\ w_i &= scale \times w_{i-1} + offset\ (1 \le i \le N) \end{aligned}$$

where C, $scale$ and $offset$ are constant. The value of N is set to 2^{18} (small arrays) and 2^{20} (large arrays), and we construct G thread blocks having T threads and execute them in parallel on GPUs, that is, the total number of threads is $P = T \times G$.

The elements of arrays are fetched from the global memory to registers of SPs and the results are directly stored to the global memory, so the shared memory is not used because no data is repeatedly used in our method. Therefore we do not care the bank conflict of the shared memory. The

 ISBN: 978-1-61208-217-2



(a) G=4 (no rearrangement)

(b) G=4 (w/ rearrangement)

(c) G=32 (no rearrangement)

(d) G=32 (w/ rearrangement)

Figure 3. Results of $N = 256K(= 2^{18})$

global synchronization is implemented by invoking different kernels. Since our method consists of three phases, only two global synchronizations are required between the phases. So, since the overhead of the synchronization is quite small, it does not affect the total performance.

B. Occupancy and threads

Occupancy is one of performance metrics that predict the effective performance on GPU execution. As mentioned earlier, one MP has 8 SPs and each SP executes one thread, so the efficiency of the MP does not reach 100% unless there are at least 8 threads. Due to the difference between the clock frequencies of the SP (shader clock) and that of the MP (core clock), at least 4 threads should be concurrently executed on one SP in order to keep the instruction pipeline of the MP full. Therefore, it is recommended that at least 32 threads should be placed on each MP and this size (32) is called "warp." Moreover, since the access latency to the global memory is large, the large size of the thread group is preferable for hiding the latency.

In the CUDA execution environment, G thread blocks are assigned to MPs and T threads are assigned to SPs within each MP. As mentioned before, the number of SPs within a MP is 8, but at least 32 threads, namely the size of the warp, must be assigned to one MP in order to maintain the efficient execution. Moreover, more threads should be assigned to each MP in order to improve the occupancy. On the other hand, the GPGPU system that is used for our experiment has 30 MPs, so G should be around 30 but it may be less than 30 for the optimal P. Since P is $T \times G$ and the execution times of the pre-computation and determination phases are in inverse proportion to P, they decrease when T and G increase. However, the execution time of the propagation phase increases when T and G increase, so the total execution time may be worsen. Therefore, one of purposes of our experiments is to confirm the contribution of T and G to the execution time.

C. Experiment 1 (small arrays)

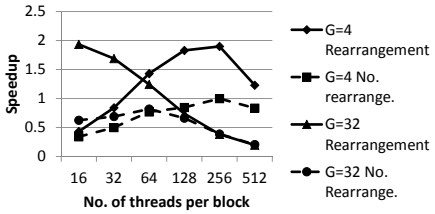
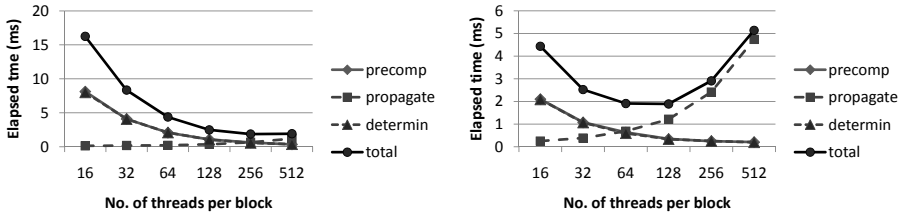


Figure 4. Speedups of $N = 256K(= 2^{18})$

The experimental results with the array size of $256K(= 2^{18})$ are shown in Figure 3. In the figure, the execution times of the pre-computation, propagation and determination phases are illustrated when G is 4 and 32 and T is varied from 16 to 512. It should be noted that the elapsed times of the pre-computation and the determination are same on the whole, and thus these lines are almost overlapped.

For both cases with no rearrangement and with the rearrangement of array configurations, the execution times of these phases decrease as P increases. For example, when G is 4 and the rearrangement is used, the execution times of the pre-computation phase with T of 16, 32, 64 and 128 are 2.07

 ISBN: 978-1-61208-217-2



(a) Elapsed time (G=4) (b) Elapsed time (G=16)

Figure 5. Results of $N = 1M(= 2^{20})$ (w/ rearrangement)

msec, 1.07 msec, 0.59 msec and 0.34 msec, respectively. When the rearrangement is not used, the execution times of the pre-computation phase with T of 16, 32, 64 and 128 are 2.69 msec, 1.92 msec, 1.18 msec and 0.9 msec, respectively. When T is 64 and the rearrangement is used, the execution times of the pre-computation phase with G of 4 and 32 are 0.59 msec and 0.13 msec, respectively. When the rearrangement is not used, the execution times of the pre-computation phase with G of 4 and 32 are 1.18 msec and 0.59 msec, respectively. As P increases, the area where each thread is in charge is getting smaller, so the overhead like a thread creation increases relatively. Note that the speedup seems to be flat when T is over 8, because the number of SPs per MP is 8, but, as T increases, the occupancy increases until the number of threads reaches the warp size (The occupancy is 1.0 when a MP of Tesla C1060 has 128 threads). Therefore, by increasing T (over 8), the performance can be improved. It should be also noted that, by rearranging the array configuration and using the coalesced communication, the performance can be enhanced from 2 to 10 times, which depends on the value of G.

On the other hand, as P increases, the execution time of the propagation phase increases. For example, when G is 4 and the rearrangement is used, the execution times of the pre-computation phase with T of 16, 32, 64 and 128 are 0.25 msec, 0.38 msec, 0.69 msec and 1.21 msec, respectively. Since the propagation phase is carried out on one SP, there is no difference between the execution time using the rearrangement and that without the rearrangement.

The comparisons of the speedups with a variety of parameter settings based on the execution time of the CPU are shown in Figure 4.

On the whole, the speedups using the rearrangement outperform those without the rearrangement. When the rearrangement is used, the difference of the speedups is small since the change of the value of T does not result in the difference of the execution time so much. However, when the rearrangement is used, the value of T and P decide whose phase should be dominant among the execution times of three phases, so an optimal value of T and P results in the largest speedup. For example, when G is 4, the maximum speedup is 1.9 with the value of T of 256. As mentioned below, $G \times T$ is constant when the combination of G and T results in the maximum speedup.

Moreover, when G and T are large, the value of the speedup using the rearrangement is almost identical to that without the rearrangement. For example, this is true when $G = 32, T = 128, 256, 512$. The reason is that the propagation phase is dominant among three phases when G and T are large. But there is no difference between the execution time using the coalesced communication and that using the non-coalesced communication, since the propagation phase is carried out on one SP.

D. Experiment 2 (large arrays)

A part of the experimental results with the array size of 1 M $(= 2^{20})$ are shown in Figure 5. The results of this case are almost equal to those of the case with the size of 256 K. Namely, as P increases, the execution times of the pre-computation and the determination phases decrease. It is also found that the performance can be enhanced from 2 to 10 times by using the rearrangement of array configurations and reducing the access cost to the global memory. As P increases, the execution time of the propagation phase increases. The difference from the case with the size of 256 K is that the absolute time of the pre-computation and the determination phases increases due to the increase of the array size and then the execution time of the propagation phase is relatively smaller than the 256 K case. Therefore, the optimal value of P is larger than the 256 K case.

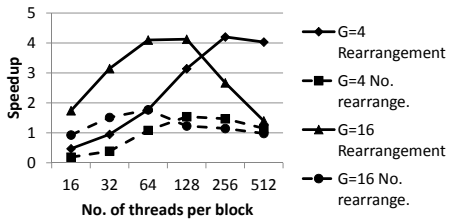
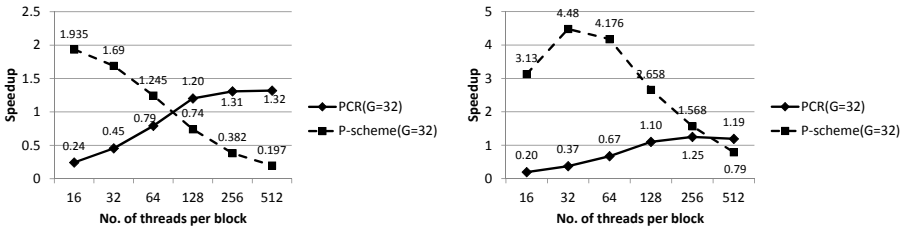


Figure 6. Speedups of $N = 1M(= 2^{20})$



(a) N=256K (b) N=1M

Figure 7. Comparison with PCR (Parallel Cyclic Reduction)

The comparisons of the speedups with a variety of parameter settings based on the execution time of the CPU are also shown in Figures 5 and 6 .

The trend of the results of the speedups are also similar to that of the 256 K case except that the value of T for the maximum speedup is smaller than the 256 K case.

E. Comparison with PCR

We compare the performance of our methods with PCR (parallel cyclic reduction) method [11] and show the experimental results in Figure 7. The maximum speedup of the P-scheme is larger than that of the PCR method for both cases because the computational complexities of the PCR method and the P-scheme are $O(N \cdot \log N)$ and $O(N)$, respectively. However, the speedup depends on the size of the thread block very much, so the tuning parameter for GPGPU programs must be carefully selected in order to achieve the maximum speedup.

F. Discussion

We discuss the optimal combination of G and T when the coalesced communication is used. The speedups using the rearrangement of array configurations when N is 256 K and 1 M are shown in Figure 8,

As shown in previous sections, the execution times of all the phases are estimated as follows:

$$pre-comp = \alpha \cdot \frac{N}{P} \quad (14)$$

$$propagation = \beta \cdot P \quad (15)$$

$$determination = \gamma \cdot \frac{N}{P} \quad (16)$$

where α, β and γ are the execution costs per data for the pre-computation phase, the propagation phase and the determination phase, respectively.

The computational complexities of these three phases are almost identical. However, while both the pre-computation phase and the determination phase are executed on T threads (over 1 warp) within thread blocks, the propagation phase is carried out only on one thread. Thus, since the core clock is 4 times slower than the shader clock, we assume the computational complexities of all the phases as follows:

$$\alpha : \beta : \gamma \simeq 1 : 4 : 1. \quad (17)$$

The parallelism is in proportion to G when G increases until the number of MPs, but it is almost flat when G is over the number of MPs. Since Tesla C1060 has 30 MPs, we evaluate the cases with the value of G of up to 32 in our experiments. Each MP has 8 SPs, so the parallelism is in proportion to T until T reaches 8 and it is in quasi-proportion to T until the warp size (32). When T is more than 32, the occupancy is getting close to 1.0 but the increase of the parallelism is almost flat. Therefore, in order to achieve the maximum speedup by increasing P, the following policy should be applied: 1) G should be maximized first and 2) the optimal T should be selected.

By using Equations (14), (15) and (16), the execution time t and the optimal parallelism P_{opt} are determined as follows:

$$t = \alpha \cdot \frac{N}{P} + \beta \cdot P + \gamma \cdot \frac{N}{P}$$

$$P_{opt} = \sqrt{\frac{(\alpha+\gamma)N}{\beta}} = \sqrt{\frac{N}{2}}. \quad (18)$$

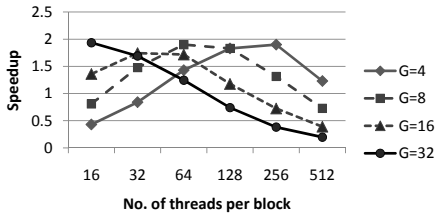
When N is 1 M, P_{opt} is nearly equal to 720, so T should be 180, 90, 45 and 22.5 for the cases with the value of G of 4, 8, 16 and 32, respectively. According to Figure 8-(a), T for the maximum speedup is 256, 128, 64 and 32 when G is 4, 8, 16 and 32. Thus, the estimation and the experimental results are identical. Moreover, When N is 256 K, P_{opt} is nearly equal to 360, so T should be 90, 45, 22.5 and 11.25 for the cases with the value of G of 4, 8, 16 and 32, respectively. According to Figure 8-(a), T for the maximum speedup is 256, 64, 32 and 16 when G is 4, 8, 16 and 32. Thus, the estimation and the experimental results are almost identical for this case as well.

Therefore, the maximum speedup is achieved when G is 32. So it is indicated that the optimization policy described above is rational. Namely, G should be maximized first and

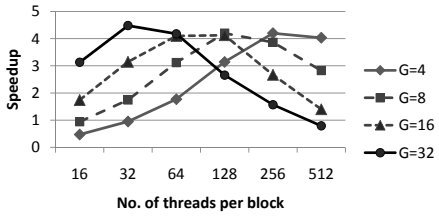
 ISBN: 978-1-61208-217-2

then the optimal T should be selected in order to achieve the maximum speedup.

It should be noted that P_{opt} is $\sqrt{\frac{N}{2}}$ and thus $\frac{N}{P_{opt}}$ is $\sqrt{2N}$. Therefore the computational complexity of the propagation phase is $O(P_{opt}) = O(\sqrt{N})$ and it is not larger than the computational complexities of other phases ($O(\frac{N}{P_{opt}}) = O(\sqrt{N})$).



(a) N=256K



(b) N=1M

Figure 8. Comparison of speedups

VI. Conclusion

We implemented a parallel recurrence equation solver on GPGPU systems by using CUDA and evaluated the effectiveness of the rearrangement of array configuration in order to utilize the coalesced communication. We also proposed a policy to decide the optimal number of threads per thread block and the optimal number of thread blocks in order to maximize the efficiency of parallelism.

In the near future, we will apply the recurrence equation solver to real applications. We will also try to implement our approach using OpenCL that recently attracts a lot of attention.

Acknowledgment

This work was supported in part by MEXT, Japan.

References

[1] D. Kirk, and W. Hwu, Programming Massively Parallel Processors: A Hands-on Approach. Morgan Kaufmann, Massachusetts, 2010.

[2] A. Wakatani, “A Parallel and Scalable Algorithm for ADI Method with Pre-propagation and Message Vectorization,” Parallel Computing, vol. 30, pp. 1345-1359, 2004.

[3] A. Wakatani, “A Parallel Scheme for Solving a Tridiagonal Matrix with Pre-propagation,” Proc. 10th Euro PVM/MPI Conference, Venice, 2003, pp. 222-226.

[4] A. Wakatani, “A Parallel and Scalable Algorithm for Calculating Linear and Non-linear Recurrence Equations,” Proc. Int’l Conf. Parallel and Distributed Computing and Networks, Las Vegas, 2004, pp. 446-451.

[5] R. Hockney and C. Jesshope, Parallel Computer 2. Taylor & Francis, London, 1988.

[6] J. Lopez and E. Zapata, “Unified Architecture for Divide and Conquer Based Tridiagonal System Solver,” IEEE Transactions on Computer, vol. 43, pp. 1413-1425, 1994.

[7] O. Eğecioğlu, et al., “A Recursive Doubling Algorithm for Solution of Tridiagonal Systems on Hypercube Multiprocessors, ” J. of Comput. and Applied Mathematics, vol. 27, pp. 95-108, 1989.

[8] E. Dekker and L. Dekker, “Parallel Minimal Norm Method for Tridiagonal Linear Systems,” IEEE Transactions on Computer, vol. 44, pp. 942-946, 1995.

[9] D. Lee and W. Sung, “Multi-core and SIMD architecture based implementation of recursive digital filtering algorithms, ” Proc. IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), 2010, pp. 1550-1553.

[10] M. Kass, A. Lefohn and J.D. Owens, “Interactive Depth of Field Using Diffusion,” Technical report 0601,Pixar Animation Studios, 2006, pp. 1-8.

[11] Y. Zhang, L. Cohen and J.D. Owens, “Fast Tridiagonal Solvers on the GPU,” Proc. PPoPP 2010, Bangalore, 2010, 10 pages.

[12] D. Goddeke and R. Strzodka, “Cyclic Reduction Tridiagonal Solvers on GPUs Applied to [Mixed Precision Multigrid,” IEEE Transactions on Parallel and Distributed Systems, vol. 22, pp. 22-32, 2011.

 ISBN: 978-1-61208-217-2

Statistical Analysis of Cost of Energy Due to Electricity Outages in Developing Countries

Venkat Natarajan
Staff Researcher, Frugal Innovation
Intel Lab, Bangalore, India
venkat.natarajan@intel.com

Amit S. Closepet
Intern, Frugal Innovation
Intel Labs, Bangalore, India
amit.s.closepet@intel.com

***Abstract*—This paper describes significant cost saving opportunities for consumers in developing countries by the use of computational intelligence and demand-side-management techniques to mitigate the massive use of diesel back-up during grid outages. We propose novel statistical algorithms to model electricity outages, heavy diesel use and associated customer energy costs in developing countries. Using a blackout simulator and Monte Carlos analysis, we assess the impact of different blackout types such as scheduled, unscheduled and annual forecasted outages on consumer energy costs. Variables for the analysis included time of outage (morning, evening etc.), duration of outage (1-5 hours), Diesel costs (20cents/KW-hr) and type of outage. We found that cost savings opportunities by the use of demand side management to mitigate outages can exceed 30% in many cases.**

Keywords-computational intelligence; algorithms for energy management; stochastic grid; power outages; energy cost; developing countries

I. INTRODUCTION

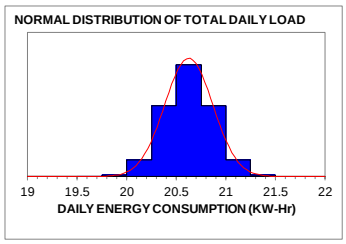
Consumers in developing countries are faced with major challenges with the power grid such as rampant grid outages (Table 1), unreliable power quality with fluctuating voltages and power, uncertain grid restoration schedules, etc. Consumers need to invest heavily in back-up systems (e.g., diesel) and often pay as much as 40-50% of their monthly power bills on back-up diesel costs [1-3]. The customer is often faced with difficult decisions such as whether compromise and curtail his loads or pay high costs of diesel to run his normal loads. There is a lack of computational tools to effectively predict, manage and optimize back-up systems in countries such as India. Also, there is a lack of demand side management approaches to help the consumer to mitigate outages. This study has two goals: one, to statistically model and understand the energy cost behavior caused by a stochastic grid and second, to define the maximum cost-savings potential by the use of demand-side-management techniques.

Risk management for power outages is an enormous challenge for consumers in a developing country. There is the value of lost load, which is essentially the loss that the consumer suffers on account of unsupplied power. The back-up system needs a large capital and recurring operating expenditures. Managing resources such as diesel storage, scheduling factory operations with built-in grid-fail-safe systems, providing adequate and expensive equipment protection including cascading power failures are commonly encountered issues. While diesel generators offer a simple and effective solution, there are many challenges that the emerging market customer needs to address. These mainly include optimal sizing of the diesel systems, minimizing op-ex costs by intelligent forecasting, risk mitigation by proper resource allocation etc. Figure 1 shows a typical model for a grid wherein all the system parameters (power quality, system voltages consumer loads) behave as statistical distributions. Furthermore, one also finds that all dependent parameters such as economic metrics (cost) behave similarly. In this study, we are exploring the use of normal distributions for all system parameters. The paper first proposes a novel approach to model grid outages using different computational techniques, analyzes impact of different outage scenarios on a home consumer and characterize the high energy costs to the consumer by use of diesel. Finally, as a means to cut this high cost, we recommend the use of load optimization or demand-side management and assess the maximum potential savings possible.

TABLE I. RECENT POWER OUTAGES IN INDIA [1]

Year	Indian City	Power Outage Per Month in Peak Season (Hours)	Type of Power Outage
2012	Chennai	60	Planned
2012	Vijayawada	60	Planned
2012	Hyderabad	90	Planned
2012	Haryana State	150	Planned
2012	Jahangirpur	180	Planned
2011	Maharashtra State	16	Planned
2011	Pune	90	Unplanned
2011	Mumbai Suburbs	7	Unplanned
2011	West Bengal Outside Kolk	150	Planned
2011	Gaya	20	Unplanned
2011	Coimbatore	60	Planned
2011	Vijayawada	60	Planned
2011	Madurai	120	Planned
2010	Gulbarga-Bidari	12	Unplanned
2010	Visakhapatnam	60	Unplanned
2010	Chennai	5	Unplanned

 ISBN: 978-1-61208-217-2



Grid Parameters
(Mean μ, Variance σ^2)
Grid Availability,
Outages
Power Quality
Consumer Load Profile
& Cost Parameters
Daily Energy

Figure 1. Model for Stochastic Grid Behavior in Developing Countries.

II. LITERATURE SURVEY

Several studies have focused on the impact of unreliable grids as seen in the following works: An in-depth investigation into the impact of power outages for consumers and businesses in Africa is performed in [4]. This study also assesses the economic consequences of the unreliable grids. A report on real power cost in India [5] reveals that the overall intent of providing cheap and affordable power to the consumers in the country is noble, but if the supplies are inadequate or unreliable, the consumers could actually end up paying a much higher price.

A report from United Nations [6] provides directions to expand access of modern energy services at the household level. An application of combined model of extrapolation and correlation techniques for short term load forecasting of an Indian substation is presented in [7]. Specific opportunities for DSM in the Indian scenario are presented in [8]. Low-cost energy generation using bio-mechanical energy is presented in [9] and this provides technology options for both off-grid users as well as on-grid users who have unreliable power. Specific demand-side-management techniques to mitigate power outages are proposed and assessed in [10]. There is enormous body of literature on the use of demand side management algorithms for power and cost optimization for the end consumer as follows. The use of casual scheduling loads with time-varying prices using stochastic dynamic programming is studied in [11] and its effect on consumer cost are reported. In [13], a power scheduling protocol for demand response in smart grid system is explored which focuses on limiting the allowable power loads. Algorithmic enhancements to a scheduler for residential DSM are presented in [15].

The problem of excessive power outages, its impact to the consumer and mitigation strategies need to be addressed as it is a major painpoint to consumers in developing countries. There are no published algorithmic works that predict the diesel requirements under periodic and rampant outages such as seen in emerging countries. There is also a lack of computation methods in the literature that tackle the problem of load-optimization or demand-response during outage. In this study, we develop computational methods that specifically focus on power outages and quantify their impact using different statistical means. Furthermore, we show the potential savings possible through the use of load optimization.

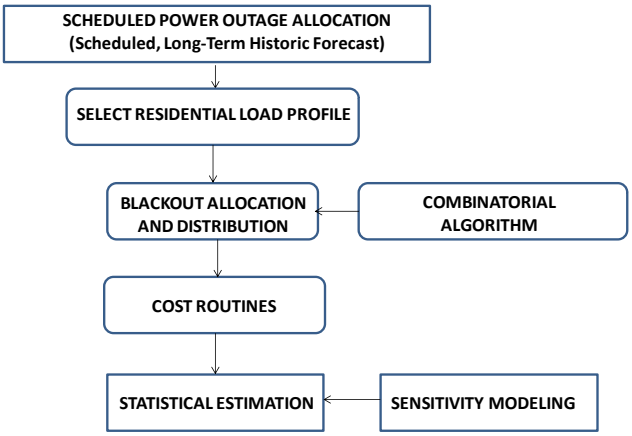


Figure 2(a). Fast Blackout Assessment Tool Architecture.

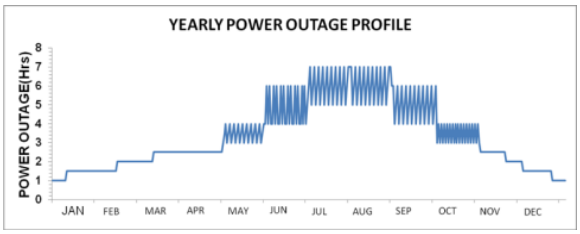


Figure 2(b). Historical Profile of Yearly Outages – Example.

III. SIMULATION APPROACH

Computational methods used in the present study are described here. To assess different blackout scenarios, a blackout assessment tool is developed (Fig. 2(a)). In this study, we have explored the use of normal distributions to model the different statistical parameters. Based on our data gathered from 20 households in India, normal distribution appears to fit the data well. The types of power outage scenarios modeled we studied include: scheduled, unscheduled and long-term historic forecast. For each case, we study the timing (time of day) and duration of power outage. For long-term forecast analysis, since one does not know that actual distribution of electricity outage, combinatorial algorithms are developed to determine the maximum cost impact. Statistical estimation methods such as Poisson estimation, Monte-Carlo limit analysis are built-in the tool for cost benefit analysis. Output parameters for the tool are statistical distributions for consumer load profile, daily energy consumption, energy costs (e.g. grid cost, diesel cost, daily energy cost), power outages (duration and length), effective cost of power etc.

 ISBN: 978-1-61208-217-2

The key outcomes of the tool are as follows. First, the statistical impact of an unpredictable and uncertain grid combined with usage of diesel back-up is evaluated to determine the actual consumer cost of energy use. Second, maximum cost savings potential that can be achieved by the use of demand-side-management techniques is reported. In the fast analysis tool, the user can select the desired blackout scenario. For example, to perform a scheduled blackout analysis, the user can input a fixed power outage time in a day. The blackout profile is then generated using the user input and the analysis and sensitivity modeling is performed using the cost inputs. An unscheduled power outage can be triggered in a given timeframe or if the user desires, the tool can automatically select create unscheduled outages. Large numbers of profiles (10000) are created for each configuration.

For long-term forecasting, the analysis requires knowledge of historical data. Since most cities in emerging markets do not have organized historical blackout data, a typical power outage profile for a year (Fig. 2(b)) is created for this study. Actual historic data can be input into the tool as required. The Poisson estimator is used to determine the expectation value of blackout duration in a given day. Since the number of ways that this power outage can be distributed in a day is extremely large, the tool uses a combinatorial methodology to allocate the distribution of the blackout. This method is based on the classic balls and bins allocation problem wherein each of the n balls is allocated to a set of m bins in a random, but uniformly distributed, way. The algorithm developed uses an iterative solver combined with a random distributor. The user can generate the specified number of distributions for the required design of experiments. In this study, it was found that 500 iterations are sufficient to provide statistically meaningful data. Nominal costs assumed for the grid is 5c/KWhr and baseline diesel costs assumed in the simulation are 20c/KWhr. The standard deviations assumed are approximately 10%. Sensitivity analysis is performed for diesel cost ratio from 1X to 3X from current diesel costs.

It is useful to calculate an effective cost parameter, C_{eff}, which captures the net cost of power to the customer. For a typical supply configuration with the grid and a diesel system, this would provide an average combined cost per unit for the customer. This is valuable to the customer as it provides a way for the customer to quickly estimate his costs, provide insight into quality of the grid by understanding the deviation of cost w.r.t. to the grid ad provide a benchmark and standardized method to compare grid stability and quality across customers and geographical regions. This effective cost parameter can be represented as,

C_{EFF} = Daily Cost Per Day/ Energy Consumption
= (Grid + Diesel Cost)/Energy Consumption

IV. RESULTS AND DISCUSSION

The key results of this study are presented in this section. From the consumer viewpoint, several parameters are of interest such as load distribution, distribution of electricity outage duration and timing, daily energy expenditure and increase in energy cost due to diesel usage, fraction of load executed during outage, and finally, the maximum cost savings possible by applying demand-side-management. Baseline cost data for running the loads on grid power are shown in Figure 3 for an Indian home consumer. Sample results from analysis of scheduled power outages are presented in Fig. 4. Herein, we show the results for an outage scenario of 5 hours commencing at 12PM. In all, four different outage start times are simulated and for each, the duration of power outage is varied statistically. The average cost of energy is roughly 2X grid cost (fig. 4(a)) and diesel expenditure (Fig. 4(b)) as a fraction of total energy cost is approximately 66% which is significant. Again, the rise in daily energy cost is a 100% over a grid-only cost.

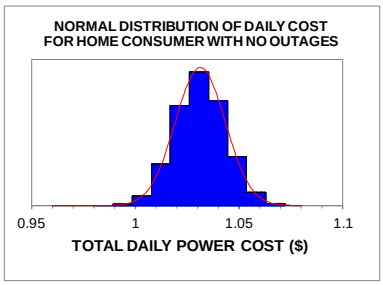
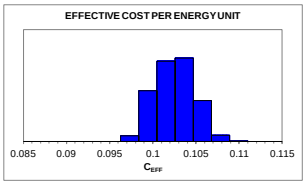
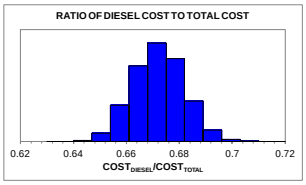
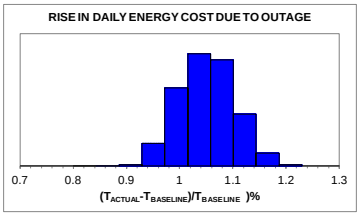


Figure 3. Results of Daily Energy Cost.





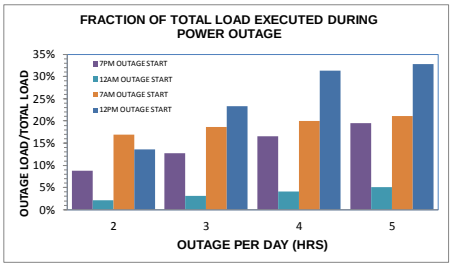
(a) Effective Cost of Energy (b) Ratio Diesel/Total Costs



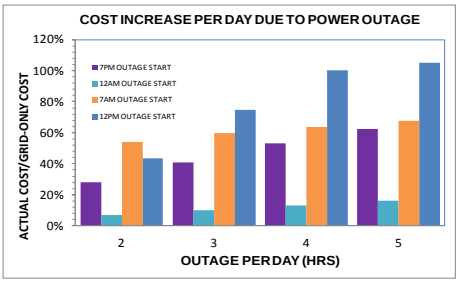
(c) Rise in Daily Energy Cost Due to Outages

Figure 4. Sample Results.

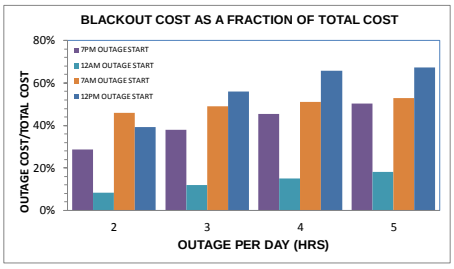
Figure 5 provides further details on the consumer cost and load execution patterns for a residential consumer as a function of the outage time. A power failure during afternoon has the largest impact on consumer costs, followed by morning and evening times. For an outage that is 5 hours long, the maximum load executed is as much as 35% of total daily load and for cases wherein the blackout times are <5 hours, the load fraction can be as high as 25%. An important finding is that in order to execute only 15% of the daily load, the consumer spends as much as 40% of daily total energy cost (Fig. 5(b)). For most power outage scenarios simulated the consumer pays ~40% over the normal grid-only costs for load execution. To execute 30% of the daily load, the consumer cost increases by 2X compared to the baseline case. In other words, a customer in a developing country could spend half of his power expenditure for back-up power during peak seasons when the power outages are incessant. The ratio of diesel costs over the total cost ranges from a typical value of 35% to 65% per day. Maximum diesel usage is either at 7AM or 12PM.



(a) Fraction of Total Load Executed During Outage



(b) Cost Increase Per Day Due to Power Outages



(c) Diesel Usage Cost as Fraction of Total Cost

Figure 5. Key Results of Scheduled Power Outage.

Next, we consider the cases of unscheduled power outages wherein the timing and duration of the outage are random variables (Fig. 6, Table 2). It must be stated that unscheduled power outage generator model needs to be calibrated with local grid conditions and measurements to provide accurate results. In this study, two types of unscheduled power outages are simulated. First, unscheduled power outage distributed through the day as multiple outages and second, the unscheduled random power cuts are limited to a part of the day. Both these scenarios are fairly common in developing countries. As described earlier, a random variable generator is used to simulate the unscheduled power outage scenario. In our model, the average unscheduled power generated is 4.9 hours per day. Unlike scheduled outages, it is difficult to execute power saving approaches such as demand side management during unscheduled blackouts due to its random nature. Stochastic modeling combined with machine learning to refine the predictive models can potentially be used to address the problem of unscheduled power outage management.

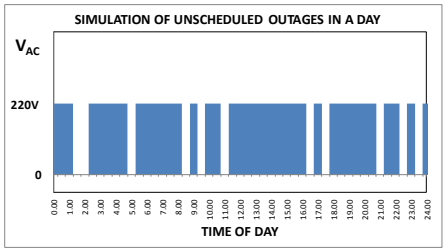


Figure 6. Unscheduled Multiple Power Outages in a Day.

TABLE II. GRID PARAMETERS FOR UNSCHEDULED BLACKOUTS

UNSCHEDULED POWER OUTAGE TIMING	FRACTION OF LOAD EXECUTED DURING OUTAGE	COST INCREASE/ DAY	DIESEL COST/TOTAL COST
ENTIRE DAY	20%	64%	49%
7PM - 10PM	2%	8%	9%
12AM-3AM	1%	2%	2%
7AM-10AM	4%	12%	12%
12PM-3PM	4%	47%	28%

TABLE III. MAXIMUM COST SAVINGS POTENTIAL BYUSING DEMAND-SIDE TECHNIGIES

	MAXIMUM POSSIBLE COST SAVINGS BY LOAD-SHIFTING			
	POWER OUTAGE START TIME			
OUTAGE (HRS)	7PM	12AM	7AM	12PM
2	22%	6%	35%	40%
3	28%	9%	37%	42%
4	34%	11%	38%	49%
5	38%	14%	40%	50%

 ISBN: 978-1-61208-217-2

TABLE IV. EFFECTIVE COST PARAMETER PER UNIT OF ENERGY DUE TO OUTAGE

	EFFECTIVE COST PARAMETER C_{EFF}			
	POWER OUTAGE START TIME			
OUTAGE (HRS)	7PM	12AM	7AM	12PM
2	0.06	0.05	0.08	0.07
3	0.07	0.05	0.08	0.09
4	0.08	0.06	0.08	0.10
5	0.08	0.06	0.08	0.10

TABLE V. SENSITIVITY ANAYSIS OF COST PARAMETERS DUE TO OUTAGES

	COST INCREASE		BLACKOUT COST/TOTAL COS		MAX. SAVINGS BY DSM	
DIESEL PRICE	2 HRS	5 HRS	2 HRS	5 HRS	2 HRS	5 HRS
Current	43%	105%	39%	67%	40%	50%
1.25X	54%	130%	43%	71%	35%	57%
1.5X	68%	164%	48%	75%	40%	62%
2X	95%	230%	55%	80%	49%	70%
3X	150%	361%	65%	85%	60%	78%

Finally, we show that the high power back-up costs due to use of diesel during electricity outages can be significantly mitigated through the use of intelligent scheduling or demand-side-management (DSM) techniques such as load shifting, peak-shaving and valley-filling. These techniques enable the user to execute the full load requirement with simultaneous cost savings. Since grid power is typically inexpensive, the purpose of the demand-side-management method is to shift the loads to the grid when it is available. As a result, the maximum cost savings potential is calculated when the entire load is shifted outside the power outage region and executed on grid power. However, in reality, the actual savings will be dictated by the specific distribution of schedulable and interruptible loads. The present data provides an upper limit achievable for the demand response schemes and the actual implementation of the DSM during power outages is presented in [10]. The maximum potential savings possible is generally upwards of 30% for most cases modeled (Table 3) and maximum DSM gains are possible when the power outage occurs during the afternoon. Measure of the grid cost disparity is given in Table 4 through the effective cost parameter. The efficiency of a demand-side management scheme is measured by how close the effective parameter is to the grid cost. For most cases in the present study, the grid disparity ranges from 1.5X to 2X the cost of per unit of grid power. Table 5 shows the sensitivity analysis using an increasing cost of diesel (varied from 1X to 3X to the current cost of diesel). The energy costs, based on this sensitivity analysis, show that, the effective cost of power can be upwards of four times the cost of a grid supported load. Such high costs can currently be seen in remote locations wherein availability and accessibility to diesel fuel is a challenge. As expected, the corresponding potential for load optimization is also high.

Long-term forecasting using historical data enables system planners to mitigate business risks due to grid uncertainty. Unfortunately, such historical data is largely not available except for a few areas. It is anticipated that upon the advent of the smart grid, highly localized behavior models of the grid will be available for the customer. In this study, we show a methodology to analyze historical data to predict annual diesel estimates for the consumer. Using combinatorial algorithms, one can generate different blackout scenarios. The costs are calculated for each individual case and statistical analysis is performed on the entire set of data. This is used to forecast diesel projections for an entire year as represented in Fig. 7. Overall, in summer months of May till November, users incur heavy cost due to power outages.

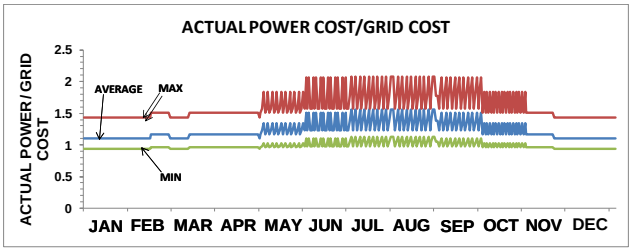


Figure 7. Forecasting of Yearly Cost for Home Consumer.

V. CONCLUSIONS

Computational intelligence methods are used to explore the problem of rampant power outages and associated high diesel back-up costs in developing countries. Using consumer load profiles, outage information and energy cost information, we show that in order to execute only 15% of the daily load, the consumer spends as much as 40% of the daily total energy cost. Further, to execute 30% of the daily load, the consumer cost roughly doubles. That is, a customer in a developing country could spend half of his power expenditure for back-up power during peak seasons when the power outages are incessant. In the case of unscheduled outages caused by random system failures, Monte Carlo analysis shows that the cost impact to the customer can be anywhere as much as 60% more than normal operation. Similar behavior is also seen for annual forecasting of power outages and consumer costs. Assessment of the maximum cost-savings potential using demand-side-management methods is seen to be huge and upwards of 30%.

VI. REFERENCES

[1] Power Annual report 2010-12, Central Electricity Authority, Govt. of India, www.cea.nic.in, [Retrieved: July, 2012]

[2] "Millennium Dev. Goals in Africa", 2006, United Nations Report, www.un.org [Retrieved: June, 2012]

[3] Development Goals in Africa Electricity Supply Monitoring Initiative (ESMI), Research Report, Prayas, India, www.un.org, [Retrieved May 2012]

[4] Vivien Foster and Jevgenijs Steinbuks, "Paying the Price for Unreliable Power Supplies" In-House Generation of Electricity by

 ISBN: 978-1-61208-217-2

Firms in Africa, Research Report, April 2009, www.worldbank.org, [Retrieved July 2012]

[5] "Real cost of power" 2009, Research Report, Universal Consulting, www.universalconsulting,com [Retrieved May 2012]

[6] "Energy poverty" Special excerpts of the World Energy Outlook 2010 for the UN General Assembly on the Millennium Development Goals Sept. 2010, www.un.org [Retrieved June 2012]

[7] Rabindra Behera, Bibhu Prasad Panigrahi1, and Bibhuti Bhusan Pati, "A Hybrid Short Term Load Forecasting Model of an Indian Grid" Energy and Power Engineering, 2011, 3, pp. 190-193, http://www.scirp.org [Retrieved March 2012]

[8] S. Padmanaban and Ashok Sarkar "Electricity Demand Side Management In India –Strategic & Policy Perspective", 2005, www.usaid.org [Retrieved July 2012]

[9] Venkat Natarajan, Amit Baxi, Ramanath Padmanabhan, and Vincent Mageshkumar, "Low-Cost Bio Mechanical Energy Generator for Off-Grid Users in Emerging Markets", 2011, IEEE Global Humanitarian Technology Conference, pp. 253-256, http://doi.ieeecomputersociety.org [Retrieved Dec 2011]

[10] Venkat Natarajan and Amit Closepet, "Application of Demand-Side-Management Techniques to Manage Electricity Outages in Emerging Markets", accepted for *ENERGYCON* 2012, Italy, www.energycon2012.org

[11] Tung Kim and Vincent Poor, "Scheduling Power Consumption with Price Uncertainty", IEEE Trans Smart Grid, 2011, pp. 519-527, ieeexplore.ieee.org [Retrieved March 2012]

[12] Amir-Hamed Mosenian-Rad and Alberto Leon-Garcia, "Optimal Residential Load Control With Price Prediction in Real-Time Electricity Pricing Environments", IEEE Trans. Smart Grid, 2010, pp. 120-133, ieeexplore.ieee.org [Retrieved March 2012]

[13] Michael Angelo A. Pedrasa, Ted D. Spooner, and Ian F. MacGill, "Scheduling of demand side resources using binary particle swarm optimization," IEEE Trans. Power Syst., vol. 24, no. 3, Aug. 2009, ieeexplore.ieee.org [Retrieved May 2012].

[14] K. Herter, "Residential implementation of critical-peak pricing of electricity," Energy Policy, V35, Apr. 2007, www.elsevier.com [Retrieved June 2012].

[15] Michael Angelo A. Pedrasa, Ted D. Spooner, and Iain F. MacGill, "Coordinated Scheduling of Residential Distributed Energy Resources to Optimize Smart Home", IEEE Trans. Smart Grid, 2010, Vol. 1, pp. 134-143, ieeexplore.ieee.org [Retrieved July 2012]

 ISBN: 978-1-61208-217-2

The Ontological Programming Paradigm

Valeriya Gribova, Alexander Kleschev
Intelligent Software Laboratory
Institute of Automation and Control Processes,
Far Eastern Branch of RAS
Vladivostok, Russia
gribova@iacp.dvo.ru, kleschev@iacp.dvo.ru

***Abstract* — This paper is devoted to the new ontological programming paradigm, an evolution of the declarative programming paradigm. The data model, basic structures of the language, and example are described.**

Keywords - programming paradigm; data model; semantic network; ontology.

I. INTRODUCTION

Software development is a time consuming process. Even more difficult is software maintenance. They both require new approaches from developers to resolve these problems. The declarative programming paradigm and languages realized in this paradigm were proposed as an approach to the problem mentioned above.

In general, a program in a declarative language is a description of an abstract model of the task to be solved, or, in accordance with [1], an executing specification of a result of computation. The programmer does not have to describe a process to control computation; it is the function of the language processor. Among other advantages is the fact that it is easier to write, to understand, and to maintain programs using languages of this paradigm in comparison with those written in imperative languages.

Declarative languages comprise logical and functional languages. However, the main idea of declarative programming in the modern logical and functional languages has not been realized completely yet. As a result, programs written in Prolog [2] and Lisp [3] are not considerably easier to develop, understand, and maintain than programs written in imperative languages. Among other drawbacks of declarative languages are poor facilities for user interface realization, and the difficulty of including imperative operators, if necessary.

The aim of this report is to suggest a new programming paradigm, called the ontological programming paradigm, as a further evolution of the declarative programming paradigm and to suggest an ontological programming language (OPL) satisfying the declarative programming definition, where a program is an ontology of results of computation. Mechanisms for user interface realization and facilities for including imperative structures are proposed.

The paper has the following structure. The problem statement is discussed complication of processes of development, modification, and understanding a program in the imperative, functional, and declarative paradigms; the ontological programming paradigm and its basic principles are described. Then the data models of the paradigm, basic structures of the ontological programming language are suggested. At the end of the paper the expert system of medical diagnosis using OPL is presented.

II. PROBLEM STATEMENT

Output data of task solving are the result of a computation process. The complications of program writing for solving a certain task are the following: the developer has to understand a set of computation processes to obtain results (extension of a task) for various possible input data, and to specify this set in a programming language (to write a program) [4]. The complications of a program modification during the life cycle are the following: the maintainer must recover extension of the task and comprehend why the computation processes result in exactly these output data. Then he or she must understand how to change these processes in order to obtain new output data and then modify the program.

We will discuss how computation processes to obtain results are connected with programs for obtaining these results in the imperative, functional, and declarative paradigms.

The basis of the imperative paradigm is computational models [5, 6]. The process of obtaining results in these models is a sequence of states, the first of them is generated from input data using an input procedure, every follow-up state is generated from the previous one using an operator of direct processing; the terminal state gives the output result. All the states of the computation process, except for the terminal state, are only indirectly connected to the output result. In currently-used imperative languages, the state of a computation process is a set of variable values, and the operator of direct processing is the assignment operator. Using this operator, the next state is a modification of a variable value; the remainder variables keep their values. Sequences of operator execution in programs written in imperative languages are defined by linear fragments of the program, conditional operators, and cycle operators (and also a procedure call).

The basis of the functional paradigm is the lambda calculus [5, 6]. The process of obtaining the result can be represented by an oriented marked network of a function call. The label of every terminal vertex is input data, the label of every non-terminal vertex is a function value. Arguments of this function are labels of arcs outgoing from this vertex (the arc orientation is from the result to the argument). The computation result is the label of the network root. All temporary values (labels of non-terminal vertexes), except

 ISBN: 978-1-61208-217-2

for the root label, are indirectly connected to the output result. In current functional languages there is a set of basic functions and facilities for designing of new functions from basic and already defined (among them being conditional terms and recursion).

The basis of the logical paradigm is the first order predicate calculus [5, 6]. The process of obtaining result can be represented by an oriented marked network of result inference. The label of every terminal vertex is a relationship tuple representing input data; the label of every non-terminal vertex is a relationship tuple representing the result of applying a rule to premises. Premises are labels of arcs outgoing from this vertex (the arc orientation is from the consequence to premises). The computation result is the label of the network root. All temporary values (labels of non-terminal vertexes), except for the root label, are indirectly connected to the output result. A program in a logical language is an inquiry and a set of rules (implications) and facts (relationship tuples).

The computation result in each of the three paradigms is obtained at the last step of the computation process, all remainder steps are indirectly connected to the output result and are temporary values.

Thus, to simplify the program development and its understanding and modification, it is necessary to suggest a programming paradigm where processes of obtaining result, represented by oriented graphs are direct. It means that a fragment of a result is formed at the every step of the computation process. In this case a program is an executable specification of a set of results of computation (but not a set of indirect processes of obtaining them). Developing such a program the programmer must only specify a set of results of computation; analyzing such a program the programmer must only conceive a set of results obtained; modifying such a program the programmer must only understand how to change the specification of the set of computation results to obtain required changes of these results, and every changing is local. The specification of a set of results, according to the up-to-date view in the artificial intelligence is an ontology of computation results. So, we name the suggested paradigm the ontological programming paradigm. It is considered as a more complete realization of the declarative programming paradigm (functional and logical paradigms).

III. THE DATA MODEL OF THE PARADIGM

All data in the ontological programming paradigm are semantic networks. The semantic network is an oriented graph without cycles; all arcs of the network have labels, vertexes can be simple and structural; the network vertex without entering arcs is the root. Every simple terminal vertex of the network has a label. A label is a constant of a type. The root of the network has two labels. The first of them is the label of a class (the name of a function), the second is the individual label of this network. Every structural vertex is a container comprising an ordered set of hierarchical semantic networks with the same label of a class and different individual labels.

The semantic network may be stored constantly (out of the programs) or temporarily (within the program). Using semantic networks as a data model means that objects of processing are integrated information structures. For example, integrated information structures in the expert system of medical diagnosis are: a case report, a knowledge base, and an explanation represented in the form of semantic networks. It is the difference of the ontological paradigm from others that support processing informational structures divided into some fragments (for example, some objects). Relations between these fragments are only known to programmers.

IV. THE APPLICATION

In general, an application has one or some output semantic networks (results), may have/do not have one or some input semantic networks (input data), and also may have/do not have one or several temporary semantic networks (temporary data). Every temporary or the output semantic network is computed by a function (the name of the function is a label of a class of the semantic network root).

Every function may have/do not have input semantic networks and the only output semantic network and does not have any temporary networks.

Input semantic networks may be stored constantly (out of the application) or temporary (within the application), so output semantic networks of an application or a function may be stored both constantly and temporary.

Among all semantic networks of an application (input, output, and temporary ones) a partial order is defined. Therefore, an application is a superposition of these functions, connected with computation of all semantic networks of an application, except for input semantic networks stored constantly. Thus, application executing is computation of the output semantic network using input semantic networks stored constantly and created before application launching (if any). Every function of an application is an ontology (structure) of an output semantic network created by the ontological programming language.

V. THE ONTOLOGICAL PROGRAMMING LANGUAGE

The OPL is a visual logical language of programming. A function is the ontology of the output semantic network. It is an oriented graph with the marked vertex and arcs. Arc labels are terms which become arc labels of an output semantic network of a function in the process of application executing. A vertex label is logical formulas.

The computation process of an output semantic network of a function built this network starting from the root. During the process of semantic network building one-to-one correspondence between the vertex and arcs of the output semantic network and the vertex and arcs of its ontology (function) is determined. The output semantic network can be built as both automatic and interactive ones.

Figure 1. The simple formula

 ISBN: 978-1-61208-217-2

Logical formulas of the language are: a simple formula, a unary formula, a propositional formula, a simple quantifier formula, a structural quantifier formula and a set of implications.

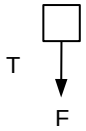


Figure 2. The unary formula

The simple formula (see Figure 1) is a graph comprising the only vertex with the label c. Label c is a constant of any type, or variable v, or variable value v*.

The unary formula (see Figure 2) is a graph comprising the initial vertex and the arc with the label T (term), going out of the initial vertex and coming into the initial vertex of a logical formula F.

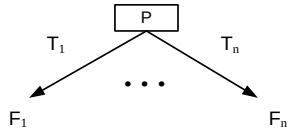


Figure 3. The propositional formula

The propositional formula (see Figure 3) is a graph comprising the initial vertex with the propositional label P and n arcs (two and more) going out of this vertex. Each of these arcs has the label Ti (term; i varies from 1 to n, and all of these arcs must be different) and coming into the initial vertex of a logical formula Fi. The propositional labels P are: & — conjunction, ∨ — disjunction, and | — XOR. The set of propositional labels is extended.

The simple quantifier formula (see Figure 4) is a graph comprising the only vertex with the label QMT, where Q is a quantifier, M is a set, and T is a term. The quantifier is ∀ (universal), ∃ (existential), ∃2 (existential not less than two), ∃? (existential but not for all), ∃! (existential and only), ∃ [] (existential subinterval).

The set is defined by values of the elements (constants), or the type of valid values, or an integer and a real interval, or a variable. The type is "string", "integer", "real", "integer interval ", "real interval ", and "data-time ". The set of types is extended. The determined set and the chosen quantifier define conditions and contingencies during the consequence process of a result fragment.

Figure 4. The simple quantifier formula

The structural quantifier formula (see Figure 5) is a graph comprising the only vertex with the label QMT, where Q is a quantifier, M is a set, and T is a term, and a logical formula F, the initial vertex of which is inside the initial vertex of the structural quantifier formula.

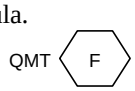


Figure 5. The structural quantifier formula

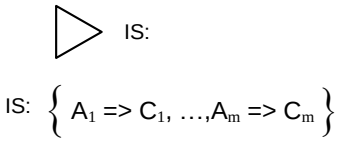


Figure 6. The set of implications

The set of implications (see Figure 6) is a graph comprising the only vertex with the label {A1 ⇒ C1, …, Am ⇒ Cm}, where A1, …, Am are antecedents and C1, …, Cm are consequents of implications. The antecedent of the implication is a finite set of components. They are logical formulas; each of them can have a prefix. The prefix is a name of an application written in the OPL. The consequent of the implication is a logical formula.

Variables declared in logical formulas can be only in antecedents and consequent of implications. The special labels of vertexes of unary formulas can be only in antecedents of implications. If a variable is in the consequent of the implication, it must be in antecedent of the implication.

For realization of complicated calculations computed predicates are added to the OPL. They are intended for describing calculations using operators of an imperative language (for example, Java). A computed predicate has name and a set of formal parameters.

Abstract interface commands define functional of a user interface. They are divided into four classes: output commands, input (edit) commands of a value of a type, input (edit) commands of a set of values of a type, choice commands of a subset from the set of values.

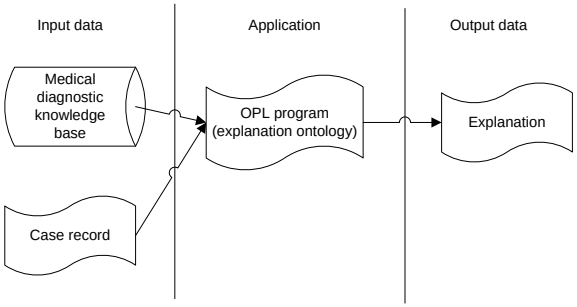


Figure 7. Architecture of the expert system of the medical diagnosis

Using the OPL the expert system of medical diagnosis is described (see Figure 7). Input data for the system are the knowledge base and a case record. The application (the program in OPL) is the semantic network of explanation (forming the explanation). The explanation consists of two parts: hypotheses explanation that a patient is healthy and hypotheses explanation that a patient suffers from a disease from the knowledge base (see Figure 8).

 ISBN: 978-1-61208-217-2

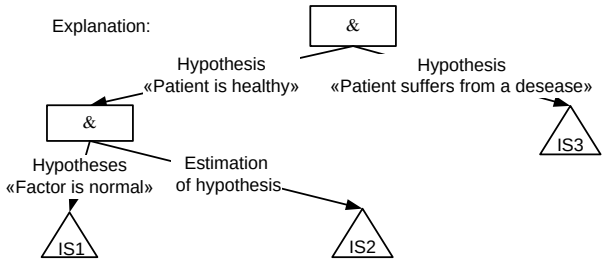


Figure 8. Explanation of the expert system of the medical diagnosis

Hypotheses explanation that a patient is healthy consists of hypotheses explanation that all observed factors of the patient is normal and hypotheses estimation that the patient is healthy.

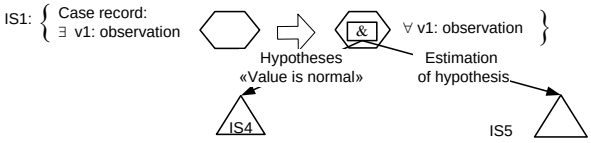


Figure 9. Description of "Factor is normal" hypothesis

Implication IS1 (see Figure 9) forms variable value v1. It is a set of all observed factors in the case record. Hypotheses explanation consists of two parts. The first part is hypotheses explanation that all observed values of this factor is normal. The second part is hypotheses estimation that this factor is normal. Similarly to IS1 implications IS2, IS3, IS4, and IS5 are described.

VI. CONCLUSION

The new paradigm is intended for reducing labor-intensiveness of development and maintenance of intelligent systems. The main idea is to describe an ontology of results using the visual logical language of programming. The programmer does not have to describe a process of obtaining result; it is the function of the language processor. All data in the ontological paradigm are semantic networks. The language has facilities for user interface realization and for including imperative structures.

ACKNOWLEDGMENT

The research was supported by the Russian Foundation for Basic Research, the project 12-07-00179-a and the Far Eastern Branch of Russian Academy of Science, the project 12-I-ОНИТ-04

REFERENCES

[1] J. Lloyd Practical advantages of declarative programming. In: Proceedings of the 1994 Joint Conference on Declarative Programming, GULP-PRODE'94, Springer Verlag, 1994, Vol. 94, pp. 1-15.

[2] I. Bratko Prolog programming for artificial intelligence. Harlow, England ; New York: Addison Wesley, 2001, 678 p.

[3] C. Orlov Technology of software development – SPb: Piter, 2002, 464 p. (in rus.)

[4] UspenskiyV.A. Semenov A.L. The theory of algorithms. Fundamental discoveries and applications - M.: Nauka, 1987, 288 p.(in rus.)

[5] Sebesta R.W Concepts of Programming Languages. - AW: 2001, 672 p.

[6] Floyd R.W. The Paradigms of Programming// Communications of the ACM, 1979, 22(8), pp. 455-460.

 ISBN: 978-1-61208-217-2

JION: A JavaSpaces Implementation for Opportunistic Networks

Abdulkader Benchi, Pascale Launay, Frédéric Guidec
IRISA, Université de Bretagne-Sud
Vannes, France
{abdulkader.benchi, pascale.launay, frederic.guidec}@univ-ubs.fr

***Abstract*—Disconnected mobile ad hoc networks (or D-MANETs) are partially or intermittently connected wireless networks, in which continuous end-to-end connectivity between mobile nodes is not guaranteed. The ability to self-form and self-manage brings great opportunities for D-MANETs, but developing distributed applications capable of running in such networks remains a major challenge. A middleware system is thus needed between network level and application level in order to ease application development, and help developers take advantage of D-MANETs' unique features. In this paper, we introduce JION (JavaSpaces Implementation for Opportunistic Networks), a coordination middleware specifically designed for D-MANETs, and with which pre-existing or new JavaSpaces-based applications can be easily deployed in such networks.**

***Index Terms*—peer-to-peer computing; opportunistic networking; D-MANETs; coordination middleware; JavaSpaces.**

I. INTRODUCTION

A mobile ad hoc network (or MANET) is a dynamic wireless network that requires no fixed infrastructure. It is generally formed on-the-fly by a collection of wireless nodes without the aid of any centralized administration. Each mobile host can communicate with its neighbors using direct pair-wise wireless links. Communications in MANETs have been enhanced over the years thanks to multi-hop forwarding protocols, such as OLSR, AODV, DYMO, DSR, etc. [1].

Yet most of these protocols rely on the assumption that the whole MANET remains continuously connected, i.e., between any pair of hosts in the MANET, there actually exists at least one temporaneous end-to-end path. Unfortunately, this assumption does not hold in real conditions; many real MANETs are, under the most favorable conditions, only partially or intermittently connected.

The sparsely or irregular distribution of a MANET's hosts can, for example, induce link disruptions in the whole MANET. These disruptions may in turn split the whole MANET into a collection of distinct, continuously changing, disconnected "islands" (connected components) as shown in Figure 1. This kind of MANET is called a Disconnected MANET (D-MANET). The "*store, carry and forward*" approach is the foundation of Delay/Disruption Tolerant Networking (DTN) [2]. In a D-MANET, it can help bridge the gap between non-connected parts of the network; the mobility of hosts makes it possible for messages to propagate network-wide by using mobile hosts as carriers (or *data mules*) that can move between network islands. As shown in Figure 1, connectivity disruptions between islands 1 and 2 can for example be tolerated thanks to users moving (deliberately or by chance) between these islands. The device of a user moving from island 1 to island 2 acts as a data mule for messages addressed to hosts located in island 2. Considering that many carriers may be involved successively for the transmission of a single message, this approach provides message delivery at the price of additional transmission delays. Figure 1 shows that the transmission of a message from island 1 to island 4 can for example involve two message carriers: first a carrier moving from island 1 to island 2, and then another carrier –or the same one– moving from island 2 to island 4.

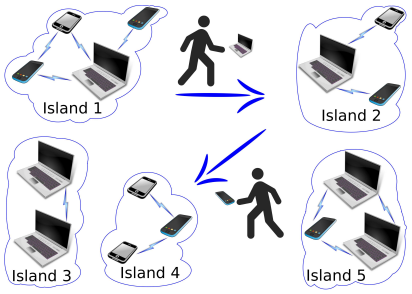


Figure 1. Example of a disconnected mobile ad hoc network

In the DTN community, some approaches make the assumption that communications between the hosts can be predicted accurately, and routing strategies can be thus devised based on contact predictions. But, in most real D-MANETs, communications are not planned in advance and can hardly be predicted, especially if the physical carriers are for example human beings carrying laptops or smartphones. The term opportunistic networking is often used to denote such disruption tolerant networks where the contacts must be exploited opportunistically [3]. In such wise, each contact represents an opportunity for two hosts to exchange messages. Consequently, communication protocols for D-MANETs usually provide no more than best-effort delivery. Consider again the example shown in Figure 1, and assume a message is addressed by a host in island 2 to a host in island 3. If no human carrier ever visits island 3, then there is no chance that the message ever gets delivered in this island.

The dynamic nature of D-MANETs creates many challenges for application developers. As a general rule, no host can be considered as stable and accessible enough to play the role

 ISBN: 978-1-61208-217-2

of a server for all the other hosts. Consequently, applications developers should generally use a peer-to-peer model rather than a client-server one. Developers, while writing their applications, should additionally take into consideration occasional transmission failures and long transmission delays.

All these reasons result in an increasing need for a middleware system that, while coping with D-MANETs issues, provides the developers with a set of APIs that eases the development of distributed applications over D-MANETs. Moreover, any middleware system for D-MANETs must have an asynchronous nature in order to fit with the long transmission delays observed in such networks.

In the remainder of this paper, we present JION (JavaSpaces Implementation for Opportunistic Networks), a coordination middleware system we designed and implemented specifically for D-MANETs. JION is actually an implementation of the JavaSpaces specification [4], so any pre-existing distributed application basing on the JavaSpaces API can be executed in a D-MANET using JION, with no further development.

This paper is structured in the following way: the JavaSpaces specification is described briefly in Section II. Section III presents the JION's architecture along with details about its implementation. Evaluation results are shown in Section IV. Section V discusses related work. Section VI concludes this paper and describes our plans for future work.

II. JavaSpaces Background

The JavaSpaces technology, implemented by JION, is a Java specification of the concept of tuple space, which was originally introduced in the Linda programming language [5]. In this section, we provide a brief introduction to the tuple space as introduced in Linda. The JavaSpaces technology is presented as well.

A. Tuple space

The tuple space concept has its root in the Linda parallel programming language developed at Yale University [5]. A tuple space is a shared data space acting as an associative memory used by several processes for communication and/or coordination requirements. A Linda application is viewed as a collection of processes cooperating via the flow of data structures, called "*tuples*", into and out of a *tuple space*. Each tuple is a record of typed fields containing the information to be communicated. The coordination primitives provided by Linda allow processes to insert a tuple into the tuple space (*out*) or retrieve tuples from the tuple space, either removing those tuples (*in*) or preserving the tuples in the space (*read*). For retrieving operations, the tuples are selected using a simple pattern matching from a given set of parameters.

B. JavaSpaces

The JavaSpaces technology is a Java specification of the tuple space concept, implemented inside the JINI architecture. It defines a set of application programming interfaces (APIs) that extend the simple core of Linda primitives. The JavaSpaces version of Linda tuples, called "entries", are Java objects that contain public fields that act as Linda's typed fields. JavaSpaces provides *read*, *take* and *write* operations in order to implement Linda's *read*, *in*, and *out* operations respectively. Additionally, it provides a *notify* operation that allows processes to perform a lookup operation in an asynchronous manner. This operation notifies the processes by sending a special object called *event* containing information, to which the processes react. The matching in read, take and notify operations are performed using a special kind of entry, called a *template*, that characterizes the kind of entries the process wants to look for. The selected entries are those whose types and fields match the template. JavaSpaces also provides light versions of *read* and *take* operations, with which processes do not need to wait for the answer. These operations, called *readIfExist* and *takeIfExist*, can be useful when a process requires an answer without blocking. As JavaSpaces' entries are passive data, processes cannot perform operations on tuples directly. In order to modify an entry, a process must explicitly remove, update and reinsert it into the space. It is worth taking into consideration that till now there is no standard JINI security model to be used by JavaSpaces users.

III. JavaSpaces Implementation For Opportunistic Networks (JION)

The JavaSpaces technology was primarily designed to provide persistent object exchange areas (spaces), through which processes coordinate actions and exchange data. Most of the JavaSpaces implementations are server-based systems where centralized servers are used to manage such spaces. As explained in Section I, a server-based system is hardly compatible with the characteristics of D-MANETs, as no host in a D-MANET can act as a reliable server for all the other hosts. A server-less JavaSpaces implementation must then be developed in order to provide JavaSpaces services for D-MANETs.

JION, or JavaSpaces Implementation for Opportunistic Networks, is a JavaSpaces implementation that was designed along that line. Its architecture is composed of two basic modules: the communication middleware system, and the JavaSpaces services system.

A. Communication Middleware

Building any application for D-MANETs requires some communication middleware system, with which hosts can collaborate in a peer-to-peer manner to ensure message transportation and deal with high latency and high failure rate. JION relies on a communication middleware system called DoDWAN *(Document Dissemination in mobile Wireless Ad hoc Networks)* [6]. DoDWAN has been designed in our laboratory, and it is now distributed under the terms of the GNU General Public License[1].

DoDWAN supports content-based information dissemination in D-MANETs. In content-based networking, information flows towards interested receivers rather than towards specifically set destinations. This approach notably fits the needs of

[1] http://www-irisa.univ-ubs.fr/CASA/DoDWAN

 ISBN: 978-1-61208-217-2

Board= "Sports" *Topic*= "football" *Language*= "English"	*Board*= "Sports" *Topic*= ".*ball" *Language*= "English | French | German"
(a)	(b)

Figure 2. A message descriptor and a message selector

applications and services dedicated to information sharing or event distribution. It can also be used for destination-driven message forwarding, though, considering that destination-driven forwarding is simply a particular case of content-driven forwarding where the only significant parameter for message processing is the identifier of the destination host (or user).

Messages in DoDWAN are composed of two parts: a descriptor and a payload. The payload is simply perceived as a byte array. The descriptor is a collection of attributes expressed as *(name, value)* tuples, as illustrated in Figure 2a. These attributes can be defined freely by the developers of application services built on top of DoDWAN.

DoDWAN implements a selective version of the epidemic routing model proposed in [7]. It provides application services with a publish/subscribe API. When a message is published on a host, it is simply put in the local cache maintained on this host. Afterwards, each radio contact with another host is an opportunity for the DoDWAN system to transfer a copy of the message to that host whenever it is interested.

In order to receive messages, an application service must subscribe with DoDWAN and provide a *selection pattern* that characterizes the kind of messages it would like to receive. A selection pattern is expressed just like a message descriptor, except that the *value* field of each attribute contains a regular expression, as shown in Figure 2b. The selection patterns specified by all local application services running on the same host define this host's *interest profile*. DoDWAN uses this profile to determine which messages should be exchanged whenever a radio contact is established between two hosts. Details about this interaction scheme and about how it performs in real conditions can be found in [6].

As a general rule, a mobile host that defines a specific interest profile is expected to serve as a mobile carrier for all messages that match this profile. Yet, a host can also be configured so as to serve as an *altruistic carrier* for messages that present no interest to the application services it runs locally. This behavior is optional, though, and it must be enabled explicitly by setting DoDWAN's configuration parameters accordingly.

Mobile hosts running DoDWAN only interact by exchanging control and data messages encapsulated in UDP datagrams, which can themselves be transported either in IPv4 or IPv6 packets. Large messages are segmented so that each fragment can fit in a single UDP datagram. Fragments of a large message can propagate independently in the network and be reassembled only on destination hosts.

B. JION Implementation

According to the JavaSpaces specification, processes coordinate by exchanging entries through the space using a simple set of operations. Entries and operations represent the basic JavaSpaces' elements. This section describes the JION's architecture and its entry module, along with the supported operations.

1) JION's architecture: As mentioned in Section III, JION is a server-less JavaSpaces implementation, as it is intended to be used in D-MANETs where server centralization is impractical. Each host maintains a local space, in which JION stores the entries produced locally (that is, entries produced by write operations, which have been invoked by local processes). If entries were propagated all over the D-MANET and managed in a collaborative manner, this could result in *orphan entries*; as stated in [8], it is impossible to obtain a consensus between hosts in a distributed disconnected environment. Consequently, D-MANET's hosts could not agree to remove any entry from the space, for example when a process wants to take it. Imagine that an entry has been propagated over the hosts in the islands shown in Figure 1, and a process in island 1 takes this entry. If no user ever visits island 5 for example, the copies of this entry in this island will become orphan entries. In JION, the write operations are only processed locally, while matching and fetching operations (read, take and notify) are processed by querying hosts over the network for the entries they own.

2) Entries and Templates: According to the JavaSpaces specification, an *entry* is an object reference characterized by its "*fields*". In the JavaSpaces terminology, entry fields refer only to the public fields of the entry objects. In fact, entry fields are meant to act as a set of attributes characterizing an entry, and are used to perform matching operations while retrieving entries from the space.

As mentioned before, a DoDWAN message has two parts: a descriptor and a payload. Since DoDWAN's descriptor is also meant to characterize the content of the message, JION maps the entry's fields to the DoDWAN's descriptor, as their content is needed by JION to do match operations. The rest of the entry (that is, non-public fields) is simply carried in the message as its payload, and considered as a simple byte array.

Templates are special entry objects whose fields' values are used in match operation. This notion of a template in JavaSpaces is mapped to that of DoDWAN's selection pattern, which is used by JION's pattern matching operation.

3) Operations: According to the JavaSpaces specification, access to entries must be done through a set of basic operations, which are: write, read, take and notify. Below is a description of the way JION supports these operations.

write: this operation stores a new entry into the host's local space for a specific period of time, called a lease. A lease represents the lifetime of the associated entry. As mentioned above, each entry is only stored on the local host and is not replicated over the D-MANET. Therefore, it is up to each host to monitor its local space and manage its own entries, and especially ensure that out-of-date entries are not used anymore.

read: this operation requests the JION service to locate an entry that matches the template provided as a parameter. When a process on a host performs a read operation, the host's local

 ISBN: 978-1-61208-217-2

space is queried first in order to find a matching entry. If no matching entry is found, JION disseminates the specified template over the D-MANET. Each host, when it receives this template, queries its local space to find a matching entry and forward a copy of this entry back to the requesting process. It is then up to the requesting process to choose one entry as an answer to its read request. Operation *readIfExists* is also supported by JION. This version queries only the local space to find an entry that matches the specified template. The request is not disseminated over the D-MANET.

take: this operation basically performs the same function as *read*, except that it removes the matching entry from the space. JION first searches the local space. If no match is found, it queries all the hosts over the D-MANET in order to discover which hosts (if any) have a matching entry. Upon receiving the proposals, JION selects one host, from which the entry should be taken. JION then asks the chosen host to permanently remove the matching entry from its local space and hand it back to the requesting process. The entire operation may take more time than the read operation since it needs four messages while only two messages are required in the *read* operation. JION also supports a *takeIfExists* operation, which performs exactly like the corresponding *readIfExists*, except that a matching entry is only requested from the local space.

notify: this operation notifies a process when entries that match a given template are written into a space. When a notify operation is performed by a process, JION disseminates the given template all over the hosts in the D-MANET. The hosts register the notify request in the hope that a matching entry will be written before the request's lifetime expires. Consequently, when a matching entry is written in a host, the host forwards an event object containing information about this entry and its location to the requesting process.

4) Transactions: According to the JavaSpaces specification, it is possible to group multiple operations (participants) into a bundle that acts as a single atomic operation. This is done using the optional concept of transaction. Either all operations within the transaction will be performed or none will. In fully-connected stable networks, a transaction is controlled by a specific manager (server), which should always be reachable by all the participants. If a participant is momentarily disconnected, the whole transaction is aborted. Considering that hosts in D-MANETs can neither rely on a reliable server nor reach a consensus, it is not possible to ensure transactions as defined by JavaSpaces. For this reason, JION does not support the concept of transaction and each operation is considered as a singleton operation.

IV. Evaluation

A D-MANET is a wireless network whose topology is continuously changing, and where radio contacts between mobile hosts do not necessarily follow any predictable pattern. Therefore, protocols and systems designed for D-MANETs are usually evaluated using network simulators. The originality of our work lies in the fact that JION has been fully implemented in Java and is now distributed under the terms of the GNU General Public License[2].

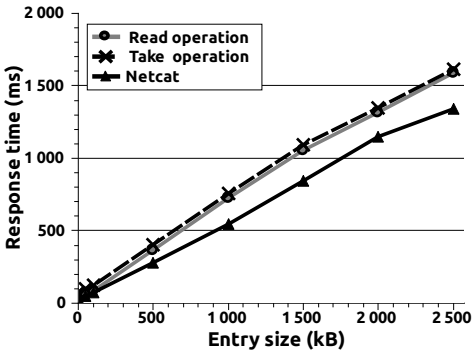


Figure 3. Response time observed between two netbooks

JION has seen extensive testing to examine how well it performs in D-MANETs. While conducting these tests we strived to evaluate how easy it is for an application developer to implement a distributed application using JION. Furthermore, we have evaluated the efficiency of JION in a real D-MANET.

A. Developing distributed applications with JION

JION implements Sun Microsystems' JavaSpaces Technology specification, provided as a part of the Java Jini Technology [4]. Since JION implements a well-known middleware specification, developers do not need to learn a new programming language, or get familiar with an exotic programming model or API. A developer can simply focus on writing a standard JavaSpaces application, and JION will take care of its execution in a D-MANET. Indeed, any pre-existing JavaSpaces application can be deployed using JION, without any change in its source code.

Developers should however be aware of the specific constraints posed by D-MANETs, where message delivery, message ordering, and transmission delays are usually not guaranteed. Such constraints are not due to limitations in JION; they are due to the very nature of D-MANETs. As explained in Section I, opportunistic protocols and middleware systems designed for D-MANETs can do no magic; they can support network-wide communication in a D-MANET, using mobile hosts as carriers that help to bridge the gap between non-connected parts of the network. Yet, unless otherwise specified they do not control how mobile hosts move in the network, so they cannot guarantee that a message will ever reach (or reach in time) any particular host in the network. A developer working on an application for D-MANETs should therefore assume that delivery failures and late deliveries may be more common than in-time deliveries, and design the distributed application or organize its deployment accordingly.

For testing and evaluation purposes, we have developed a distributed bulletin board system (D-BBS) inspired from the classical bulletin board system (BBS). A BBS typically consists of a number of bulletin boards, which serve as discussion areas relating to general themes. Each bulletin board is generally labeled by an expressive name describing

[2]http://www-irisa.univ-ubs.fr/CASA/JION

 ISBN: 978-1-61208-217-2

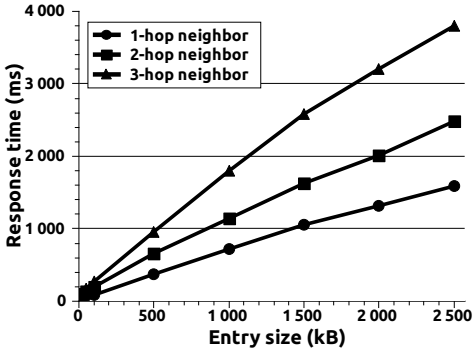


Figure 4. Response time between multi-hop neighbors

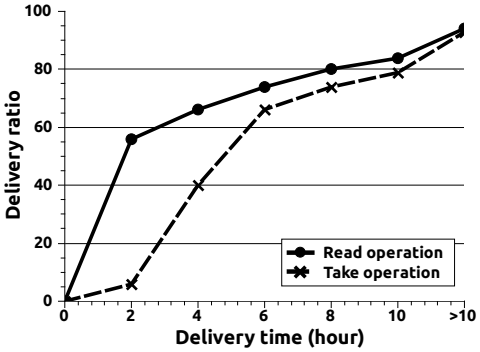


Figure 5. Entry delivery in a real D-MANET

its contents (e.g., Sports). On each board, users can post, read and delete messages under different topics. Since BBS typical implementations are usually server-based, they are not well adapted to D-MANETs, so we developed this application using the JION middleware.

In D-BBS, posting a message on a board under a topic is implemented by creating a tuple having the board's and the topic's names as fields, and writing it into the JION's space. Similarly, reading/deleting a message from a specific board under a specific topic is performed by creating a template having the board's and topic's name as fields. This template can then be passed as a parameter to JION's *read/take* function. Furthermore, the developer can get benefit from JION's *notify* function in order to keep users up to date with changes on a specific board under a specific topic using an appropriate template.

Developing and deploying this application for a D-MANET was a straightforward task. Using JION, the programmer focuses on the application features without paying attention to the specific issues of this kind of very challenging environment.

B. JION's efficiency over D-MANETs

Before trying to observe how entries can propagate between several islands in a disconnected network, one can first try to measure how fast they can propagate within a single island. Since JION is implemented on top of the DoDWAN communication system, which itself relies on UDP transmissions, our first objective was to evaluate how our multi-layer middleware architecture performs over the underlying wireless transmission medium.

We first used two netbooks A and B, running JION over a Linux operating system. These netbooks were installed next to each other in the same room, and their built-in Wi-Fi 802.11bg chipsets were configured to operate in ad hoc mode. We actually focused on the response time, which is here defined as the time interval between the time netbook A writes entries of different sizes and the time when netbook B receives them using the read/take operations. In order to get reference values regarding the capacity of the wireless link at application-level, we used the basic Netcat (nc) networking utility, that can read and write chunks of data across network connections. 200 series of tests were conducted in this scenario. The results of these tests are presented in Figure 3. Read and take operations show similar performance. However, JION shows about 20% overhead over Netcat. Considering that JION's communication middleware (DoDWAN) implements a sophisticated opportunistic protocol in order to orchestrate communications between neighbor hosts, we consider that these results are quite reasonable.

To increase realism, another real world test was conducted to investigate the behavior of JION when messages can propagate over multiple hops. The tests was carried out with four netbooks (A, B, C and D) distributed in our laboratory. Because of the effect of concrete walls on signal attenuation, the connectivity between these netbooks was such that netbook B could only communicate with A and C, while netbook D could only communicate with C. This test relied on write/read operations: a total amount of 225 entries were written on B, C, D, and host A was configured to read these entries. We measured the average time required for these entries to reach host A. The results of this test are shown in Figure 4. It can be observed that the delay before host A gets an entry changes with the size of the entry and the number of transmission hops. This is because when host B serves as a relay between its neighbors A and C, the radio channel around B is twice as busy as when B interacts only with host A. The same observation applies for host C when it must serve as a relay between hosts B and D. It must also be considered that DoDWAN strives to ensure a high delivery ratio, so entries are retransmitted again and again if they are not received in the first place. Indeed, during the test all entries got received by host A.

After measuring how JION can perform in a single, connected island we used our D-BBS application to observe how it can perform in a real D-MANET. A dozen of volunteers in our laboratory were equipped with netbooks running D-BBS. Several one-day tests were conducted by asking the volunteers to carry their netbook whenever possible –and use D-BBS services of course– during a few days while roaming the laboratory building or its surroundings. To analyze the results special attention was paid to the cumulative delivery rates of *read/take* operations, as shown in Figure 5. The cumulative delivery rates were measured in terms of time slices where a measure of 2 hours means that the average delivery time observed was between 0 and 2 hours; a measure of 4 hours

 ISBN: 978-1-61208-217-2

means it was between 0 and 4 hours, etc. Using the *read* operation, it can be noticed that nearly 59% of the entries were delivered in less than 2 hours. However, the majority of hosts had to wait between 4 and 6 hours in order to get the required entries using the *take* operation. This difference was expected since operation *take* requires more rounds than operation *read*. In general, the results show that most of the entries got delivered to their destination(s) in less than 10 hours. Yet, about 6% of the entries could not be delivered. This is the consequence of the unpredictable behavior of the users, which sometimes moved away from the laboratory or switched their netbook off unexpectedly.

V. Related Work

In the last few years, several projects revisited Linda [5], especially in the context of mobile ad hoc environments.

Both Ara [9] and LIME [10] are coordination middleware systems implementing tuple spaces stored on hosts acting as servers. These middleware systems target mobile ad hoc networks. They provide JavaSpaces services to mobile hosts that are in the servers' communication range. Since they rely on a server-based model, they are hardly usable in real D-MANETs. Limone [11] is a lightweight alternative to LIME requiring far less overhead. Limone is based on the premise that a single round-trip message exchange is always possible, making it impractical over D-MANETs, for in D-MANEs unpredictable disruptions are the norm rather than the exception. In contrast, CAST [12] is a server-less coordination middleware for MANETs. Since it does not rely on any centralized service, this middleware suits well the dynamics of wireless open networks. CAST makes it possible to process operations even when no end-to-end route exists between the involved hosts, by implementing a source routing algorithm. This routing strategy relies on the assumption that each host's motion profile is known. This is clearly a serious constraint, which limits the usability of CAST over the kind of D-MANETs JION targets, where hosts' motions are neither planned nor predictable. Tuple board (TB) [13] is another server-less coordination middleware for developing collaborative applications running in ad hoc networks of mobile computing devices. Like JION, this middleware has been fully implemented and distributed. It can thus be used and tested in real conditions. However, the proposal lacks flexibility, in that it is limited to a group of nearby connected devices: when a device leaves the network or turns off, all the tuples posted from this device are withdrawn. The importance that we attribute to disconnections make the disconnection tolerance a vital requirement for any middleware that is meant to support D-MANETs.

Furthermore, all the middleware systems mentioned above define their own communication protocol for route discovery and maintenance. Our work is different as JION presents a two-layer architecture: the upper layer is concerned with tuple space services, while the lower layer supports opportunistic communication. As mentioned above, DoDWAN has been chosen among a few opportunistic communication protocols that are openly distributed to support communications on D-MANETs. Yet, JION could theoretically be implemented above any other communication system, such as DTN2 (a reference implementation of protocols designed by the Delay-Tolerant Networking Research Group (DTNRG) [14]) or Haggle (a content-centric architecture for opportunistic communication among mobile devices [15]).

VI. Conclusion and Future Work

In this paper, we have introduced JION (JavaSpaces Implementation for Opportunistic Networks), a JavaSpaces implementation we designed and implemented specifically for disconnected mobile ad hoc networks (D-MANETs). Using JION, distributed applications based on the concept of tuple spaces (as defined in the JavaSpaces specification) can be deployed and executed in D-MANETs. JION provides an effective base, which eases the development of D-MANETs' applications. It has been tested in real conditions and is now distributed under the terms of the GNU General Public License.

Further tests are still under way in order to verify how stable JION is in different kinds of challenged environments. Future work should include the assessment of JION's portability (most likely by implementing it over the DTN2 reference implementation), and an investigation of security issues pertaining to the execution of JION-based distributed applications in D-MANETs.

References

[1] C. Liu and J. Kaiser, "A survey of mobile ad hoc network routing protocols," University of Magdeburg, Tech. Rep., 2005.

[2] K. Fall, "A delay-tolerant network architecture for challenged internets," in *ACM Annual Conference of the Special Interest Group on Data Communication*, Aug. 2003.

[3] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: Data forwarding in disconnected mobile ad hoc networks," *IEEE Communications Magazine*, Nov. 2006.

[4] E. Freeman, S. Hupfer, and K. Arnold, *JavaSpaces(TM) Principles, Patterns, and Practice*. Prentice Hall, Jun. 1999.

[5] N. Carriero and D. Gelernter, "Linda in context," *Commun. ACM*, vol. 32, no. 4, pp. 444–458, Apr. 1989.

[6] J. Haillot and F. Guidec, "A protocol for content-based communication in disconnected mobile ad hoc networks," *Journal of Mobile Information Systems*, vol. 6, no. 2, pp. 123–154, 2010.

[7] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Duke University, Tech. Rep., Apr. 2000.

[8] M. J. Fischer, N. A. Lynch, and M. S. Paterson, "Impossibility of distributed consensus with one faulty process," *Journal of the ACM*, vol. 32, no. 2, pp. 374–382, Apr. 1985.

[9] H. Peine and T. Stolpmann, "The Architecture of the Ara Platform for Mobile Agents," in *Proceedings of the First International Workshop on Mobile Agents*. London, UK: Springer-Verlag, 1997, p. 50–61.

[10] A. L. Murphy, G. P. Picco, and G.-C. Roman, "Lime: A coordination model and middleware supporting mobility of hosts and agents," *ACM Trans. Softw. Eng. Methodol.*, vol. 15, pp. 279–328, July 2006.

[11] C.-L. Fok, G.-C. Roman, and G. Hackmann, "A lightweight coordination middleware for mobile computing," *Coordination Models and Languages*, pp. 135–151, 2004.

[12] G.-C. Roman, R. Handorean, and R. Sen, "Tuple space coordination across space and time," in *COORDINATION*, 2006, pp. 266–280.

[13] A. Kaminsky and C. Bondada, "Tuple board: A new distributed computing paradigm for mobile ad hoc networks," *First Annual Conference on Computing and Information Sciences*, pp. 5–7, 2005.

[14] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay-tolerant networking architecture," IETF RFC 4838, Apr. 2007.

[15] E. Nordström, P. Gunningberg, and C. Rohner, "A search-based network architecture for mobile devices," Department of Information Technology, Uppsala University, Tech. Rep. 2009-003, Jan. 2009.

 ISBN: 978-1-61208-217-2

Semantic Numerical Operations on Strings

Taeho Jo
School of Computer and Information Engineering
Inha University
Incheon, South Korea
tjo018@inha.ac.kr

***Abstract*—This research is concerned with the definition, the analysis, and the simulation of the numeric semantic operations on strings. The motivation of this research is the dominance of textual data over numerical data in our reality and the necessity of defining semantic operations for analyzing meanings of words. In this research, we define and simulate the three operations: 'semantic similarity', 'semantic similarity average', and 'semantic similarity variance'. This research is expected to become the basis from which semantic analysis tools or systems of words, texts or corpus, are developed as its benefit. We present the simulations of carrying out operations on strings in the real corpus: NewsPage.com.**

Keywords-Semantic Operation; Similarity Semantic Average; Similarity Semantic Variance

I. INTRODUCTION

The semantic operations are defined as the operations based on quantities indicating semantic relations among entities. The words in textual data are given as operands of the operations which were proposed in this research. As the basis of performing the operations, we use the similarity matrix which consists of semantic similarities indicating how much corresponding words are similar as each other. In this research, we define the following three operations: 'semantic similarity', 'semantic similarity average', and 'semantic similarity variance'. Each operation generates a normalized value between zero and one as its output.

Previously, we attempted to replace numerical vectors by string vectors in representing texts. The reason of the replacement is the three problems: huge dimensionality, sparse distribution, and poor transparency; they are described in detail in the literatures [1][2][3][4][5]. The replacement leads to the successful performance in text categorization and clustering. However, in order to use the string vectors more naturally and freely, we need more systematic mathematical analysis and definitions on strings. The previous research concerned with encoding of texts into string vectors will be mentioned in Section 2.

In this research, we define the three semantic operations on strings. The semantic similarity between two words indicating how much two words are similar as each other, is included as the basic operation. The SSA (Semantic Similarity Average) is proposed as the average over similarities of all possible pairs of words[5]. From the SSA, we derive SSV (Semantic Similarity Variance), as the variance over the similarities. In this research, we call the defined operations numerical semantic operations, since numerical values are generated from the operations as their outputs.

We expect the three benefits from this research. For first, the semantic operations are potentially used for developing string vector based approaches to tasks of text mining and information retrieval. For second, this research may provide the basis for developing automatic semantic analysis tool for words and texts. For third, the possibility of developing even digital computers only for text processing is available potentially. In order to take the benefits, we need to define more semantic operations and characterize them mathematically.

This article is composed of the five sections. In Section II, we explore the previous research relevant to this research. In Section III, we describe the proposed semantic operations formally and characterize them mathematically. In Section IV, the operations are simulated on the real corpus. In Section V, as the conclusion, we mention the significances and the remaining tasks of this research.

II. PREVIOUS WORKS

This section is concerned with the exploration for the previous works relevant to this research. In 2000, Jo invented a new neural network, proposing encoding documents into string vectors; it provides the motivation for doing this research [1]. The semantic relations between words are considered for doing information retrieval tasks such as ranking and term weighting. Even for doing other tasks, the semantic relations are also considered. Therefore, in this section, we will explore previous works in terms of string vector encoding and tasks involving the semantic relations between words.

This research is initiated from encoding documents into string vectors, instead of numerical vectors, for doing text mining tasks. Encoding documents so was initiated by Jo in 2000, inventing the new neural network, called NTC (Neural Text Categorizer), as a practical approach to text categorization [1]. Subsequently, in 2005, Jo and Japcowicz invented the unsupervised string vector based neural network which was called NTSO (Neural Text Self Organizer) [6]. In 2009, Jo modified the KNN and SVM into its string vector based versions where the similarity measure between string

 ISBN: 978-1-61208-217-2

vectors was based on the semantic relations between words [7]. However, in order to use string vectors more freely, we need to define more semantic operations on strings and characterize them mathematically.

The semantic relations between words are considered especially in information retrieval tasks. In 2005, Shenkel et al. implemented the search engine which was called XXL, for searching for XML documents, using the semantic similarity between words, based on ontology and an index structure [8]. In 2005, Possas et al. proposed a term weighting scheme which was called 'set based model', considering the semantic relation between term [9]. In 2008, Vechtomova and Karamuftuoglu used the semantic relation between a query and terms for ranking retrieved documents [10]. The previous works show usefulness of the semantic relation between words in the domain of information retrieval.

The semantic relation between words may be considered in other tasks as well as the information retrieval tasks. In 1994, Kiyoki et al. defines metadata of image as words for representing their semantic relations for the image retrieval [11]. In 2004, Makkonen et al. defines semantic relations among words using ontology for doing the topic tracking and detection [12]. In 2007, Na et al. used the semantic relation between a query and terms for adjusting clustering results [13]. The previous works show that the semantic relation may be considered in various tasks.

This research is intended to define various semantic relations between words, assuming that each string has its own meaning. In the previous works, the semantic relations have been considered not mathematically but informally or implicitly. In other words, the mathematical foundations are not founded, yet; the computation of the semantic similarity has depended on very heuristic computations. Even if the modification and creation of string vector based approaches in favor of text categorization and clustering was successful, it was limited to process string vectors because of no more systematic mathematical foundations. Therefore, the goal of this research is to define more semantic operations on strings and characterize them algebraically, in order to overcome the limitation.

III. Numerical Semantic Operations

This section describes the semantic operations in detail and consists of the four sections. In Section III-A, we describe the similarity matrix as the basis of carrying out the semantic operations. In Section III-B, we mention the two opposite operations: semantic similarity and semantic distance. In Section III-C, we define the SSA formally and characterize it mathematically. Section III-D covers the SSV like the SSA.

A. Similarity Matrix

Before entering the semantic operations on strings, we will describe the similarity matrix in this section. The similarity matrix is used as the basis for performing the semantic operations on strings. In the similarity matrix, each of its rows and columns corresponds to a string. The matrix has the two properties: its elements are symmetry and its diagonal elements are 1s. The similarity matrix defines the semantic similarity of each of all possible pairs of strings, and it assumes that the matrix is always given before doing the operations on strings.

The similarity matrix refers to the square matrix which defines the semantic similarity of each of all possible pairs of strings, and it is denoted as follows:

$$\begin{pmatrix} s_{11} & s_{12} & \cdots & s_{1N} \\ s_{21} & s_{22} & \cdots & s_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ s_{N1} & s_{N2} & \cdots & s_{NN} \end{pmatrix}$$

The similarity matrix is given as the N by N matrix, and N indicates the total number of strings. Each of the columns and the rows corresponds to its unique string; both the ith row and the ith column correspond to the identical string. The element of the similarity matrix, s_{ij} indicates the semantic similarity between the string corresponding to the ith column and that corresponding to the jth row. Following the two properties, the similarity matrix may be built manually or automatically.

The first property of the similarity matrix is that its elements are symmetrical to each other. In other words, the rule $s_{ij} = s_{ji}$ applies to all elements in the similarity matrix. We already mentioned that the string corresponding the ith column is identical to that corresponding to the ith row. The two strings which correspond to the ith column and the jth column is same to those which correspond to the vice versa. The commutative raw is applicable to the semantic similarity between two strings.

The second property of the similarity matrix is that its diagonal elements are always given 1.0. In other words, s_{ii} is given as 1.0 as the maximum similarity. Every element in the similarity matrix is given as a normalized value between 0 and 1. The value, 1.0, signifies the maximum similarity between two strings. In the context of this research, it is assumed that the two identical strings have their maximal similarity.

The similarity matrix may be constructed, manually or automatically. A finite set of strings and the size of the similarity matrix are decided in advance. The 1.0 values are absolutely assigned to the diagonal elements of the similarity matrix. Keeping its symmetry property, normalized values between 0 and 1 are assigned to the off-diagonal elements. In other literatures, the process of building the similarity matrix from a corpus is mentioned; refer to the literatures for the detail description of the automatic construction.

 ISBN: 978-1-61208-217-2

B. Semantic Similarity and Distance

This subsection is concerned with the two base semantic operations on strings. One operation covered in this section is for evaluating how much two strings are similar based on their meanings. The other is for doing how much they are different from each other with respect to their meanings. The commutative law is applicable to both operations; the result is identical to the different order of the input strings. Therefore, in this subsection, we will describe the both operations with respect to their definition and properties.

The first base operation is for evaluating a semantic similarity between two strings. We already described the similarity matrix in Section 1, as the basis of these operations. It is possible to construct automatically the similarity matrix from a corpus, but the detail process is not covered in this article. As shown in Figure 1, the semantic similarity is carried out by retrieving directly the corresponding element from the similarity matrix as follows:

$$sim(str_i, str_j) = s_{ij}$$

This operation becomes the fundamental one for deriving more advanced operations, later.

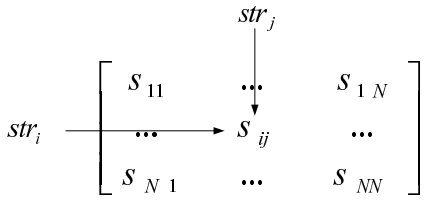


Figure 1. The Process of Retrieving the Semantic Similarity from the Similarity Matrix

The second operation is the semantic distance which is opposed to the previous operation. Like the semantic similarity, this operation generates a normalized value between 0 and 1 as the output. The semantic distance between two strings is computed by subtracting the semantic similarity from 1.0 as follows:

$$dis(str_i, str_j) = 1.0 - sim(str_i, str_j) = 1.0 - s_{ij}$$

The value generated from the semantic distance is the 1.0's complement of the semantic similarity. We may build the semantic distance matrix by subtracting each element from 1.0 as follows:

$$\begin{pmatrix} 1.0 - s_{11} & 1.0 - s_{12} & \dots & 1.0 - s_{1N} \\ 1.0 - s_{21} & 1.0 - s_{22} & \dots & 1.0 - s_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ 1.0 - s_{N1} & 1.0 - s_{N2} & \dots & 1.0 - s_{NN} \end{pmatrix}$$

Both operations are characterized as the fact that the commutative law is applicable. In the case of the semantic similarity, the commutative law applies because the similarity matrix is symmetry, as follows:

$$sim(str_i, str_j) = s_{ij} = s_{ji} = sim(str_j, str_i)$$

The commutative law also applies because the same value is subtracted from 1.0 as follows:

$$dis(str_i, str_j) = 1.0 - s_{ij} = 1.0 - s_{ji} = dis(str_j, str_i)$$

The similarity distance matrix becomes symmetry, but its diagonal elements are 0 values instead of 1.0 values. If the similarity distance matrix is given, the semantic distance is carried out by retrieving the corresponding element from the matrix.

C. Semantic Similarity Mean

This subsection is concerned with the first n-ary semantic operation on strings. The n-ary semantic operation refers to the class of semantic operations which takes an arbitrary number of strings as the input. In this operation, all possible pairs of strings are generated and the semantic similarity to each pair is computed. The semantic similarity mean of the strings is computed by averaging the similarities of the all possible pairs. In this subsection, we will describe the operation with respect to the definition, the properties, the procedure, and the utility.

This operation is denoted as follows:

$$avgsim(str_1, str_2, \dots, str_n) = \frac{2}{n(n-1)} \sum_{i<j} sim(str_i, str_j)$$

When n strings are given as the input, we generate $n(n-1)/2$ pairs of strings as all possible ones. For each pair, we may compute the similarity by retrieving it from the similarity matrix as shown in Figure 1. We obtain the average semantic similarity by summing the similarities of all pairs and dividing the sum by the number of all possible pairs, $n(n-1)/2$. The average semantic similarity signifies the semantic cohesion of the group of strings.

The properties of this operation are as follows:

- If all strings are identical, the average semantic similarity is given as 1.0 values, since the diagonal elements of the similarity matrix are given 1.0.
- $\frac{2}{n(n-1)} \sum_{i<j} sim(str_i, str_j) = \frac{2}{n(n-1)} \sum_{i>j} sim(str_i, str_j)$, since the similarity matrix is symmetry one.
- If all pairs of the strings are complementary (lowest similarity), the average semantic similarity becomes the minimum.
- The average semantic similarity is always given as a normalized value, since the similarities of all possible pairs are given as normalized values.

This operation takes an arbitrary number of strings as the input. Among the strings, all possible pairs are generated; if the number of strings is n, $n(n-1)/2$ pairs are generated.

 ISBN: 978-1-61208-217-2

For each pair, its similarity is retrieved from the given similarity matrix. The average semantic similarity is obtained by summing the similarities of the all possible pairs and dividing the sum by the number of pairs. Therefore, the averaged semantic similarity which is given as a normalized value is the output of the operation.

Figure 2 illustrates the two different groups of strings. The left group in Figure 2 contains the strings within the domain of computer science. The right group in Figure 2 contains the strings spanning over various domains. Intuitionally, the left group of strings has the higher average semantic similarity than the right group. Through the example illustrated in Figure 2, this operation may be used for estimating the cohesion of groups of strings.

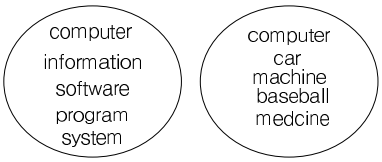


Figure 2. Two Groups of Words: One in a specific domain and the other in various domains

D. Semantic Similarity Variance

This subsection is concerned with the second n-ary semantic operation on strings. Under an identical average semantic similarity, there exist different distributions of similarities of pairs of strings. The pairs of strings may concentrate on the average semantic similarity, or they may disperse from it. We need the measure how much the similarities of the pairs concentrate on the average semantic similarity. In this subsection, we describe the operation in detail with respect to its definition, properties, and procedure.

The operation, called semantic similarity variance, is denoted as follows:

$$var(str_1, \ldots, str_n) = \frac{2}{n(n-1)} \sum_{i<j} (sim(str_i, str_j) - avgsim(str_1, \ldots, str_n))^2$$

If n strings is given as the input, the number of all possible pairs becomes $n(n-1)/2$. Before performing this operation, the average semantic similarity should be computed by the operation which was mentioned in Section III-C. This operation focuses on the individual square of difference between a similarity of each pair and the average semantic similarity. This operation corresponds to the variance in the context of statistics.

The properties of this operation are as follows:

- $\frac{2}{n(n-1)} \sum_{i<j} (sim(str_i, str_j) - avgsim(str_1, \ldots, str_n))^2 = \frac{2}{n(n-1)} \sum_{i>j} (sim(str_i, str_j) - avgsim(str_1, \ldots, str_n))^2$

  It means that swapping indexes of elements does not influence on computing the average semantic variance, since the similarity matrix is symmetric as follows:
- $sd(str_1, \ldots, str_n) = \sqrt{var(str_1, \ldots, str_n)}$

$sd(str_1, \ldots, str_n)$ is called the semantic similarity standard deviation.

In this operation, an arbitrary number of strings is given as the input. Using the operation which was mentioned in Section III-C, the semantic similarity average is computed. For each pair, the difference square between its similarity and the average semantic similarity is computed. The difference squares are averaged into the semantic similarity variance. The square root of the semantic similarity variance becomes the semantic similarity standard deviation. Whether it is the variance or standard deviation, the value is always given as a normalized value.

The operation may be used for judging whether words are distributed, randomly or not. Let us consider the two groups of words with their identical semantic similarity. One group whose semantic similarities are concentrated on the average semantic similarity has very small the semantic similarity variance. However, the other whose semantic similarities are dispersed very much has the larger semantic similarity variance. In this case, the latter group is judged as the random distribution of words.

IV. SIMULATIONS

This section is concerned with the set of simulations of carry out the semantic operations on strings. We used the collection of news articles called 'NewsPage.com' in this research as the source from which the similarity matrix is built. The similarities among words are computed automatically based on the number of texts where the words are collocated with each other. We selected words from the corpus at random and we applied the semantic operations to them. In this section, we present and discuss the simulation results from applying the semantic operations.

We illustrate the specification of the collection of news articles called 'NewsPage.com' in Table I. The collection was constructed by copying and pasting individual news articles provided by the web site, 'newspage.com' as plain texts files. The five categories were predefined and 1,000 articles are available in the collection. Previously, it had been used as the test data for evaluating the approaches to text categorization. However, in this research, we use it as the source from which the similarity matrix is built.

This set of simulations is carried out with three steps: indexing the corpus, constructing the similarity matrix, and carrying out the semantic operations on strings. The corpus, which is the collection of texts, is indexed into a list of words and their frequencies as shown in Figure 3. We selected 100 words randomly and built the 100 X 100 similarity matrix by computing semantic similarities among words based on the number of texts where the words collocates with each

 ISBN: 978-1-61208-217-2

Table I
CATEGORIES AND NUMBER OF ARTICLES IN CORPUS: NEWSPAGE.COM

Category Name	#Articles
Business	400
Health	200
Law	100
Internet	300
Total	1000

other. We made 16 lists each of which consists of five words by selecting words randomly among the selected 100 words and applied the semantic similarity average and variance to each list. We generated values of the semantic similarity averages and variances as results of this set of simulations.

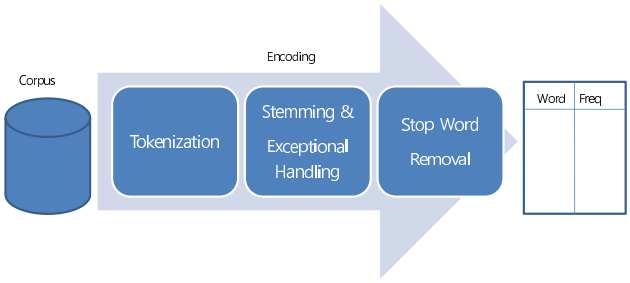


Figure 3. The Process of Indexing Corpus

In Figure 4, we illustrate the simulation results from carrying out the operations whose basis is constructed from NewsPage.com. In Figure 4, each position in the x-axis corresponds to a list of words. In the y-axis, each value indicates a normalized one between zero and one of the two operations. The gray bar and the white bar indicate values of SSA (Semantic Similarity Average) and SSV (Semantic Similarity Variance), respectively.

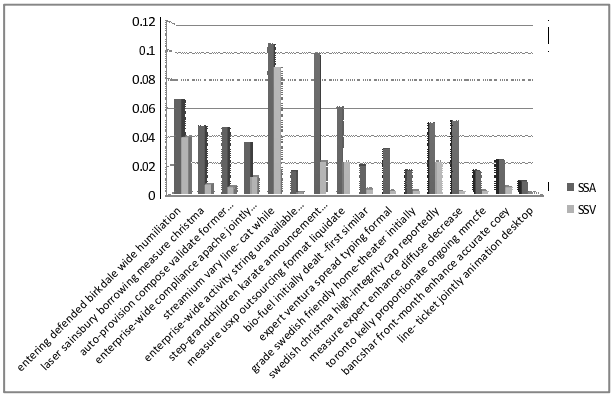


Figure 4. The Simulation Results from the SSM and SSV from the Corpus: NewsPage.com

Let us consider the results from simulating the two operations illustrated in Figure 4. The list which contains step-grandchildren, karte, announcement, and other two words, has the high SSA and the low SSV, as shown in Figure 4. The list is characterized by its values as dense semantic relations among the five words. The list with streaminum, vary, line-cat, and other two words, have large values of both SSA and SSV; it indicates that majority of words are related semantically densely, and minority are related loosely. The list with line-ticket, jointly, animation, desktop, and so on has low values of both SSA and SSV; it is characterized as loose semantic relation among them.

V. CONCLUSION AND FUTURE WORKS

Let us consider the significances of this research. From this research, we obtain the chance to measure semantic relations among words by the two simple operations: semantic similarity and semantic distance. We are able to observe the semantic cohesion of words through the operation called SSA. It gets possible to observe the distributions over semantic similarities of words, through the operation called SSV. This research provides potentially the way of developing semantic analyzer of textual data.

In spite of the above significances, let us consider remaining tasks for proceed further research. We need to make more simulations of carrying out the operations in other domains. More semantic operations will be defined and characterized mathematically. When the complexity of performing the semantic operation is high, it is necessary to reduce the complexity by developing their approximating algorithms. The operations will be applied to text processing tasks in information retrieval systems and text mining systems.

REFERENCES

[1] T. Jo, “NeuroTextCategorizer: A New Model of Neural Network for Text Categorization”, pp. 280-285, The Proceedings of ICONIP 2000, 2000.

[2] T. Jo, ”The Implementation of Dynamic Document Organization using Text Categorization and Text Clustering”, PhD Dissertation of University of Ottawa, 2006.

[3] T. Jo and D. Cho, “Index Based Approach for Text Categorization”, pp. 127-132, International Journal of Mathematics and Computers in Simulation, vol 2, no 1, 2008.

[4] T. Jo, “Topic Spotting to News Articles in 20NewsGroups with NTC”, pp. 50-56, Lecture Notes in Information Technology, vol 7, 2011.

[5] T. Jo, “Definition of String Vector based Operations for Training NTSO using Inverted Index”, pp. 57-63, Lecture Notes in Information Technology, vol 7, 2011.

[6] T. Jo and N. Japkowicz, “Text Clustering using NTSO”, pp. 558-563, The Proceedings of IJCNN, 2005.

[7] T. Jo, “Modification of Classification Algorithm in Favor of Text Categorization”, pp. 13-23, International Journal of Computer Science and Software Technology, vol 2, no 1, 2009.

 ISBN: 978-1-61208-217-2

[8] R. Schenkel, A. Theobald, and G.Weikum, “Semantic Similarity Search on Semistructured Data with the XXL Search Engine”, pp. 521-545, Information Retrieval, vol 8, no 4, 2005.

[9] B. Possas, N. Ziviani, W. Meira, Jr., and B. Ribeiro-Neto, “Set-based vector model: An efficient approach for correlation-based ranking”, pp. 397-429, ACM Transactions on Information Systems, vol. 23, no. 4, 2005.

[10] O. Vechtomova and M. Karamuftuoglu, “Lexical cohesion and term proximity in document ranking”, pp. 1485-1502, Information Processing & Management, vol 44, no 4, 2008.

[11] Y. Kiyoki, T. Kitagawa and T. Hayama, “A Metadatabase System for Semantic Image Search by a Mathematical Model of Meaning”, pp. 34- 41, ACM SIGMOD Record, vol 23, no 4, 1994.

[12] J. Makkonen, H. Ahonen-Myka, and M. Salmenkivi, “Simple Semantics in Topic Detection and Tracking”, pp. 347-368, Information Retrieval, vol 7, no 3-4, 2004.

[13] S. Na, I. Kang and J. Lee, “Adaptive document clustering based on query-based similarity”, pp. 887-901, Information Processing & Manage- ment, vol 43, no 4, 2007.

 ISBN: 978-1-61208-217-2

Normalized Table Matching Algorithm for Classifying News Articles

Taeho Jo
School of Computer and Information Engineering
Inha University
Incheon, South Korea
tjo018@inha.ac.kr

***Abstract*—In this research, we propose encoding texts into normalized tables for categorizing texts, automatically. Previously, the table based approach was proposed, but the categorical scores indicating how much the text is relevant to the given category may be overestimated or underestimated by the given text length. As the solution to the problem, in this research, we encode texts into fixed sized tables, define the operation for computing the similarity between two tables as a normalized value, and characterize it mathematically. As the benefits from this research, we are able to compute category scores independently of a given text length, consider weights from both texts, and expect the more stable performance. We validate empirically the proposed approach with respect to the performance and the stability by comparing it with the traditional approaches.**

Keywords-Text Categorization; Table based Matching Algorithm

I. INTRODUCTION

As shown in Figure 1, text categorization refers to the process of assigning one or some of predefined categories to each document. In the task, an unseen document is given as the input, and one or some of the predefined categories are generated the output. The task is regarded as an instance of pattern classification where each object is classified into its own label. For the task, a list of categories should be predefined and sample documents which are manually labeled by one or some of the categories should be prepared as its preliminary tasks. Techniques of text categorization are necessary for processing and managing efficiently textual data which are growing explosively in information systems; many state of the art approaches [1][2][5] have been developed since 1990s.

In order to use a previously developed approach for text categorization, we must encode documents into numerical vectors. Encoding them so causes the two main problems: huge dimensionality and sparse distribution. The first problem, 'huge dimensionality', refers to the phenomena where documents are encoded into too many dimensional numerical vectors for preventing information loss. In spite of using feature selection methods, documents are usually encoded into several hundred dimensional vectors. Under the problem, it takes very much cost for processing each document in terms of time and system resource, and many training examples are required proportionally to the dimension for avoiding over-fitting.

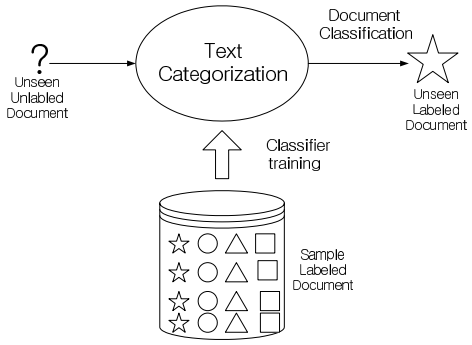


Figure 1. The Process of Indexing Corpus

The proposed version is improved over the previous one [10][11] aspects. For first, in the previous version, texts are encoded into variable sized tables, whereas, in this version, they are done into constant sized ones. For second, in the previous version, the categorical scores are computed by summing weights of tables simply, whereas in this version, they are computed by the proposed operation which is characterized mathematically. For third, in the previous version, the categorical scores are only real values, while in the current version, they are given normalized values. Therefore, in this version, we expect more stable performance as well as better performance.

We expect the three benefits from this research. For first, we overcome the overestimation and the underestimation by variable text lengths. For second, the categorical scores are given as normalized values between zero and one independently of domains; the categorical estimations are performed more stably. Compared with traditional approaches, the proposed approach is expected to have its more stable performance over corpus as well as its better performance. Together with the previous version, the proposed version also solves the main problems in encoding texts into numerical vectors.

This article consists of the five sections. In Section II, we will survey the previous research relevant to this research. In Section III, we describe the proposed version of table based matching algorithm in detail. In Section IV, we

 ISBN: 978-1-61208-217-2

validate empirically the proposed method by comparing it with the popular approaches, considering both performance and stability. In Section V, as the conclusion of this research, we mention the significances and the remaining tasks of this research.

II. Previous Works

This section is concerned with the previous research relevant to this research. Even if various kinds of approaches to text categorization are available, in this research, we count only three typical ones, KNN, Naive Bayes, and Support Vector Machine. In this section, we also survey the previous solutions to the problems in encoding texts into numerical vectors. In spite of its better performance of previous version, we will point out its demerits and mention how to improve it. Therefore, in this section, we will explore the previous research in the three directions.

Let us mention the KNN, the Naive Bayes, and the SVM as the three typical approaches to text categorization. The KNN was used for text categorization by Massand et al. and Yang in 1992 and 1999, respectively [1][2]. The Naive Bayes was used by Mladenic and Grobelink and Eyheramendy et al., in 1999 and 2003, respectively [3][4]. The SVM was used for spam mail filtering by Drucker et al. [5] and it was mentioned as typical approach to text categorization by Cristianini and Shawe-Taylor [6]. However, it requires to encode texts into numerical vectors for using one of the three approaches for the text categorization.

There were previous attempts to solve the problems in encoding texts into numerical vectors. In 2000, Jo initially encoded texts into string vectors instead of numerical vectors as the alternative representations of texts [7]. In 2002, Lodhi et al. proposed the string kernel as a kernel function in using the SVM for the text categorization [8]. In 2007, Lee and K. Kageura tried to solve the problems where many examples are required from the huge dimensionality by generating the virtual documents [9]. The trials show that the problems in encoding texts into numerical vectors were realized.

We started to encode texts into tables instead of numerical vectors and string vectors. In 2008, Jo and Cho created initially the table based matching algorithm as the approach to the text categorization [10]. In 2008, Jo applied it to soft text categorization where more than one category may be assigned to each text [11]. In 2008, Jo proposed the table based approach to the text clustering as well as the text categorization [12]. The previous version of the table based algorithm solved the problems in encoding texts into numerical vectors, but it has its own demerit where the categorical scores are overestimated or underestimated by variable sized texts.

We need to consider the demerits of the previous version, even if it was applied successfully to text categorization. Even more, the string kernel proposed by Lodhi et al. failed to improve the text categorization performance. It is not easy to implement the text categorization algorithms where texts are encoded into string vectors, because operations on string vectors are not defined systematically, mathematically, and theoretically. The previous version of the table based matching algorithm was very unstable because of the bias by text lengths. Therefore, the task of this research is to improve the table based approach into the more stable version.

III. Normalized Table Matching Algorithm

This section describes a table based matching approach to text categorization. Figure 2 illustrates conceptually the architecture of the proposed text categorization system. The part, 'Encoding' encodes a document into a table as the interface of the system, and will be described in detail in Section III-A. In Section III-B, we will describe the process of computing a similarity between two tables; the computation is used for classification of unseen documents. In Section III-C, we will describe the process of learning sample labeled documents and classifying unseen documents using the proposed approach.

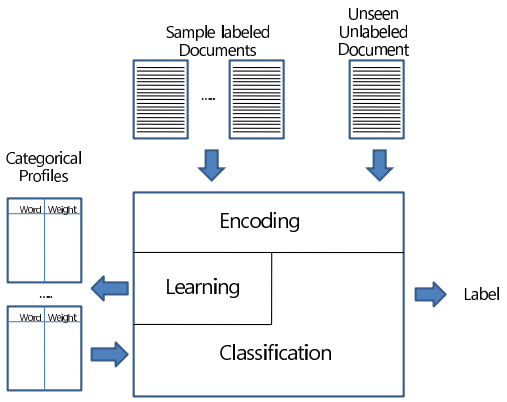


Figure 2. The Process of Indexing Corpus

A. Document Encoding

This section is concerned with the part, 'Encoding' of the architecture of the text categorization system which is illustrated in Figure 2. Here, *document encoding* is defined as the process of mapping a document into a table. Figure 3 illustrates the process of encoding a document so through the five steps. As illustrated in Figure 3, a particular document is given as the input and its corresponding table is generated as the output. In this section, we will describe in detail each of the five steps involved in the document encoding.

The first step of document encoding is tokenization as shown in Figure3. A full text in a document which is wirtten

 ISBN: 978-1-61208-217-2

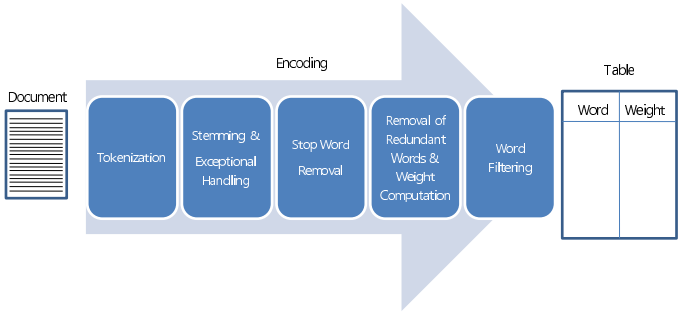


Figure 3. The Process of Encoding a Document into a Table

in a natural language is given as the input of this step. This step, 'tokenization', segments a full text into tokens by white space or punctuation mark. The step generates a list of tokens as the output. A token refers to a word in its raw form.

The second step of document encoding refers to stemming & exception handling. The list of tokens which is generated from the previous step is given as the input of this step. This step converts each token into its root form by stemming it or applying an exception rule to it. This step is carried out by loading stemming & exception rules each of which specifies conversion of each word into its root. Therefore, a list of words in their root forms is generated as the output of this step.

The third step of document encoding is to remove stop words from the list of words. A stop word refers to a grammatical word which do only grammatical functions, irrelevantly to the content of the original document. In English, conjunctions, pronouns, prepositions, and so on belong to this kind of words. Removing the kind of words is necessary for processing documents more efficiently in context of text mining and information retrieval. This step usually remains verbs or nouns as its output.

The fourth step is to remove redundant words and compute weights of each of remaining words. A list of words in their root forms except stop words is given as the input of this step and redundant words are removed among them. The weight of each word indicates how much important it is in terms of the relevancy to the content of the given document. The weight is computed using equation (1),

$$weight_i(w_k) = tf_i(w_k)(\log_2 D - \log_2 df(w_k) + 1) \quad (1)$$

where $weight_i(w_k)$ indicates the weight of word, w_k, relevantly to the content of document, i; $tf_i(w_k)$ indicates the frequency of the word, w_k, in the document, i; D means the total number of documents in the referenced corpus; and $df(w_k)$ indicates the number of documents of the corpus including the word, w_k. A particular corpus is required for computing weights of words using equation (1), and a list of pairs of a word and its weight is generated as the output of this step.

Although stop words and redundant words are removed, we need to filter out additionally words with lower weights for more efficient processing. The previous version which Jo and Cho proposed in 2008 [10], omitted the word filtering, so it took very much time for processing documents for tasks of text categorization. Especially when computing a similarity between two tables, its complexity is quadratic $O(n^2)$, so we need to cut down the size of tables as much as possible, minimizing information loss. We can consider two kinds of schemes for filtering words with their lower weights. One is called rank filtering, where a fixed number of words with their higher weights is selected, and the other is called threshold filtering where weights of words are normalized as continuous values between zero and one, and words with their weights higher than the threshold are selected. In this research, the former is adopted.

B. Similarity between two Tables

This section is concerned with the computation of a similarity between two tables. Two tables each of which represents a document or a group of documents are given as the input. A table which consists of words shared by the two tables is derived from the two tables. A similarity between the two tables is computed based on the shared words in the derived table. Whether weights of words are given as normalized or unnormalized values, the similarity is always generated as a normalized value.

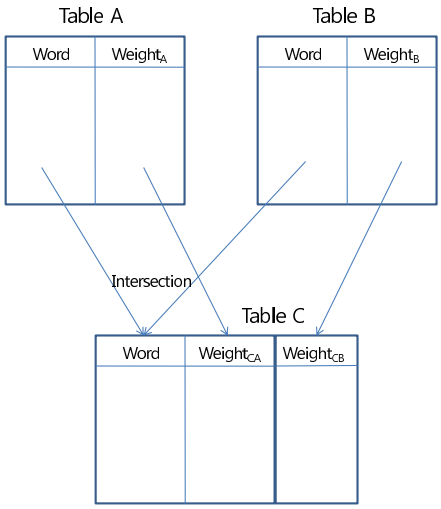


Figure 4. The Process of Deriving a Table from the Two Input Tables

The process of deriving a table from the two input tables is illustrated in Figure 4. Let table A and table B in Figure 4 be the source tables. Let table C be the destination table which is derived by extracting shared words from the source tables. Table C consists of words shared by both source tables. Each

 ISBN: 978-1-61208-217-2

entry of the destination table consists of a shared word and its two weights: one is from table A and the other is from table B.

A similarity between two tables is computed using equation (2)

$$similarity = \frac{weight_{CA} + weight_{CB}}{weight_A + weight_B} \quad (2)$$

where $weight_{CA}$ and $weight_{CB}$ indicate sums of weights of common words from table A and B, respectively, and $weight_A$ and $weight_B$ indicate sums of weights of total words in table A and B, respectively. A similarity computed by equation (2) is bound from 0 to one as a normalized value. If there is no shared word between the two tables, the similarity becomes zero. If the two source tables are exactly same as each other, the similarity becomes one. Therefore, even if weights of words are given as non-normalized values, it is guaranteed that the similarity is given as a normalized value.

We demonstrate the computation of the similarity through a simple example. Two source tables are given in Table I. The destination table is derived from the two source tables as illustrated in Table II. The similarity between the two source tables is computed based on the destination table using equation (2) as follows:

$$\frac{1.5}{1.2 + 1.7}$$

Therefore, the similarity in this example becomes 0.51.

Table I
TWO SOURCE TABLES: TABLE A (LEFT) AND TABLE B (RIGHT)

Table A		Table B	
computer	0.3	computer	0.6
system	0.2	system	0.4
hardware	0.5	information	0.5
CPU	0.2	data	0.2

Table II
DESTINATION TABLE: TABLE C

computer	0.3	0.6
system	0.2	0.4

C. Learning & Classification

This section is concerned with the process of learning sample labeled documents and classifying an unseen document. There exist two functions in the text categorization system: learning and classification. Learning refers to the process of building rules or equations of classification using sample labeled documents in context of text categorization. Classification refers to the process of classifying an unseen document based on the defined rules or equations. Note that learning is prerequisite for classification.

In the view of the proposed text categorization system, learning is defined as the process of building tables corresponding to categories using sample labeled documents. Categories are predefined, and sample documents are allocated to their corresponding categories. Figure 5 illustrates the part, 'Learning', in the proposed text categorization system which is illustrated in Figure 2. From a collection of documents labeled identically, as the learning process, a table is built and called categorical profile in this paper; learning is carried out by attaching the concatenation which concatenates full texts of documents into a full text, to the process of encoding which is illustrated in Figure 3. Therefore, learning generates categorical profiles as many as categories as its output as shown in Figure 5.

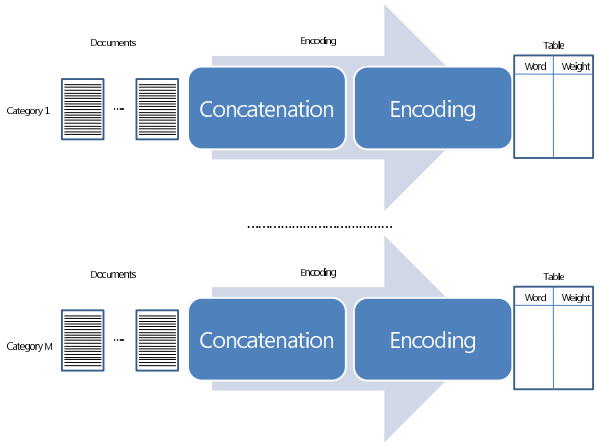


Figure 5. The Process of Learning Sample Labeled Documents in the Proposed Text Categorization System

Classification is defined as the process of deciding one of the predefine categories to each unseen document. The process of classifying an unseen document is illustrated in Figure 6. An unseen document is encoded into a table by the process illustrated in Figure 3, as shown the left part of Figure 6. As shown in the middle part of Figure 6, similarities of the table with categorical profiles given as tables are computed; the computation was already described in Section III. Therefore, the unseen document is classified into the category corresponding to the maximum similarity between its table and the corresponding categorical profile.

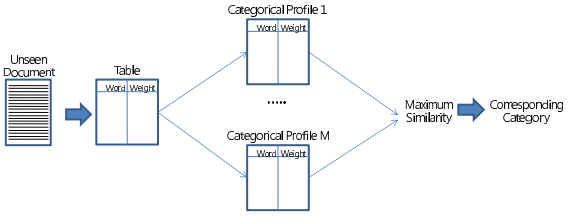


Figure 6. The Process of Classifying a particular Unseen Document

IV. EXPERIMENTS AND RESULTS

This section is concerned with the empirical results of the first set of experiments. The test data used in this set of experiments is a collection of news articles called 'NewsPage.com'. The texts in the collection encoded into numerical vectors for using the machine learning based approaches, and tables for using the proposed one. We selected the four categories and decompose the tasks of categorizing news articles into the four binary tasks. In this section, we describe the test data and the experimental process, present the empirical results, and discuss on them.

In this set of experiments, we use the collection of news articles 'called NewsPage.com'. The collection was built by copying and pasting news articles available in the web site, "newspage.com", manually into plain text files. The collection is partitioned into the training set and test set as shown in table 1. The four categories are selected among the five, and the task is decomposed into the four binary classifications as many as the selected categories.

The configurations of the approaches participating in the experiments are presented in Table III. In using the KNN, the number of nearest neighbors is set three. In using the SVM, the kernel function, the capacity, and the maximum iteration are set the inner product, 4.0, and 1,000, respectively. In using the MLP, the learning rate, the number of hidden nodes, the iterations, are set 0.1, 10, and 1000, respectively. Texts are encoded into 100 dimensional numerical vectors and ten sized table for using the three machine learning algorithms and the proposed approach, respectively.

Table III
THE CONFIGURATIONS OF THE APPROACHES PARTICIPATING IN EXPERIMENTS

Approaches	Configurations	Document Encoding
Naive Bayes	N/A	100 dimensional numerical vectors
KNN	$K = 1, 3$	
NNBP	#Hidden Nodes=10 #Epochs=500 Learning Rate = 0.1 Sigmoid Function	
SVM	Capacity=4 Inner Product	
Proposed Approach	10 sized Tables	

The results from this set of experiments are illustrated in Figure 7. The y-axis indicates the value of F1 measure and the white bar indicates the result of the proposed approach. As shown in Figure 7, the proposed approach is better outstandingly in the category, 'business', than the three approaches. The approaches involved in this set of experiments are comparable to each other in the rest categories. When considering the simplicity and input size, the proposed is more recommendable than the others, even if its performance is comparable to those of the others.

Table IV presents the average F1 measures and the variances of the approaches over the four categories. The

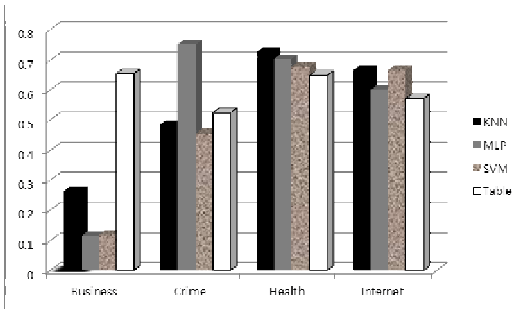


Figure 7. The Results from the Set of Experiments in NewsPage.com

variance of F1 measures indicates the stability of the approaches to the categories; the smaller it is, the more stable. The proposed approaches have largest averaged F1 measure since it is outstandingly better in the category, 'business', as shown in Figure 7. It has smallest variance of its F1 measures; it indicates that it has the best stability in addition. Therefore, from this set of experiments, we conclude that the proposed approach works better and more stable than the three approaches based on Table 2.

Table IV
THE OVERALL PERFORMANCE AND STABILITY OF APPROACHES IN NEWSPAGE.COM

	KNN	MLP	SVM	Table Matching
F1 Average	0.5349	0.5426	0.4796	0.5998
F1 Variance	0.0327	0.0636	0.0511	0.0030

V. CONCLUSION

Let us consider the significances of this research. Like the previous version of the table based algorithm, we are free from the three main problems in encoding texts into numerical vectors: huge dimensionality, sparse distribution, and poor transparency. Because the tables representing texts are symbolic, we trace the classification more easily, in order to provide the evidences. In the proposed version, the categorical scores of the given text are independent of its length. The table based approach is improved to reach more stable performance as shown in the set of experiments presented in Section V.

In order to reinforce the current research, we may consider the four directions of further research. In the first direction, we need to validate the categorization performance of the proposed approach in multiple labels categorization as well as single label one. In the second direction, we may consider that a document or documents are encoded into a committee of tables rather than a table by using multiple schemes for weighting words. In the third direction, in order to keep efficiency and reliability of the proposed approach, we may build the text categorization system in evolutionary fashion by incrementing tables gradually. In the last direction, we

 ISBN: 978-1-61208-217-2

may implement a text categorization system as a prototype program where the proposed approach is adopted.

REFERENCES

[1] B. Massand, G. Linoff, and D. Waltz, “Classifying News Stories using Memory based Reasoning”, pp. 59-65, The Proceedings of 15th ACM International Conference on Research and Development in Information Retrieval, 1992.

[2] Y. Yang, ”An evaluation of statistical approaches to text categorization”, pp. 67-88, Information Retrieval, vol 1, no 1-2, 1999.

[3] D. Mladenic and M. Grobelink, “Feature Selection for unbalanced class distribution and Naive Bayes”, pp. 256-267, The Proceedings of International Conference on Machine Learning, 1999.

[4] S. Eyheramendy and D. Lewis and D. Madigan, “On the Naive Bayes Model for Text Categorization ”,pp. 165-171, The Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics, 2003.

[5] H. Drucker, D. Wu, and V. N. Vapnik, “Support Vector Machines for Spam Categorization”, pp. 1048-1054, IEEE Transaction on Neural Networks, vol 10, no 5, 1999.

[6] N. Cristianini and J. Shawe-Taylor, “Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, 2000.

[7] T. Jo, “NeuroTextCategorizer: A New Model of Neural Network for Text Categorization”, pp. 280-285, The Proceedings of ICONIP 2000, 2000.

[8] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, “Text Classification with String Kernels”, pp. 419-444, Journal of Machine Learning Research, vol 2, no 2, 2002.

[9] K. Lee and K. Kageura, “Virtual relevant documents in text categorization with support vector machines”, pp. 902-913, Information Processing and Management, vol 43, no 4, 2007.

[10] Taeho Jo and Dongho Cho, ”Index Based Approach for Text Categorization”, pp. 127-132, International Journal of Mathematics and Computers in Simulation, vol 2, no 1, 2008.

[11] Taeho Jo, ”Table based Matching Algorithm for Soft Categorization of News Articles in Reuter 21578”, pp. 875-882, Journal of Korea Multimedia Society, vol 11, no 6, 2008.

[12] Taeho Jo, ”Single Pass Algorithm for Text Clustering by Encoding Documents into Tables”, pp. 1749-1757, Journal of Korea Multimedia Society, vol 11, no 12, 2008.

 ISBN: 978-1-61208-217-2

A Dynamic (m,k)-firm Constraint to Avoid Dynamic Failures in a Real-Time DBMS

Sami Limam, Leila Baccouche
Riadi-GDL Laboratory
Insat, National Institute of Applied Science And Technology
C.U. Nord, B.P 676, Tunis Cedex 1080, Tunisia
Limamsami@yahoo.fr
Leila.Baccouche@free.fr

Henda Ben Ghezala
Riadi-GDL Laboratory
ENSI, National School of Computer Science
University of la Manouba, Tunisia
hhbg.hhbg@gnet.tn

***Abstract*—Current applications such as stock markets, e-business, multimedia and telecommunications require further real-time services. Such systems deal with large quantities of data, and transactions have temporal constraints. Among these applications, some of them must face unpredictable workloads. The aim of our work is to maintain a robust RTDBS behavior and to decrease the number of transactions which miss their deadline.In this paper, we propose an approach based on the (m,k)-firm model that allows to control the RTDBS behavior, in particular during the instability phases, in order to improve the quality of service (QoS) of applications. Multi-class transactions are scheduled by DBP_CC and we have to adapt QoS level for each class to the load of the system.**

Keywords-(m,k)-firm; real time scheduling; Imprecise Computation.

I. INTRODUCTION

Recently, the demand for real-time services has increased considerably in many applications such as multimedia applications (video conferencing, video on demand services,Web services and e-business,) . In addition, these applications reflect an important requirement in data management. Therefore, real-time database systems (RTDBS) become the support to the implementation of these applications. In these applications, it is important to obtain complete and accurate results before a certain deadline, while using fresh data. However, as the users requests remain unforeseeable, the RTDBS can be overloaded leading to the system inability to meet deadlines and to control the data freshness.

The real-time systems distinguish generally between firm and hard real time tasks. In firm systems, Authors suggested relaxing the real time constraints of the tasks such as using the (m,k)-firm model [8], [5], [11] or imprecise computations [13]. The notion of (m, k)-firm constraints was introduced in particular for periodic tasks [9]. The (m,k)-firm model tolerate some deadline misses according to the (m,k)-firm constraints.

In this paper, we propose such an architecture which allows a differentiation of services between transaction classes. To this end, we exploit the (m,k)-firm transactions model and we propose a scheduling algorithm based on this model, which guarantees a certain QoS for each transaction class. However, we notice that if the system is overloaded, the (m,k)-firm constraints are not respected. Thus we must adapt the (m,k)-firm constraints to system load and introduce imprecision in order to respect these constraints.

In Section 2, we present the related work in management of QoS in RTDBS. In Section 3, we present the architecture that we propose to manage transaction classes: we present our transaction's model, our queue's model. Then we formulate the problem. In Section 4, we present our solution to manage transactions classes in overload situations. Section 4 is devoted to the presentation of the simulations we conducted to validate our model, as well as to the comments on the results obtained. The simulation platform used is called RTDS (Real-Time Database Simulator) that we have developed in our laboratory. The simulations results show the effectiveness of the proposed architecture and its ability to support overload situations and to guarantee QoS for each transaction class. Finally, we conclude the paper.

II. RELATED WORK

As one of the first efforts to address QoS management in RTDBs, Kang et al. [10] developed a novel QoS management architecture called QMF. In QMF, a formal control theoretic approach is applied to compute the required workload adjustment considering the miss ratio error, i.e., the difference between the desired miss ratio and the currently measured miss ratio. Feedback control is used since it is very effective to support the desired performance. To the feedback control loop, Amirijoo et al. [1] added the use Imprecision in order to manage the QoS. Imprecise computation techniques [13] provide means for achieving graceful degradation during transient overloads by trading off resource needs for the quality of a requested service. Imprecise computation and feedback control scheduling have been used for QoS management of RTDBs. In this approach the notion of imprecise computation is applied on transactions as well as data, i.e., data objects are allowed to deviate, to a certain degree, from their corresponding values in the external environment. Many other works trade with QoS by using imprecision. Bouazizi et al. [6] proposed an approach based on Feedback Control Scheduling (FCS) that allows to control the RTDBS behavior, in particular during the instability

 ISBN: 978-1-61208-217-2

phases, in order to improve the quality of service (QoS) of applications. They proposed a technique which allows to minimize the number of conflicts by exploiting multi-versions data, while taking into account the database size.

III. System architecture

A. Transaction's Model

In a RTDBS, we distinguish two types of transactions:

- Update Transactions: they are periodic and have to refresh regularly the database by updating data reported by sensors.
- User transactions: they read/write non real-time data and/or only read real-time data. They are generally non periodic. As their arrival in the system is unpredictable, they may cause overload situations. That is why imprecise transactions models are often used.

In this paper, we assume that transactions have firm deadlines, i.e., a transaction which misses its deadline becomes useless and is aborted. In addition, we exploit the imprecision model of transactions in overload situations [13]. A transaction consists in a mandatory part and an optional part. The mandatory part must be executed before the transaction deadline. Optional part is composed of sub-transactions which are executed if there remains enough time before deadline. More the number of optional sub-transactions executed is great, better is the result, i.e., the quality of service of the result is enhanced.

We base our work on (m,k)-firm model. A transaction is divided into k sub-transactions. To each sub-transaction is assigned a weight according to the criticality of the data it accesses. The m (less than k) sub-transactions having the higher weight are labeled mandatory; the others are optional. So, the transaction is composed of m mandatory sub-transactions and (k-m) optional sub-transactions. Like in Milestone model, the execution of the optional sub-transactions will make the result more precise. The transaction is successfully executed if at least the m mandatory sub-transactions are executed before deadline.

B. Queue's Model

We adopt a multi queues model with a single server. Indeed, in our model of transactions, we have different types of transactions, executed by only one processor. We begin to determine the number of queues needed by the architecture. We have defined a parameter, called Importance which will differentiate the types of transactions. We define three levels of importance:

- Update transactions.
- High importance user transactions that perform write operations.
- Low importance user transactions that perform read operations.

In our queue model, we have as many queues as levels of importance. So, we can differentiate service between transaction classes.Also, as user and distributed transactions consist of a set of mandatory sub-transaction and several optional sub-transactions, we have mandatory parts and optional parts, in each queue. As transaction classes need to be separated to allow service differentiation, each queue is divided into two sub-queues: one for the mandatory parts and one for the optional parts (cf. Fig. 1).

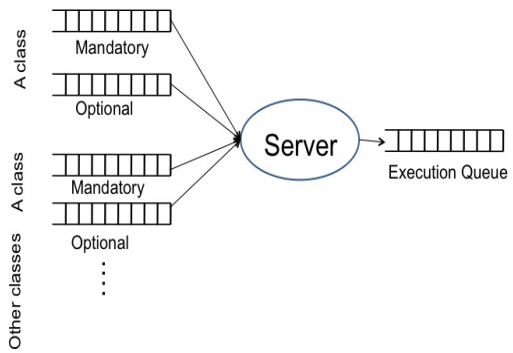


Figure 1. Queue model

C. Serving queues

In a real-time system, attempting to guarantee the respect of all transaction deadlines often leads to overload of the system. Then, to avoid this situation where the system becomes unstable, its better to guarantee a certain level of quality of service instead of guaranteeing all the deadlines. This is the idea behind (m,k)-firm model. The principle of (m,k)-firm model is to guarantee that m tasks meet their deadlines among k consecutive tasks. Therefore, we tolerate that (k-m) tasks miss their deadlines, among any sequence of k tasks.

Our problem is to schedule transactions inserted in different queues. We thus choose DBP algorithm [9], [11] already proposed to schedule network packets on several queues. This algorithm is adapted to the real-time context and allows to guarantee QoS. However, it does not include an access control to data, i.e., a concurrency controller.

For this purpose, we proposed DBP_CC algorithm (Distance Based Priority and Concurrency Control) [3], which is an adaptation of DBP algorithm [9](Distance Based Priority) to the RTDBS context.

DBP is a dynamic algorithm issued from the (m,k)-firm model. The algorithm we propose provides several levels of QoS described by various couples (m,k) specified by the database administrator for each class of tasks.

DBP uses the history of the execution to compute the DBP priority of each queue and determine the queue which is going to miss its (m,k)-firm requirements and be in dynamic failure state (less than m tasks among k consecutive tasks respect their deadlines). The selected queue is considered of

 ISBN: 978-1-61208-217-2

high priority DBP and the task at the head of the queue is extracted and served.

DBP saves the history of execution in a structure named k-sequence which is a sequence of k bits updated after task execution (1 indicates the respect of deadline and 0 the opposite). The priority (distance) is computed by DBP in the following way:

$$Priority_{DBP} = k - l(m, s) + 1$$

where l(m,s) is the position leaving from the right of the m^{th} success (1) in the k_sequence, s (the state of the queue). If there are less than m byte 1 in the k-sequence s then l(m,s) = k+1. Each queue has its own (m,k)-firm constraint and its own k-sequence. Before extracting the transaction at the head, a conflict detection and resolution test is executed.

D. Problem Formulation

This policy which we adopt has numerous advantages:

- It allows a differentiation between transactions (We so obtain five classes and a queue is suited to every class of transactions).
- It allows a differentiation of service between classes. This is guaranteed by the attribution of a (m, k)-firm constraint specified by the administrator of the database and who fix the desired quality level. Besides, the algorithm DBP_CC aims to respect for this constraint by extracting a transaction from the closest queue to the state of dynamic failure. In addition, a differentiation is made during the execution step as the system assigns more time to the execution of the most important classes.
- We can introduce imprecision easily. With this model we can go of the most precise case (no tolerance to the imprecision: constraints (k, k)-firm for the diverse queues and the constraint (1,1)-firm for the transactions) in the most imprecise case. The degree of imprecision is controlled and it has two types:
  - Imprecision of queues: controlled by an (m, k)-firm constraint for each queue and which gives an idea of the desired level of QoS of the queue: m transactions have to succeed for any window of k successive transactions and (k-m) transactions at most can fail.
  - Imprecision of transactions: controlled by a single (m, k)-firm constraint for the user transactions as only them are decomposable.

Naturally, it also presents inconveniences such as:

- In front of an overload, all the queues sink in dynamic failure. In that case, the algorithm DBP_CC applies the EDF policy which is inappropriate in the case of overload.
- The imprecision is introduced with only the (m,k)-firm model. We can add to our model other techniques of imprecision to decrease the probability of dynamic failure (as the notion of epsilon-data or delta-deadline[7]).

IV. Solution

To improve this policy, it is necessary to address both quoted problems:

A. Dynamic Failure

As we consider an opened system, the overload is a state which can arise. In contrast to a closed system where all the transactions are known (their resources also), in an opened system we have transactions from the users that are unpredictable. When the system becomes overloaded, the system is unable to execute all the transactions before their deadlines. If the overload is important, the (m, k)-firm constraint is violated because we have less than m transactions which respect their deadlines. So, the queue is in a state of dynamic failure and if we have several queues in our queue's model, all will meet in this state. According to DBP_CC, if all the queues are in a state of dynamic failure, the EDF policy is applied (badly suitable to the excess loads) and we notice a decrease of the performances.

To address this problem, we have to avoid the state of dynamic failure or to minimize it. So the solution of this problem is quite clear: the priorities of queues have not to be equal to 0.

The DBP priority of a queue is according to the position of the m^{th} 1 (success of the transaction). Thus more m is big, smaller will be the priority and we approach the case of the dynamic failure (priority equal to 0). This is confirmed by experiments when we choose (m, k)-firm constraints where m approaches of k. In such systems, if some transactions miss their deadlines, the queue is in dynamic failure. In this situation, the system is forced to extract transactions of this queue and a transaction has to succeed to go out of the state of dynamic failure. In case we have flexible constraints, the system is less often in state of dynamic failure. In view of these observations and their confirmation by simulations, we intend to deploy a policy which decreases the probability of dynamic failure. This is possible by the following two actions:

- Apply as departure a flexible (m, k)-firm constraints (where m goes away from k).
- Decrease the probability of dynamic failure for the diverse queues. This can be made only on acting on the DBP priorities (take away these priorities of 0 has for consequence to decrease the probability of dynamic failure)

B. How to avoid Dynamic Failure

To decrease the probability to be in dynamic failure means to avoid having a priority equal to zero. The solution thus is to analyze the formulation of the priority DBP_CC and

 ISBN: 978-1-61208-217-2

to ensue from it the solution. As we know, the priority expression is as follow:

$$Priority_{DBP} = k - l(m, s) + 1$$

We have the following parameters in this formulation:

- The parameters of the constraint (m, k)-firm: m and k.
- The history execution of the k last transactions: the k-sequence.

We can act only on the (m, k)-firm constraint as we cannot touch the execution history of the queue. Indeed, when the system is overloaded, to require less allows improvement of the performances. Thus, if the priority of a queue aims towards zero, we can decrease the value of m. Consequently l (m, s) decrease and then the DBP_CC priority increase. So, to take away the priority of zero it is enough to decrease the level of wished QoS (m). However it is necessary to clear up when and how?

- When: indeed on step when this priority becomes equal to zero because there it is too late. It is necessary to specify a threshold S below which we suppose that the system aims towards the failure. This presents another advantage as the system becomes preventive to the overload state.
- How: by specifying a law of regression of m. Indeed, m has to oscillate between its original value and a minimal value.

C. Expressing dynamically the (m,k)-firm constraint

The policy is to decrease the value of m when we notice that the queue approaches the state of dynamic failure. The detection of the approach of a state of dynamic failure is noticed by fixing a threshold for every queue below which we envisage the regression of m. However, we cannot decrease m until it reach its minimal value of 1 because in that case all the queues will have the same constraint (the same importance). Thus, it is necessary to fix for every queue a minimal value of m (by operating a differentiation of service so that $m_{minH} > m_{minM} > m_{minL}$).

By drawing up the curve of regression of m between its original value and its minimal value, we notice that m follows an exponential function between the priority 0 and the fixed threshold. Except this interval m takes its original value.

So, we obtain the following mathematical formulation:

$$m = m_{min} + (c \times Priority^{\Omega}) \quad if\ 0 \leq Priority \leq Threshold$$

$$m = m_{original} \quad if \quad Priority \geq Threshold$$

By setting (priority = Threshold), we can fix the values of c and Ω and we obtain:

$$c \times Threshold^{\Omega} = m_{original} - m_{min}$$

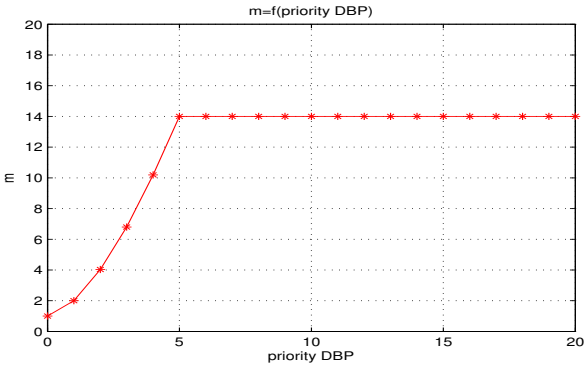


Figure 2. Evolution of m according to the DBP priority

D. More imprecision

To introduce more imprecision during the phases of overload, we can resort much techniques which introduce imprecision at the level of data and transactions. The imprecision is introduced on two levels: data and Transactions. We deal with Quality of Data (QoD) and Quality of Transactions(QoT). The QoD can be translated by the notion of epsilon-data or Data-Error where the data becomes valid on an interval not on a given value. It can also be translated in the notion of multi-versions data. The QoT is translated by a partial execution of the transaction or its replacement by the other one less expensive at CPU time. So the main advantage of our approach is that it combines QoD and QoT. It integrates two notions :

- the notion of ϵ - data
- the notion of Δ-deadline

1) ϵ-data: Under epsilon-data, the isolation is not maximal as a transaction can read a data which is being updated by another transaction. Thus, a transaction T which wishes to acquire a lock in a conflicting mode (W/R or W/W) can obtain this lock provided that the generated imprecision does not overtake epsilon.

2) Δ-deadline: Under delta-deadline, we can relax the deadline of a transaction and so the new deadline is equal to the former one to whom we add Δ.

V. Simulations and Results

A. Context

Simulation of the protocols developed for RTDBS requires a platform which respects transactions ACID (Atomicity, Consistency, Isolation, Durability) properties and which offers adequate mechanisms to the support of time constraints. Most research teams on RTDBSs developed their own simulator : Beehive [14], DeeDS [2], J-Radex [12]. In our research team, we have developed a simulator, called RTDS [4], which is described briefly, bellow.

 ISBN: 978-1-61208-217-2

B. RTDS simulator

RTDS [4] is a discrete-event simulator written in Java, designed by our research team to simulate real-time database behavior. It is especially devoted to test and to compare concurrency control and scheduling algorithms that are proposed in a RTDBS.

C. Simulation parameters

Parameter	Value
Number of ressources	100
Number of temporal ressources	20
Number of non real-time resources	80
Load of update transactions	40
Execution time of periodic transactions	10 to 20 ms
Execution time of a user transaction	70 to 100 ms
Number of mandatory sub-transactions in an user transaction	1
Number of optional sub-transactions	1 to 4
Number of resources used per sub-transaction	1

Table I
SIMULATION PARAMETERS

D. Simulation results

We carried out four experiments:

1) In the first simulation, all transactions are inserted in the same queue. So, in our Queue's Model we have one single queue. Then we apply EDF (Earliest Deadline First) to schedule this queue.
2) In the second simulation, we show the impact of the system load on the system performances. In the simulation, we have five queues in our Queue's Model, each queue holding a class of transactions (mandatory and optional parts of a transaction are put in different queues).
3) In the third experiment, the (m,k)-firm constraint becomes dynamic and it is expressed formally according to the established equation. We show that the system prevents the state of overload by relaxing the (m,k)-firm constraints of queues which approach the dynamic failure. This relaxation has for consequence improvement of results.
4) In the last experiment, we measure the impact of the introduction of imprecision on the performances of the system.

1) Simulation 1: Effect of Scheduling policy and Queue's Model: In this simulation, our queue's model includes a single queue that will contain the different classes of transactions. We use EDF to schedule the queue and the queue is ordered according to the deadline (the transaction in the top is that having the nearest deadline).

In Figure 3, we notice:

- EDF is not suitable for transient overload: When the load of the system increases, we have transactions with nearest deadlines trying to execute and they are unable to meet their deadlines. In addition, the time wasted to execute failed transactions has an impact on the respect of deadlines of waiting transactions.
- Inserting transactions on a single queue leads to a poor differentiation of service as the criteria that matters is the deadline not the importance of the transaction.

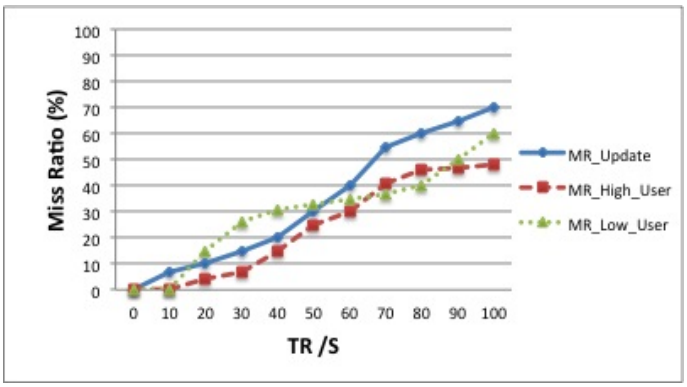


Figure 3. Transaction Miss Ratio (MR) when using EDF, with one single queue

2) Simulation 2: Variation of the system load: Figure 4 traces the variation of the miss ratio according to the system load. We vary the system from under loaded state (10 transactions per second) to an overload state (arrival rate of 40 transactions per second). In under loaded state, all transactions meet their deadlines, whereas in overload state several transactions miss their deadlines. We notice that the system becomes overloaded at the arrival rate of 40 transactions per second. We also note that when we use the (m, k)-firm model, we can differentiate service between different classes of transactions, thus, transactions' miss ratio of the prioritized class is lower than other classes.

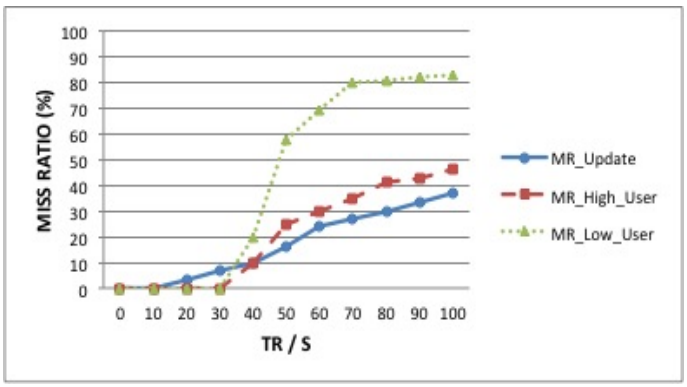


Figure 4. Transaction Miss Ratio (MR) when using (m,k)-firm model, with five queues in the system

In Figure 4, MR_Update, MR_High_User and MR_Low_User have the following meaning respectively: miss ratio of update transactions, miss ratio of high

	m	K	m_{min}	Threshold	c	Ω
Update	18	20	10	2	6	1
High Mandatory	14	20	6	5	1.2	1
High Optional	7	20	2	1	5	1
Low Mandatory	4	20	1	1	3	1
Low Optional	1	20	1	1	0	0

Table II
QUEUES PARAMETERS

importance user transactions and miss ratio of low importance user transactions. In the simulation, we have three classes of transactions dispatched in five queues: Update, High_Mandatory, High_Optional, Low_Mandatory and Low_Optional with respectively the following (m,k)-firm constraints: (18,20), (14,20) ,(7,20), (4,20) and (1,20).

3) Simulation 3: A dynamic (m,k)-firm constraint: In the second experiment, m follows the mathematical function established previously and we have the parameters expressed in Table II.

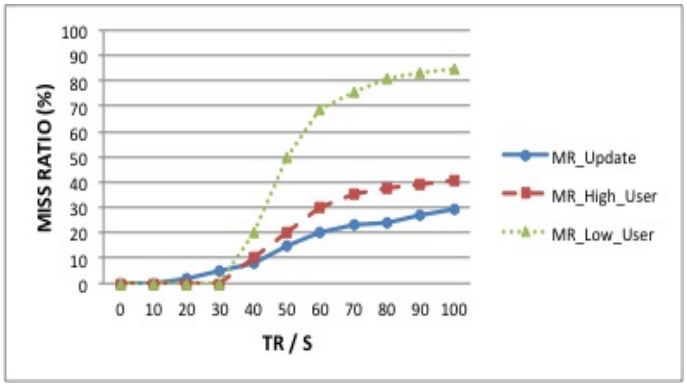


Figure 5. Transaction Miss Ratio when we apply dynamic (m,k)-firm constraints

In the Figure 5, we notice that usage of dynamic (m,k)-firm constraints allows to improve the results as they decrease the probability of dynamic failure. We notice that the performances in this simulation are better than those of the first simulation. However, the performances with flexible constraints are better as the flexible constraints correspond to the minimal values required for m (less probability of dynamic failure). Applying the minimal constraints present however two anomalies:

- The performances do not correspond to the wished levels of QoS: the performances are widely superior to the constraints even when the system is overloaded or not.
- The distance between the constraints is small and so we have a bad differentiation of service.

To address the noticed problems, we suggest the use of imprecise actions, to specify hard constraints and lead the system to respect them.

4) Simulation 4: Impact of imprecision.: In the last simulation, we apply dynamic constraints and we use imprecise actions. Indeed, if the DBP priority becomes lower than the threshold specified for the queue, we decrease the value of m and we apply two imprecise actions:

- We omit to execute any transaction which wishes to update a data if the new value is meanwhile [real value - ϵ , real value + ϵ] and we consider them as successful. This action (allows us to spare CPU time whom we can assign to the execution of the other transactions.
- We relax the deadlines of all the transactions. So, the system can respect the new ones. The performances obtained in Figure 6 confirm that the combination of (m, k)-firm constraints and imprecision allow us to obtain the best performances while respecting the levels of quality of service specified.

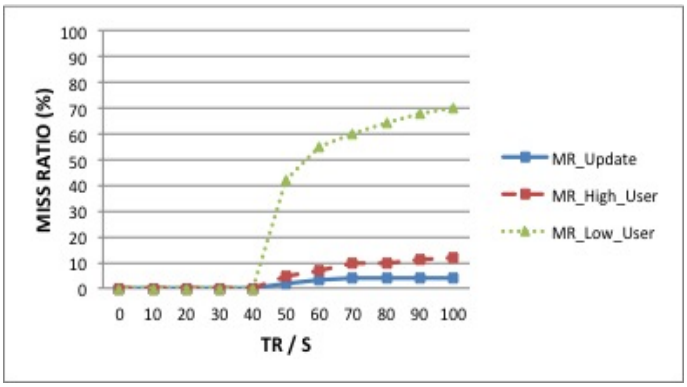


Figure 6. Transaction Miss Ratio when we apply dynamic (m,k)-firm constraints and we use imprecision

VI. CONCLUSION AND FUTURE WORK

In RTDBS, execution of transactions before their deadlines often leads to overload situations, during which the system performances are degraded. Based on (m,k)-firm model, we proposed, in this paper, an approach to control this degradation for each class of transactions. Results of the simulations we have carried out have shown that when using dynamic constraints and imprecision, we are able to differentiate service in a RTDBS according to the transaction classes. A transaction consists of a mandatory part and several optional parts. We insert each transaction class into two queues: one containing the mandatory part and the other contains optional parts. Then, we choose an (m,k)-firm constraint for each queue and we serve queues by applying the DBP_CC algorithm. Finally, when a queue approaches a dynamic failure, we decrease formally the value of m and we use imprecise actions in order to respect levels of QoS of each queue. In the near future, we plan to deploy and adapt this approach to a distributed context. Also, we project to use the (m,k)-firm model for load balancing or data replication.

 ISBN: 978-1-61208-217-2

REFERENCES

[1] M. Amirijoo, J. Hansson and S. Son. Specification and Management of QoS in Real-Time Databases Supporting Imprecise Computations. IEEE Trans. Computers, 55(3), 304-319, 2006.

[2] S.F. Andler, J. Hansson, J. Eriksson, J. Mellin, M. Berndtsson and B. Eftring. DeeDS Towards a Distributed and Active Real-Time Database System. ACM SIGMOD Record, 25(1), pp. 38-40, 1996.

[3] L. Baccouche and S. Limam. (m,k)-firm scheduling of real-time transactions with concurrency control. In Proceedings of the second international conference on systems, ICONS'07, Guadeloupe, French Caribbean, pp. 14-21, April 2007.

[4] L. Baccouche and S. Limam. RTDS: A component and Aspect-based Real-Time Database System Simulator. In Proceedings of ESM08, Le Havre, October 2008, pp. 551-555.

[5] G. Bernat, A. Burns and A. Llamos. Weakly Hard Real-Time Systems. IEEE Transactions on Computers, pp. 308-321, 1999.

[6] E. Bouazizi, C. Duvallet and Bruno Sadeg. Using Feedback Control Scheduling and Data Versions to enhance Quality of Data in RTDBSs. Proceedings of International Computer System and Information Technology (IEEE ICSIT'2005), Alger, Algeria, 322-327, 2005.

[7] S. Bouzefrane, J.P.Etienne, C. Kaiser. Handling Overload and Data-Relaxation Control in Distributed Real-Time Database Systems. I. J. Comput. Appl. 15(3),pp. 187-200 (2008).

[8] C. Evequoz. Guaranteeing Optional Task Completions on (m,k)-Firm Real-Time Systems. Proceedings of the 2010 10th IEEE International Conference on Computer and Information Technology.CIT '10. pp. 1772–1779.2010.

[9] M. Hamdaoui and P. Ramanathan. A dynamic priority assignment technique for streams with (m,k)-firm deadlines. IEEE Transactions on Computers, 44(12), pp. 1443-1451 (1995).

[10] K. Kang, S. Son and J. Stankovic Managing Deadline Miss Ratio and Sensor Data Freshness in Real-Time Databases. IEEE Trans. Knowl. Data Eng, 16(10), pp. 1200-1216, 2004.

[11] A. Koubaa and Y-Q. Song. Graceful Degradation of Loss-Tolerant QoS using (m,k)-Firm Constraints in Guaranteed Rate. Networks Computer Communications Journal, (Elsevier), Vol. 28/12, pp. 1393-1409, July 2005.

[12] J. Haubert, B. Sadeg, L. Amanton. and R. Coma. J-RADEx : un simulateur de SGBDTR convivial. Journal of Information Science for Decision Making, pp. 64-71, 2004.

[13] J.W.S. Liu, W.K. Shih, K.J. Lin, R. Bettati and J.Y. Chung. Imprecise Computations, Proc. IEEE, 82(1), pp. 83-94 (1994).

[14] J. Stankovic, S. Son and J. Liebeherr. BeeHive: Global Multimedia Database Support for Dependable, Real-Time Applications. Chapter in Real-Time Databases and Information Systems (Kluwer Academic Publishers, 1997), pp. 409-422.

 ISBN: 978-1-61208-217-2

Multiple Trajectory Search for the Resource-Constrained Project Scheduling Problem

Lin-Yu Tseng
Department of Computer Science and Communication Engineering
Providence University
Taichung, Taiwan
lytseng@pu.edu.tw

Kuan-Cheng Lin
Department of Computer Science and Engineering
National Chung Hsing University
Taichung, Taiwan
kclin@cs.nchu.edu.tw

***Abstract*—The resource-constrained project scheduling problem is one of the most important scheduling problems and has attracted much attention of researchers. In this study, we proposed a new metaheuristic called the multiple trajectory search for solving this problem. The multiple trajectory search algorithm was previously proposed by us to solve the real-parameter optimization problems, both single-objective and multi-objective. And its performance was good as revealed by the ranking in the competitions held in 2008 and 2009 IEEE Congress on Evolutionary Computation. In this study, we arranged the multiple trajectory search algorithm to solve a combinatorial problem – the resource-constrained project scheduling problem. The experimental results show that the proposed method is competitive with other state-of-the-art methods, especially for the problem sets with 30 and 60 activities.**

Keywords- resource-constrained project scheduling problem; multiple trajectory search; peak crossover; forward-backward improvement

I. INTRODUCTION

The resource-constrained project scheduling problem (RCPSP) is an important scheduling problem and many researchers have devoted much effort solving it. Being an NP-hard problem, Alcaraz and Maroto [1] mentioned that the optimal solution could only be achieved by exact solution procedure in small projects, usually with number of activities less than 60 and with the project not highly resource-constrained. Therefore, heuristic and metaheuristic methods were designed to solve large and highly resource-constrained projects.

Some algorithms are exact and are based on the branch-and-bound strategy. Demeulemeester and Herroelen [2] developed a depth-first branching scheme with dominance criteria and the bounding rules. Brucker, Knust, Schoo and Thiele [3] presented a branch-and-bound algorithm whose branching scheme applied a set of conjunctions and disjunctions to pairs of activities.

Some algorithms are heuristic. They are briefly described in the following. Möhring, Schulz, Stork and Uetz [4] proposed a heuristic based on the Lagrangian relaxation and minimum cut computations. The heuristic methods based on priority rules can be divided into two classes: single-pass methods and multi-pass methods. The following studies presented single-pass methods: Boctor [5], Kolisch [6], Valls, Perez and Quintanilla [7], Ulusoy and Özdamar [8], and Özdamar and Ulusoy [9]. Multi-pass methods were presented in Ulusoy and Özdamar [8] and Boctor [5]. The forward-backward methods were proposed in Li and Willis [10] and Özdamar and Ulusoy [9].

Some metaheuristic algorithms were also proposed for RCPSP. They include the genetic algorithm, the simulated annealing, the tabu search, the ant colony optimization, the path relinking, and hybrid algorithms. The following studies incorporated genetic algorithms: Lee and Kim [11], Özdamar [12], Mori and Tseng [13], Alcaraz and Maroto [1], and Hartmann [14][15]. Some simulated annealing algorithms were proposed by Boctor [16], Lee and Kim [11], Cho and Kim [17], and Bouleimen and Lecocq [18]. Methods based on tabu search were presented in Valls, Quintanilla and Ballestin [19] and Artigues, Michelon and Reusser [20]. Merkle, Middendorf and Schmeck [21] presented an ant colony optimization by using the summation of the values in the pheromone set for this problem. Valls, Ballestin and Quintanilla [22] presented a simple technique named justification that can be applied in many methods to improve the quality of solution without generally requiring more computing time. They also designed a peak crossover operator within a hybrid genetic algorithm with justification [23].

The investigation of Hartmann and Kolisch [24] and its updated version conducted an elaborate study on state-of-the-art heuristic and metaheuristic methods. They presented performance comparisons among heuristic and metaheuristic methods in their study by applying these methods to different standard instance sets, namely J30, J60 and J120, generated by ProGen in the PSPLIB [25]. As shown by the latest experimental evaluation, metaheuristic methods outperformed heuristic methods.

In this study, we arranged the multiple trajectory search (MTS) algorithm that was previously proposed for continuous optimization to solve the RCPSP. The results obtained are comparable with the results obtained by other state-of-the-art methods.

The remainder of the paper is organized as follows. In Section II, the definition of the RCPSP is described. In Section III, the proposed MTS for the RCPSP is elaborated.

 ISBN: 978-1-61208-217-2

Section IV gives experimental results and Section V concludes the paper.

II. DEFINITION OF THE PROBLEM

The single-mode RCPSP considers each activity in the project having an operation mode only. All resources available in the project are renewable. The objective is to find a schedule of operation start times for activities, subject to the precedence constraints and resource constraints, such that the makespan of the project is minimized. The notations are defined as follows. The set of activities in the project is denoted by $\{0, 1, \ldots, n, n+1\}$, where *0* and *n+1* are dummy activities and all other activities are non-dummy. The operation start time of activity *j* is denoted by S_j. The duration of activity *j* is denoted by d_j. The set of resources in the project is denoted by $\{1, 2, \ldots, K\}$. The capacity of resource *k* available in each time period during the process of the project is denoted by R_k. The quantity of resource demand of activity *j* to resource *k* is denoted by r_{jk}. $i \rightarrow j$ denotes that activity *i* is a predecessor of activity *j*. P_j denotes the set of all predecessors of activity *j*.

The project can be represented as an activity-on-node network by the precedence relations. Fig. 1 shows an example of a project of this problem. There is only one resource with capacity 5 in this example. d_j / r_{jk} above each node denotes the duration of activity and the demand of activity *j* to resource *k*. The dummy activities which have zero duration and no any resource demand are the single source and sink in the network.

In this study, solutions are represented in the form of a precedence-feasible activity list. When an activity list is given, the serial schedule generation scheme is applied to produce a schedule. We consider both of *forward scheduling* and *backward scheduling*. A *forward schedule* is a mapping, by this mapping, activity *0* to activity *n+1* are mapped to a set of operation start times which are set to be as early as possible while satisfying the resource constraints. A *backward schedule* is also a mapping, by this mapping, activity *n+1* to activity *0* are mapped to a set of operation start times which are set to be as late as possible while satisfying the resource constraints.

III. MULTIPLE TRAJECTORY SEARCH FOR RCPSP

The MTS was previously proposed by us to solve the single-objective real-value optimization problem [26] and the multi-objective real-value optimization problem [27]. In this section, the proposed MTS for the RCPSP is elaborated. The MTS consists of two phases. In the first phase, a set of seeds are found, and in the second phase, a region search method is applied to do the search beginning with each seed. A genetic local search algorithm is used both in the first phase and in the region search. We first introduce the MTS.

A. Multiple Trajectory Search

The MTS algorithm is described in the following.

Step1. Randomly generate the initial population and apply the genetic local search algorithm to find *N* good solutions.

Step2. Using these *N* good solutions as the initial population, apply the genetic local search algorithm again to find *m* seeds.

Step3. For each of these *m* seeds, apply the region search algorithm to it.

Step4. When the maximum number of evaluations is reached, output the best found solution.

Because most studies on the RCPSP used the maximum number of evaluations as the termination condition, this study also adopted it as the termination condition in order to compare the results with those of others methods. The first phase of the MTS consists of Step1 and Step2. One-tenth of the maximum number of evaluations is used in Step1 to globally find good solutions. Another one-tenth of the maximum number of evaluations is used in Step2 to find *m* seeds. Step3 and Step4 are the second phase. In the second phase, we apply the region search algorithm to each seed until the maximum number of evaluations is reached. In the following subsections, we introduce the region search algorithm and the genetic local search algorithm.

B. Region Search Algorithm

The region search begins with a seed. It first generates *k* solutions from the seed by randomly picking *p* activities and re-inserting them into the activity list. These *k* solutions form the initial population of the genetic local search algorithm. Then, it applies the genetic local search algorithm to search for better solutions. If the best solution found by the genetic local search algorithm is better than the original seed, the best solution found replaces the original seed, and the region search begins again with the new seed. If the best solution found is not better than the original seed, the value of *p* is set to 0.8* *p*. The value of 0.8 is determined based on empirical experiences. The region search terminates when the value of *p* reaches one-third of the original value of *p*.

C. Genetic Local Search Algorithm

In the genetic local search algorithm, we use a modified version of the peak crossover operator proposed by Valls, Ballestin and Quintanilla [23]. Also, based on the FBI (forward-backward improvement) [10], we proposed the FBI-WP (forward-backward improvement with perturbation) as the local search scheme. We now describe the genetic local search algorithm in the following.

Step1. The following Step2 to Step4 are executed #_of_generation times.

Step2. Randomly pair all chromosomes in the parent population, then apply modified peak crossover operator to each pair of parents. All child chromosomes are put in the child population.

 ISBN: 978-1-61208-217-2

Step3. Apply FBI-WP to each chromosome in the child population.

Step4. Apply 2-tournament selection to the parent population and the child population to produce the population for the next generation.

Next, let us explain the modified peak crossover operator. Let the parent chromosomes be *F* and *M*. Two resources r_1 and r_2 are randomly selected as the basis for calculating the peak. The average usage avg_r and the maximum usage max_r of the resource *r* are calculated, and a threshold t_r is defined as $t_r = avg_r + (max_r - avg_r)*0.5$. For *F* (or *M*), the time units where the resource usage is greater than this threshold are taken as the peak. Suppose that the crossover of *F* and *M* is to produce *S* and *D*, then *S* preserves the peak of *F* and the other activities come from *M*, and in a similar way, *D* preserves the peak of *M* and the other activities come from *F*.

Forward-backward improvement (FBI) was proposed by Li and Willis [10]. Basically, this scheme iteratively applies serial forward and backward scheduling until no further improvement in the makespan can be obtained. The activity finish times of a forward schedule determine the activity priorities when we produce the next backward schedule. And the activity start times of a backward schedule determine the activity priorities when we produce the next forward schedule. Valls, Ballestin and Quintanilla [22] showed that the FBI is rather effective. In this study, we add a perturbation mechanism to the FBI. If two consecutive applications of the FBI does not improve the solution, we perturb (mutate) the solution before continue to apply the FBI. In the perturbation, we re-insert randomly chosen activities *num_of_mutate* times. The application of the FBI will be terminated only when the number of no improvement reaches a pre-specified value (*Max_No_Improve*).

IV. Experimental Results

The proposed MTS for the RCPSP was implemented in C++ and applied to solve the standard instance sets J30, J60, and J120 generated by the problem generator ProGen devised by Kolisch and Sprecher [25], which were available in PSPLIB [28]. Both J30 and J60 contain 480 instances, and J120 contains 600 instances. Only in J30 the optimal makespans for all instances are known. In J60 and J120 the optimal makespans for some instances are not known and only upper bounds (best known solutions) and lower bounds are provided. The lower bounds are determined by the critical path method. The performance comparison between different methods is conducted by evaluating the same number of schedules, namely, 1000, 5000 and 50000 schedules in order to make it a machine-independent comparison.

The values of parameters were set by some preliminary experiments. These values are shown in Table I. The average deviation from the optimal makespans or the lower bounds and the CPU time used by the MTS are shown in Table II. Tables III to V list the performance comparison of the MTS with other state-of-the-art methods on J30, J60 and J120 respectively. From the last three Tables, the performances of the MTS are ranked the third on J30, the second on J60, and the fifth on J120 when compared with other state-of-the-art methods.

V. Conclusions and Future Research

The MTS was previously proposed for solving continuous optimization, both single-objective and multi-objective. The performance of the MTS was good for the real-parameter optimization. In this study, we arranged the MTS to solve the RCPSP, which is a combinatorial optimization problem. The result obtained is promising, though not excellent.

The combinatorial optimization is quite different from the continuous optimization. How to convert the concepts used in the continuous version of the MTS to fit the characteristics of the combinatorial optimization is the main issue in the future research. The other future research topics are trying to solve other combinatorial optimization problems by applying the MTS.

Acknowledgment

The authors thank National Science Council of the Republic of China, Taiwan (Contract No. NSC98-2221-E126-014-MY3) for partially supporting this research work.

References

[1] J. Alcaraz and C. Maroto, "A robust genetic algorithm for resource allocation in project scheduling," Annals of Operations Research, 2001. **102**: pp. 83-109.

[2] E Demeulemeester and W. Herroelen, "A branch-and-bound procedure for the multiple resource-constrained project scheduling problem," Management Science, 1992. **38**(12): pp. 1803-1818.

[3] P. Brucker, S. Knust, A. Schoo, and O. Thiele, "A branch and bound algorithm for the resource-constrained project scheduling problem," European Journal of Operational Research, 1998. **107**(2): pp. 272-288.

[4] R. Möhring, A. Schulz, F. Stork, and M. Uetz, "Solving project scheduling problems by minimum cut computations," Management Science, 2003. **49**(3): pp. 330-350.

[5] F.F. Boctor, "Some efficient multi-heuristic procedures for resource-constrained project scheduling," European Journal of Operational Research, 1990. **49**(1): p. 3-13.

[6] R. Kolisch, "Efficient priority rules for the resource-constrained project scheduling problem", J. Oper. Manag., vol. 14, pp. 179-192, 1996.

[7] V. Valls, M.A. Perez, and M.S. Quintanilla, "Heuristic performance in large resource-constrained projects", Technical Report, Department of Statistics and Operations Research, Universitat de Valencia, Valencia, Spain, 1992.

[8] G. Ulusoy and L. Ozdamar, "A constraint-based perspective in resource constrained project scheduling," International Journal of Production Research, 1994. **32**(3): pp. 693-705.

[9] L. Ozdamar and G. Ulusoy, "A note on an iterative forward backward scheduling technique with reference to a procedure by Li and Willis," European Journal of Operational Research, 1996. **89**(2): pp. 400-407.

 ISBN: 978-1-61208-217-2

[10] K.Y. Li and R.J. Willis, “An iterative scheduling technique for resource-constrained project scheduling,” European Journal of Operational Research, 1992. **56**(3): pp. 370-379.

[11] J.K. Lee and Y.D. Kim, “Search heuristics for resource constrained project scheduling,” Journal of the Operational Research Society, 1996. **47**(5): pp. 678-689.

[12] L. Ozdamar , “A genetic algorithm approach to a general category project scheduling problem,” Ieee Transactions on Systems Man and Cybernetics Part C - Applications and Reviews, 1999. **29**(1): pp. 44-59.

[13] M. Mori and C.C. Tseng, “A genetic algorithm for multi-mode resource constrained project scheduling problem,” European Journal of Operational Research, 1997. **100**(1): pp. 134-141.

[14] S. Hartmann, “A competitive genetic algorithm for resource-constrained project scheduling,” Naval Research Logistics, 1998. **45**(7): pp. 733-750.

[15] S. Hartmann, “A self-adapting genetic algorithm for project scheduling under resource constraints,” Naval Research Logistics, 2002. **49**(5): pp. 433-448.

[16] F.F. Boctor, “Heuristics for scheduling projects with resource restrictions and several resource-duration modes,” International Journal of Production Research, 1993. **31**(11): pp. 2547-2558.

[17] J.H. Cho and Y.D. Kim, “A simulated annealing algorithm for resource constrained project scheduling problems,” Journal of the Operational Research Society, 1997. **48**(7): pp. 736-744.

[18] K. Bouleimen. and H. Lecocq, “A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version,” European Journal of Operational Research, 2003. **149**(2): pp. 268-281.

[19] V. Valls , S. Quintanilla, and F. Ballestin, “Resource-constrained project scheduling: A critical activity reordering heuristic,” European Journal of Operational Research, 2003. **149**(2): pp. 282-301.

[20] C. Artigues , P. Michelon, and S. Reusser, “Insertion techniques for static and dynamic resource-constrained project scheduling,” European Journal of Operational Research, 2003. **149**(2): pp. 249-267.

[21] D. Merkle , M. Middendorf, and H. Schmeck, “Ant colony optimization for resource-constrained project scheduling,” IEEE Transactions on Evolutionary Computation, 2002. **6**(4): pp. 333-346.

[22] V. Valls, F. Ballestin, and S. Quintanilla, “Justification and RCPSP: A technique that pays,” European Journal of Operational Research, 2005. **165**(2): pp. 375-386.

[23] V. Valls , F. Ballestin, and S. Quintanilla, “A hybrid genetic algorithm for the resource-constrained project scheduling problem.,” European Journal of Operational Research, 2008. **185**(2): pp. 495-508.

[24] S. Hartmann and R. Kolisch, “Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem,” European Journal of Operational Research, 2000. **127**(2): pp. 394-407.

[25] R. Kolisch and A. Sprecher, “PSPLIB - A project scheduling problem library,” European Journal of Operational Research, 1997. **96**(1): pp. 205-216.

[26] L. Y. Tseng and C. Chen, “Multiple Trajectory Search for Large Scale Global Optimization,” Proceedings of the 2008 IEEE Congress on Evolutionary Computation, June 2-6, 2008, Hong Kong.

[27] Lin-Yu Tseng and Chun Chen, “Multiple Trajectory Search for Unconstrained/Constrained Multi-Objective Optimization,” Proceedings of the 2009 IEEE Congress on Evolutionary Computation, May 18-21, Trondheim, Norway.

[28] PROJECT SCHEDULING PROBLEM LIBRARY - PSPLIB, http://129.187.106.231/psplib/

[29] M. Ranjbar and F. Kianfar, “A Hybrid Scatter Search for the RCPSP,” Scientia Iranica Transaction E-Industrial Engineering, 2009. **16**(1): pp. 11-18.

[30] Y. Kochetov and A. Stolyar, “Evolutionary local search with variable neighborhood search for the resource-constrained project scheduling problem”, Proc. of the 3rd International Workshop of Computer Science and Information Technologies, Russia, 2003.

[31] V. Valls, F. Ballestin, and S. Quintanilla, “A population-based approach to the resource-constrained project scheduling problem”, Ann. Oper. Res., vol. 165, pp. 375-386, 2005.

[32] P. Tormos and A. Lova, “A competitive heuristic solution technique for Resource-Constrained Project Scheduling,” Annals of Operations Research, 2001. **102**: pp. 65-81.

[33] K. Nonobe and T. Ibaraki, “Formulation and tabu search algorithm for the resource constrained project scheduling problem”, in *Essays and surveys in metaheuristics*, Ribeiro, CC. and Hansen, P. (Ed.), Kluwer Academic Publishers, pp. 557-588, 2001.

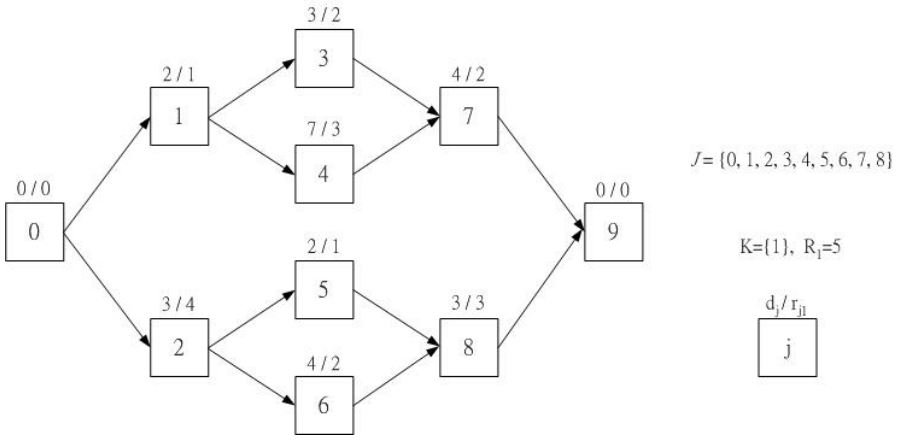


Figure 1. A project example for the RCPSP.

Table I The setting of parameter values

Parameter	**Max_#_sched**	***J*30**	***J*60**	***J*120**
N (size of initial population)	All	20		
m (# of seeds)	All	3		
k (popul. size of region search)	All	12		
Max_No_Improve	All	3		
Number of GLS generations	1000	10		
	5000	15		
	50000	45		
num_of_mutate	All	15	30	60
p (# of activities to be re-inserted)	All	24	30	50

Table II The average deviation achieved and the CPU time spent by the MTS

	max-sch	***J*30**	***J*60**	***J*120**
Avg. Dev.(%)	1000	0.12	11.72	35.81
	5000	0.04	11.05	33.67
	50000	0.01	10.67	32.11
Avg. CPU (seconds)	1000	0.005	0.019	0.089
	5000	0.013	0.080	0.396
	50000	0.054	0.783	3.781

Table III Performance comparison of MTS and other state-of-the-art methods on J30

Author(s)	**Algorithm type**	**Average deviation(%)**		
		1000	**5000**	**50000**
M. Ranjbar and F. Kianfar [29]	HSS, path reli.	0.10	0.03	0.00
Kochetov and Stolyar [30]	GA, TS, path reli.	0.10	0.04	0.00
This study	MTS, GLS, FBI	0.12	0.04	0.01*
Valls et al. [23]	Hybrid GA	0.27	0.06	0.02
Alcaraz and Maroto [1]	GA	0.33	0.12	–
Valls et al. [31]	DJGA	0.34	0.20	0.02
Tormos and Lova [32]	Sampling + BF/FB	0.25	0.13	0.05
Nonobe and Ibaraki [33]	Tabu Search	0.46	0.16	0.05
Tormos and Lova [32]	Sampling + BF	0.30	0.16	0.07
Hartmann [15]	Self-adapting GA	0.38	0.22	0.08

 ISBN: 978-1-61208-217-2

Table IV Performance comparison of MTS and other state-of-the-art methods on J60

Author(s)	Algorithm type	Average deviation(%)		
		1000	5000	50000
M. Ranjbar and F. Kianfar [29]	HSS, path reli.	11.59	11.07	10.64
This study	MTS, GLS, FBI	11.72	11.05	10.67*
Valls et al. [23]	Hybrid GA	11.56	11.10	10.73
Kochetov and Stolyar [30]	GA, TS, path reli.	11.71	11.17	10.74
Valls et al. [31]	DJGA	12.21	11.27	10.74
Hartmann [15]	Self-adapting GA	12.21	11.7	11.21
Hartmann [14]	Activity list GA	12.68	11.89	11.23
Tormos and Lova [32]	Sampling + BF/FB	11.88	11.62	11.36
Tormos and Lova [32]	Sampling + BF	12.14	11.82	11.47
Alcaraz and Maroto [1]	GA	12.57	11.86	–

Table V Performance comparison of MTS and other state-of-the-art methods on J120

Author(s)	Algorithm type	Average deviation(%)		
		1000	5000	50000
Valls et al. [23]	Hybrid GA	34.07	32.54	31.24
M. Ranjbar and F. Kianfar [29]	HSS, path reli.	35.08	33.24	31.49
Valls et al. [31]	DJGA	35.39	33.24	31.58
Kochetov and Stolyar [30]	GA, TS, path reli.	34.74	33.36	32.06
This study	MTS, GLS, FBI	35.81	33.67	32.11*
Valls et al. [31]	DJ, population based, changes operator	35.18	34.02	32.81
Hartmann [15]	Self-adapting GA	37.19	35.39	33.21
Tormos and Lova [32]	Sampling + BF/FB	35.01	34.41	33.71
Merkle et al. [21]	Ant Co. Opt.	–	35.43	–
Hartmann [14]	Activity list GA	39.37	36.74	34.03

 ISBN: 978-1-61208-217-2

Feature Selection for Clustering by Exploring Nearest and Farthest Neighbors

Chien-Hsing Chen
Department of Information Management, Hwa Hsia Institute of Technology
Taipei, Taiwan
ktfive@gmail.com

***Abstract*—Feature selection has been explored extensively for use in several real-world applications. In this paper, we propose a new method to select a salient subset of features from unlabeled data, and the selected features are then adaptively used to identify natural clusters in the cluster analysis. Unlike previous methods that select salient features for clustering, our method does not require a predetermined clustering algorithm to identify salient features, and our method potentially ignores noisy features, allowing improved identification of salient features. Our feature selection method is motivated by a basic characteristic of clustering: a data instance usually belongs to the same cluster as its geometrically nearest neighbors and belongs to a cluster different than those of its geometrically farthest neighbors. In particular, our method uses instance-based learning to quantify features in the context of the nearest and the farthest neighbors of every instance so that clusters generated by the salient features maintain this characteristic.**

Keywords-feature selection; nearest neighbor; farthest neighbor; salient feature; cluster analysis.

I. INTRODUCTION

Feature selection has been explored extensively for use with several real-world applications, such as text processing [1] and image representation [2]. Typically, a larger number of features to represent the patterns is more informative for a learning algorithm. However, in a high-dimensional dataset, some features are noisy, and thus the learning algorithms often suffer from the bias of noisy features that influence the learning process. Recently, many extensive studies have been proposed for feature selection with unsupervised learning, and the selected salient feature subsets were found to aid cluster analysis [3-6]. The goal of feature selection for clustering is to identify a subset of relevant features and remove redundancy from the original representation space. In addition, feature selection is also used to choose selected features to partition a dataset into clusters while effectively increasing both the cluster compactness for the data instances within a cluster and the cluster separability of the data instances between clusters.

In this paper, we propose a new method for selecting a subset of original features from unlabeled data, and the selected feature subset is adaptively used to identify natural clusters in the cluster analysis. The main contribution of this work is that our method achieves the goal of feature selection for clustering without the need to exactly explore the clustered information, thus potentially ignoring the bias of noisy or uninformative features that influence the identification of salient features.

Our method uses instance-based learning for quantifying features in the context of the nearest and the farthest neighbors of every instance. This quantification is motivated by one of the most well-known characteristics of clustering: an instance usually belongs to the same cluster as its geometrically nearest neighbors and belongs to a cluster different than those of its geometrically farthest ones. Therefore, the purpose of our feature selection method is to quantify features so that clusters generated by the salient features (i.e., with higher quantity) maintain this well-known characteristic. Therefore, our method is advantageous because our method does not need to explore natural clusters using a predetermined clustering algorithm.

With our method, a feature is quantified by its ability to distinguish between the nearest and the farthest neighbors of every instance. The quantifying features learning process is to identify the best feature salience vector (represented by a real-valued quantity vector to indicate its salience for this distinguishability) instead of to heuristically search a subset of features in the space of all possible feature subsets. In addition, we implement a gradient descent iterative method employing cooperative and competitive strategies to identify the best feature salience vector.

II. THE PROPOSED METHOD

2.1 Observations of our Proposed Method

We present an example in which we have a set of instances each with two dimensions, "Feature 1" and "Feature 2" (Fig. 1), and all instances can be clustered into two assumptive clusters, "Cluster 1" and "Cluster 2." For this example, we discuss our presented feature selection method to extract salient features which are adaptively used for discovering natural clusters.

We briefly introduce our method for determining a salient feature. This method begins with the instance "$\mathbf{x}_1$", which has both specific nearest and farthest neighbors (see Figure 1). We define feature separability as the average distance from an instance to its farthest neighbors (its magnitude is represented as a dotted line) and the feature compactness as the average distance from this instance to its nearest neighbors (its magnitude is represented as a solid line), where both distances are measured with respect to a particular feature. We assume

 ISBN: 978-1-61208-217-2

that a feature is more salient if it yields a higher value of the following measure:

$$\frac{\#\{\text{average length of corresponding to the separability}\}}{\#\{\text{average length of corresponding to the compactness}\}}$$

Therefore, based on this assumption, "Feature 1" is more salient. In the context of clustering, for which we want to partition the dataset into clusters, we would be much more likely to believe that the contribution of "Feature 1" is higher than that of "Feature 2."

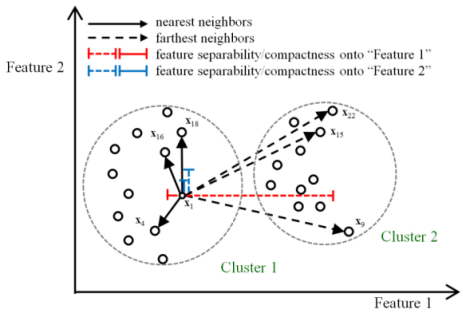


Figure 1. A schematic example: an instance "$\mathbf{x}_1$" has its nearest neighbors {"$\mathbf{x}_{16}$", "$\mathbf{x}_{18}$", "$\mathbf{x}_4$"} and its farthest neighbors {"$\mathbf{x}_9$", "$\mathbf{x}_{22}$", "$\mathbf{x}_{15}$"}. For the instance "$\mathbf{x}_1$", "Feature 1" should be more salient than "Feature 2".

2.2 Preliminaries of our Proposed Method

In this paper, we present a new unsupervised feature selection method that does not require any clustering learning algorithm to identify salient features; the salient features selected by our method can be then effective for clustering. Our method is based on a basic characteristic of clustering: an instance usually belongs to the same cluster as do its nearest neighbors and belongs to a cluster different than those of its farthest neighbors. With our method, a feature is quantified by its ability to distinguish between the nearest and the farthest neighbors of every instance.

Assume a dataset X of n data instances ($\boldsymbol{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$), where $\mathbf{x}_i = [x_{1,i}, \ldots, x_{j,i}, \ldots, x_{d,i}]^{\mathrm{T}}$ represents the i^{th} instance in $\boldsymbol{X}$ with d dimensions; also assume a non-zero feature salience vector $\mathbf{w}(t) = [w(t)_1, \ldots, w(t)_j, \ldots, w(t)_d]^{\mathrm{T}}$, where the element $w(t)_j$ is a real-valued quantity at the t^{th} iteration. We first consider $\mathbf{w}(t)$ to obtain the nearest and the farthest neighbors for a given instance. The k^{th} nearest neighbor $\mathbf{x}^{\Theta}_{i\to k,\mathbf{w}(t)}$ and the l^{th} farthest neighbor $\mathbf{x}^{\phi}_{i\to l,\mathbf{w}(t)}$ of $\mathbf{x}_i$ are subject to, respectively,

$$\pi(k) = \sum_{r=1, r\neq i}^{n} I\Big(dist\big(\mathbf{x}_i, \mathbf{x}_r|\mathbf{w}(t)\big) \leq dist\Big(\mathbf{x}_i, \mathbf{x}^{\Theta}_{i\to k;\,\mathbf{w}(t)}|\mathbf{w}(t)\Big)\Big) \quad (1)$$

$$\pi(l) = \sum_{r=1, r\neq i}^{n} I\Big(dist\big(\mathbf{x}_i, \mathbf{x}_r|\mathbf{w}(t)\big) \geq dist\Big(\mathbf{x}_i, \mathbf{x}^{\phi}_{i\to k;\,\mathbf{w}(t)}|\mathbf{w}(t)\Big)\Big) \quad (2)$$

where $\pi()$ transfers an ordinal number to an interval number that represents the number of instances that satisfy the whole condition. Θ represents a nearest neighbor and ϕ represents a farthest neighbor. $I()$ outputs 1 when the condition is satisfied and outputs zero otherwise. $dist(\mathbf{x}_a,\mathbf{x}_b|\mathbf{w}(t))$ is a distance function in which we use the weighted Euclidean metric to measure the distance between $\mathbf{x}_a$ and $\mathbf{x}_b$ under $\mathbf{w}(t)$.

$$dist(\mathbf{x}_a, \mathbf{x}_b \mid \mathbf{w}(t)) = \sqrt{\sum_{j=1}^{d} w(t)_j \times \big(x_{j,a} - x_{j,b}\big)^2} \quad (3)$$

2.2.1 Measure Manhattan Distance with Element-Wise Absolute Operator

We then scale each feature and define two sets of distances from $\mathbf{x}_i$ to its K nearest neighbors and L farthest neighbors. Then, we obtain two new sets of instances, $\mathbf{NS}^{\mathbf{w}(t)}_{i,K}$ and $\mathbf{FS}^{\mathbf{w}(t)}_{i,L}$, which include K and L instances, respectively.

$$\mathbf{NS}^{\mathbf{w}(t)}_{i,K} = \{d\big(\mathbf{x}_i, \mathbf{x}^{\Theta}_{i\to 1;\,\mathbf{w}(t)}\big), \ldots, d\big(\mathbf{x}_i, \mathbf{x}^{\Theta}_{i\to K;\,\mathbf{w}(t)}\big)\} \quad (4)$$

$$\mathbf{FS}^{\mathbf{w}(t)}_{i,L} = \{d(\mathbf{x}_i, \mathbf{x}^{\phi}_{i\to 1;\,\mathbf{w}(t)}), \ldots, d(\mathbf{x}_i, \mathbf{x}^{\phi}_{i\to L;\,\mathbf{w}(t)})\} \quad (5)$$

where $d(.,.)$ is an element-wise absolute operator, thus yielding d-dimensional data.

Let us assume that two sets of neighbors can be distinguished maximally by a subset of features. Therefore, the next step is to extract the salient features. We assign two labels, one for the nearest neighbors and the other for the farthest neighbors. The idea is that our method scales each feature and measures which feature can better achieve separability between the instances in $\mathbf{NS}^{\mathbf{w}(t)}_{i,K}$ and $\mathbf{FS}^{\mathbf{w}(t)}_{i,L}$. First, we define a data fraction $\boldsymbol{\rho}_i(\mathbf{w}(t))$ that includes the instances in $\mathbf{NS}^{\mathbf{w}(t)}_{i,K}$ and $\mathbf{FS}^{\mathbf{w}(t)}_{i,L}$ for a given $\mathbf{x}_i$ under $\mathbf{w}(t)$. The fraction $\boldsymbol{\rho}_i(\mathbf{w}(t))$ is expressed as

$$\boldsymbol{\rho}_i(\mathbf{w}(t)) = \left\{\mathbf{NS}^{\mathbf{w}(t)}_{i,K}, \mathbf{FS}^{\mathbf{w}(t)}_{i,L}\right\} \quad (6)$$

This fraction consists of $K + L$ data instances with d dimensions (i.e., $s_1, \ldots, s_j, \ldots, s_d$, where s_j is the j^{th} variable in

 ISBN: 978-1-61208-217-2

$\boldsymbol{\rho}_i(\mathbf{w}(t))$). Next, we assign a categorical variable c to represent labels for these instances. The label c_r for an instance $\mathbf{x}_r \in \boldsymbol{\rho}_i(\mathbf{w}(t))$ is assigned by

$$c_r = \begin{cases} 0, \text{if } \mathbf{x}_r \in \mathbf{NS}_{i,K}^{\mathbf{w}(t)} \\ 1, \text{if } \mathbf{x}_r \in \mathbf{FS}_{i,L}^{\mathbf{w}(t)} \end{cases} \quad (7)$$

Our method is an unsupervised feature selection method; therefore, the class labels are not considered by the process of feature selection. Fortunately, while the new labels are assigned using Eq. (7), the filter-based and wrapper-based feature selection methods in supervised learning can be used to help our method identify salient features. For example, we can use a filter-based feature selection method (e.g., mutual information) to evaluate how an individual feature informs the target variable [7-9] or apply a wrapper-based feature selection method (e.g., by using SVM to construct a classifier) to observe how a feature better distinguishes between the instances in $\mathbf{NS}_{i,K}^{\mathbf{w}(t)}$ and $\mathbf{FS}_{i,L}^{\mathbf{w}(t)}$ using this classifier [9-12].

2.2.2 Evaluate salient feature using dependency and redundancy metrics

In this paper, we use the filter-based feature selection method to evaluate features because of its efficiency. Thus, we avoid the process of training instances required by the wrapper-based method. The basis of the method to evaluate feature salience is to determine whether a feature is able to distinguish the nearest and the farthest neighbors. In particular, a feature that is more dependent on the target variable and is less redundant with other features is more salient [8]. The criterion to quantify a feature is

$$\Phi_j = D(s_j, c) - R(s_j) \quad (8)$$

in which we use the $D()$ and $R()$ criteria to evaluate dependency and redundancy, respectively. The functions $D()$ and $R()$ are respectively expressed as

$$D(s_j, c) = I(s_j; c) \quad (9)$$

$$R(s_j) = \frac{1}{d-1} \sum_{u=1, u \neq j}^{d} I(s_j; s_{u \neq j}) \quad (10)$$

where $I()$ is a mutual information criterion; other standard criteria can also be used. Because we have a data fraction $\boldsymbol{\rho}_i(\mathbf{w}(t))$ and a categorical variable c, we can obtain a quantification vector $\mathbf{u}_i^{\mathbf{w}(t)} = [u_{1,i}^{\mathbf{w}(t)}, \ldots, u_{j,i}^{\mathbf{w}(t)}, \ldots, u_{d,i}^{\mathbf{w}(t)}]^{\mathrm{T}}$, where $u_{j,i}^{\mathbf{w}(t)}$ represents a quantity measured by Φ_j.

2.3 Searching the Best Feature Salience Vector

In this section, we attempt to search the best feature salience vector $\mathbf{w}$ for which the goal is to satisfy the condition that the $\{\mathbf{u}_i^{\mathbf{w}}\}_{i=1}^{n}$ values are constant for the particular instances $\{\mathbf{x}_i\}_{i=1}^{n} \in X$. Recall that $\mathbf{w}(t)$ is used to find $\mathbf{NS}_{i,K}^{\mathbf{w}(t)}$ and $\mathbf{FS}_{i,L}^{\mathbf{w}(t)}$ applied to evaluate quantification vectors $\{\mathbf{u}_i^{\mathbf{w}(t)}\}_{i=1}^{n}$, which are further applied to reflect $\mathbf{w}(t)$. The criterion to determine the best $\mathbf{w}$ is an NP problem.

We reduce the searching problem to optimize the sum-of-squared error (i.e., $||\mathbf{u}_t^{\mathbf{w}} - \mathbf{w}(t)||^2$). The goal of the learning task is to identify the best $\mathbf{w}$ such that the error is minimized. Therefore, we should use a method will optimize both $\mathbf{u}_t^{\mathbf{w}}$ and $\mathbf{w}(t)$.

We implement a gradient-descent-based approach, Least-Mean-Squares (LMS) [13, 14] with some modifications, and used cooperative and competitive iterative strategies to find $\boldsymbol{W} = [\mathbf{w}(1), \ldots, \mathbf{w}(t), \ldots, \mathbf{w}(T)]^T$. Each iteration t has only one instance $\mathbf{x}_t$ that was randomly selected and participated in learning. Cooperatively, $\mathbf{x}_t$ considers all elements in $\mathbf{w}(t)$ to yield $\mathbf{u}_t^{\mathbf{w}(t)}$. Competitively, $\mathbf{u}_t^{\mathbf{w}(t)}$ is used to perceive the more salient feature and thus inform $\mathbf{w}(t+1)$. Furthermore, $\alpha(t)$ is a monotonically decreasing learning coefficient, so the updated function for $\mathbf{w}(t+1)$ was adaptively written as follows (11). Iterations then stop when $\mathbf{u}_T^{\mathbf{w}}$ and $\mathbf{w}(T)$ are balanced.

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \alpha(t) \times \left[\mathbf{u}_t^{\mathbf{w}(t)} - \mathbf{w}(t)\right] \quad (11)$$

III. EXPERIMENT

This section presents evaluations of our presented method for feature selection on clustering problems. To demonstrate the effectiveness of the selected feature subset for clustering, the usefulness is evaluated in the selected feature subset for several well-known clustering algorithms.

3.1 Parameter Setup

We set the parameters for our method. Assume that we have a dataset including a total of N instances. We first set an initial learning rate $\alpha(1)$ to 0.8 and decrease the learning rate $\alpha(t) = \alpha(1) \times [(T\text{-}t) / T]$ at the t^{th} iteration. The weight of every element in the initial feature salience vector $\mathbf{w}(1)$ is set to be the same. The number of iterations T should be large, such that most instances can be randomly selected for training on the algorithm. We thus set T to $10N$, where N is the size of the training dataset. The parameters K and L are set to $v = \{10, 20, \ldots, 100\}$ which depends on the training instances and will be discussed in later experiments.

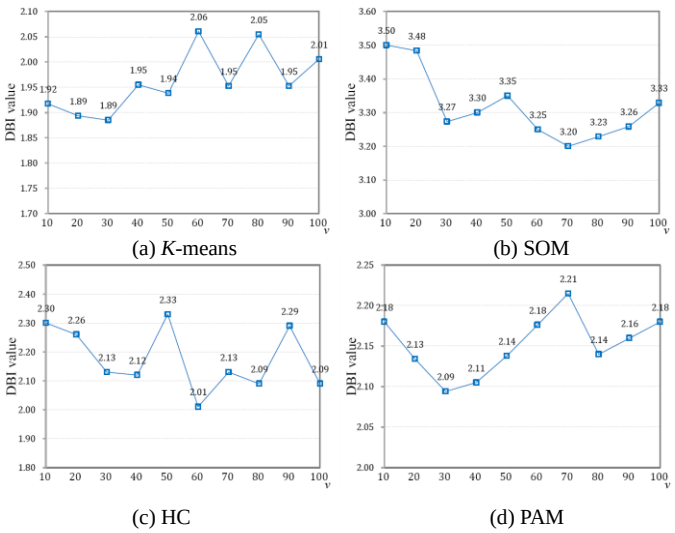
We used a dataset, OT, is mentioned in the studies [4, 15]. We followed the study [4] to randomly select 100 instances for each category and to capture features for every selected image.

3.2 Parameter Analysis

We observed how the parameter v affected the performance of our method because different size of v may produce differently salient features. Here, we discuss the performance of our presented methods for performing clustering (i.e., using K-means, SOM, HC and PAM) in the OT dataset. Each clustering algorithm partitioned the OT dataset into M clusters, where M was set according to the number of class labels (e.g.,

M = 8). With SOM, we followed the study [16] to obtain the user-defined number of clusters for the SOM units using K-means because similar units could be grouped. For the evaluation, we used the Davies-Bouldin index (DBI) [17] to measure the performance due to its popularity in cluster analysis. Lower DBI values represent higher compactness for the instances within a cluster, or higher separability for the instances between clusters. The comparison results of DBI values for various v for performing clustering in the OT dataset are shown in Figure 2.

In Figure 2, we can see that the size of v for performing K-means should be smaller so as to improve the performance. Specifically, the size of v for performing SOM and HC should be respectively set to 70 and 60, while the size of v in PAM should be 30. Therefore, we can set v to 30 rather than 40 for future analysis because v depends on the cost of searching the nearest and the farthest neighbors for every instance.



(a) K-means (b) SOM

(c) HC (d) PAM

Figure 2. Comparison of DBI values for various v in the OT dataset. The salient features were selected by our method and were used to perform clustering using (a) K-means, (b) SOM, (c) HC and (d) PAM algorithms. The parameter v = {10, 20, …, 100} was set to observe the corresponding performance.

IV. CONCLUSIONS

We present a new feature selection method that uses instance-based learning for quantifying features in the context of the nearest and the farthest neighbors of every instance. Such a method is advantageous because our method does not need to explore natural clusters using a predetermined clustering algorithm. Our comprehensive experiment on a synthetic dataset demonstrates that the salient features can be extracted effectively. Because the cluster analysis for very high data dimensionality has been in high demand, we expect that our work will generate broad interest in many research fields.

ACKNOWLEDGMENTS

The author would like to thank the reviewers for their valuable suggestions. This research was supported by National Science Council, Taiwan under grant NSC 99-2410-H-146-001-MY2.

REFERENCE

[1] Manning, C. and Schutze, H., *Foundations of statistical natural language processing*: MIT Press, 1999.

[2] Swets, D. L. and Weng, J. J., "Efficient content-based image retrieval using automatic feature selection," in *Proceedings of 1995 IEEE International Symposium on Computer Vision* 1995, pp. 85-90.

[3] Law, M. H. C., Figueiredo, M. A. T., and Jain, A. K., "Simultaneous feature selection and clustering using mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26(9), pp. 1154-1166, 2004.

[4] Boutemedjet, S., Bouguila, N., and Ziou, D., "A hybrid feature extraction selection approach for high-dimensional non-Gaussian data clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 3(8), pp. 1429-1443, 2009.

[5] Sanguinetti, G., "Dimensionality reduction of clustered data sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30(3), pp. 535-540, 2008.

[6] Dy, J. G. and Brodley, C. E., "Feature selection for unsupervised learning," *Journal of Machine Learning Research*, vol. 5, pp. 845-889, 2004.

[7] Chow, T. W. S. and Huang, D., "Estimating optimal feature subsets using efficient estimation of high-dimensional mutual information," *IEEE Transactions on Neural Networks*, vol. 16(1), 2005.

[8] Peng, H., Long, F., and Ding, C., "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27(8), pp. 1226-1238, 2005.

[9] Kwak, N. and Choi, C. H., "Input feature selection for classification problems," *IEEE Transactions on Neural Networks*, vol. 13(1), pp. 143-159, 2002.

[10] Pena, J. M. and Nilsson, R., "On the complexity of discrete feature selection for optimal classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32(8), pp. 1517-1522, 2010.

[11] Pal, M. and Foody, G. M., "Feature selection for classification of hyperspectral data by SVM," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48(5), pp. 2297-2307, 2010.

[12] Yang, J. B., Shen, K. Q., Ong, C. J., and Li, X. P., "Feature selection for MLP neural network: The use of random permutation of probabilistic outputs," *IEEE Transactions on Neural Networks*, vol. 20(12), pp. 1911-1922, 2009.

[13] Haykin, S. S. and Widrow, B., *Least-mean-square adaptive filters*: Wiley, 2003.

[14] Macchi, O., *Adaptive processing: The LMS approach with applications in transmission*: New York: Wiley, 1995.

[15] Oliva, A. and Torralba, A., "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, pp. 145-175, 2001.

[16] Vesanto, J. and Alhoniemi, E., "Clustering of the self-organizing map," *IEEE Transactions on Neural Networks*, vol. 11(3), pp. 586-600, 2000.

[17] Davies, D. L. and Bouldin, D. W., "A cluster separation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1(4), pp. 224–227, 1979.

 ISBN: 978-1-61208-217-2

Fuzzy Computer Architecture Based on Memristor Circuits

Martin Klimo
Faculty of Management Science and Informatics
University of Zilina
Zilina, Slovakia
martin.klimo@fri.uniza.sk

Ondrej Such
Mathematical Institute
Slovak Academy of Sciences
Banska Bystrica, Slovakia
ondrej.such@gmail.com

***Abstract*— von Neumann computer concept is approaching its limits and new computing paradigms are needed. This paper introduces a fuzzy computer architecture based on memristor circuits. The new concept offers naturally parallel architecture that unifies memory and data processing. We discuss both application specific and general purpose computing. We conclude with an outline of a research roadmap to bring the concept into practical realization.**

Keywords - unconventional computing; fuzzy logic implementation; memristor.

I. INTRODUCTION

In May 2011, FET Proactive Unit of DG-INFSO EC has published a document [1] that presents challenges for 2011-12 period as a result of a broad discussion within ICT community. The idea threading through several essays in this brochure is that von Neumann computer concept is reaching its limits and new computing paradigms are needed. When a program is stored in memory, its instructions need to be transferred, along with the data, into the CPU for execution. This introduces performance barriers: firstly, disjoining processing from memory and secondly, inherent serialization of program execution. These problems motivate research into new computing concepts.

Promising solutions seem to be the quantum computing and biomolecular/biocellular computing. Quantum computing is well covered by FET projects, and, even in May 2011, there was announced a sale of the first commercial quantum computer [2]. In 2010, the BACTOCOM FET Proactive project [3] was also launched focussing on bio-computer prototyping based on population of E. coli bacteria and following in 2011 by COBRA FET Proactive Coordination Action [4]. Although these technologies are promising, they are not mature enough.

Nanotechnology promises a new way to overcome performance limitations mentioned above. In 2008, the HP Lab realised a memristor [5], a two terminal circuit element. forecast by Leon Chua in 1971 [6]. This element is able to change its resistance between high and low resistance states. It does so under bias and in effect it "remembers" charge that has flown through it. Moreover, the change is non-volatile, meaning no energy is needed for memristor to maintain its state. Nanoscale has proven crucial for the functioning of the device since in this realm enormous electric fields can be realized with low voltage. . Energy savings and higher densities can be obtained by non-volatile memristor memories announced for market in 2013 [7]. They can also bridge the distance between processing unit and the memory [8]. But, memristor crossbar can unify the memory and the processing unit together. Nowadays we can find two approaches how to do this. The first idea is to use memristors in the role of a synapse within a neural network. This approach is studied by ERC Starting Grant NANOBRAIN [9] since 2010. The second idea [10] is based on the fact that Boolean implication can by executed by material implication, where the logic value is expressed by the memristance and not by the voltage. This allows one to compute Boolean logic functions [11] by memristor crossbar.

In the rest of our paper we present our new approach based on fuzzy logic circuits instead of binary logic circuits. Memristor implementation of fuzzy logic functions not only brings energy savings, but it naturally joins computation and memory functions and thus creates brain-like computing architecture.

In the next section we review the basic building blocks for fuzzy logic circuits. In section three we describe an application-specific fuzzy architecture that parallels currently employed fuzzy controllers. Section four describes a general-purpose fuzzy logic computer whose main distinguishing feature is reconfigurability. We conclude we propose a research roadmap that would bring the new architecture into reality.

II. FUZZY LOGIC IMPLEMENTATION BY MEMRISTORS

We have found [16] that elementary circuits with memristors can compute min (Figure 1), max, avrg functions and in synchronised mode (similar to [10]) the implication as well in voltage domain.

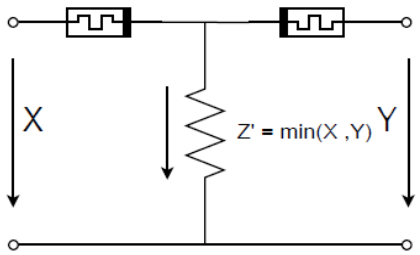


Figure 1. Example of a fuzzy function implementation.

 ISBN: 978-1-61208-217-2

Hardware implementation of min, max functions is not a new idea and fuzzy logic circuits based on CMOS, FET [17] [18] or FPGA [19] implementations have been used. However, there are two principal advantages of memristor based implementations:

1. Circuit uses only energy of its inputs and no extra source of energy is needed compared with the transistor based implementations mentioned above.
2. Memristor implementation introduces also a memory feature of these elementary circuits. This property is based on dynamic behaviour of memristors and it needs further research.

These three functions (min, max, avg) allow only building of a monotone fuzzy logic system [12]. Using clock pulses for implication or using CMOS invertors for negation, the Gődel logic can be implemented. This has a significant impact on the fuzzy computer architecture and all possibilities have to be investigated within the project for which we are looking for partners. Here we present an asynchronous architecture using the property of de Morgan law that the dual logic function with inverted inputs gives the inverted function. This dual system approach allows simpler implementation because invertors are located only in the first stage of min, max, avg based circuit. This means that energy source is needed only at the border of the fuzzy logic circuit and the rest of circuit is passive and voltage values are continuously recalculated as fuzzy logic functions when nonzero voltage inputs occur.

We can conclude that memristor-based elementary fuzzy functions with dual system approach allow building general problem solving machines like today's digital computers.

III. APPLICATION SPECIFIC FUZZY COMPUTERS

Thirty years of fuzzy systems applications show a big market potential of application specific fuzzy logic circuits. Capability of high density integration makes them suitable for control functions in daily life in applications like phones, cameras, etc. A general architecture of such a fuzzy computer is well known fuzzy system architecture, as shown in Figure 2.

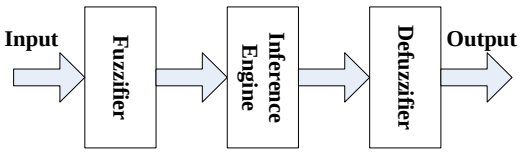


Figure 2. Application specific fuzzy system architecture.

Min, max, avg functions can be applied in all of blocs in the fuzzy system; anyhow, we will concentrate on **Inference engine,** which is a fuzzy logic circuit itself.

Although fuzzy logic applications are often times ahead of theory development, today's fuzzy logic stands on sound mathematical foundations that are continuously being extended. Fuzzy logic allows one to systematically approach solving problems dealing with „uncertainty", in much the same way as two-valued logic captures manipulation of „certain" information. This is the area where human are still order of magnitudes faster than currently used computers. However, should fuzzy logic systems approach complexity of human brain, they cannot be emulated on a von-Neumannn computer. In their implementation, one needs to use blocks that can *naturally* execute elementary fuzzy logic functions and allows them to connect them into larger circuits.

One of the main tasks in a fuzzy system design is optimization of fuzzy logic circuit for a particular application. In a small system the set of implications can be explicitly proposed, but no optimization method is available for large fuzzy circuits. Promising results give metaheuristics like evolution programming, genetic programming, fractal structures and so on.

Creating fuzzy logic functions by memristor network topology evokes reminiscences to analog computing. Comparing memristor based computing with quantum or biocellular computing the memristor technology is more mature and several feasible approaches (metal-oxid, ferro-magnetic, grapheme-oxide) for implementation exist.

Min, max functions in Gődel logic is just one example of t-norm and co-norm in fuzzy logic. On the other side they are examples of fuzzy logic functions that are easily implemented by memristors. This gives rise to the question about position of Gődel logic within the general framework of fuzzy logics for applications.

IV. GENERAL PURPOSE FUZZY COMPUTERS

A general-purpose fuzzy computer requires means to reconfigure its circuits, because, like in the analog computer, the program is given by the circuit topology

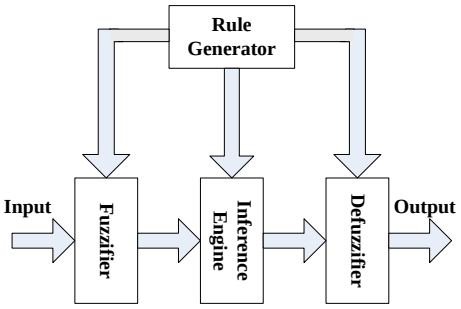


Figure 3. General purpose fuzzy system architecture.

and the initial state of the system. A rule generator, possibly controlled by a CPU/GPU computer, reconfigures memristor switches in fuzzifier, inference engine and defuzzifier as shown in Figure 3.

Then programming, compiling and program execution is a process in which topology of fuzzy logic circuit is designed and parameters of fuzzifier and defuzzifier are specified. In this phase fuzzy computing keeps continuity with classical computing where programs are written by a human expressing given logic requirements. This approach can build on previous works on the fuzzy logic programming [13] and Fuzzy Prolog-like program languages [14] or domain specific languages.

 ISBN: 978-1-61208-217-2

This means that programming can be executed on a classical computer with interfaces into a general structure of reconfigurable memristor network. Memristor crossbar seems be a suitable element to do this, but many questions on the implementation are still open.

As a starting point the following fuzzy computer architecture (shown in Figure 4) may be assumed.

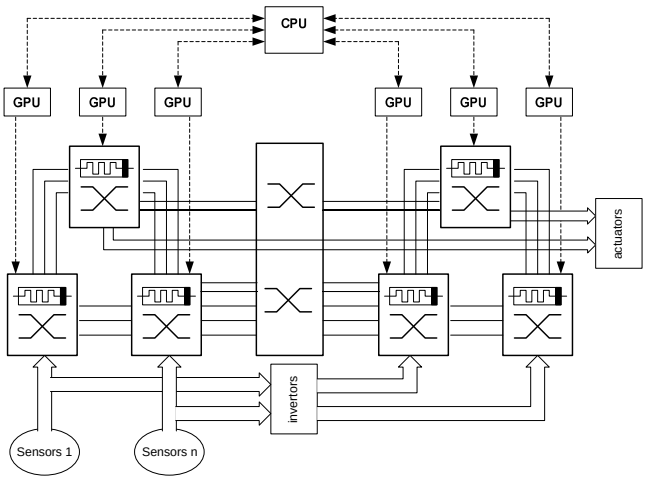


Figure 4. Fuzzy computer architecture, processing sensor inputs on memristor crossbars and yielding actuators

This is two-layer architecture. At the bottom there are memristor crossbars for real time processing with input-output interfaces from sensors – into actuators, as well as invertors for the dual system. We can see dual logic architectures consisting of two "hemispheres". If more crossbars are required to create one hemisphere than may be given by capacity constraints, sense domain separation or hierarchy needs [15], then additional switches may be needed. Also a switch that will connect the corresponding functions in both hemispheres ("Corpus Callosum") belongs to this layer. At the top layer there are classical computers (CPU/GPU) that generate topology of memristor network within crossbars and switches regarding required fuzzy logic functions given by logic program. This topology may be fixed; anyhow we assume that local rules within one crossbar can be adapted by program in corresponding GPU (short-term plasticity) while the long-term strategies (like supervised learning) will be distributed from the supervising CPU on the top.

V. CONCLUSION

Fuzzy logic has been applied in practice for several decades. Memristor technology, despite being just a few years old, is progressing rapidly, and first memristor memories will be commercialized within a year. In order to combine them, and realize aforementioned fuzzy logic computers, a natural research roadmap can be outlined:

1. Adaption of fuzzy logic system to memristor technology
   a. suitability of Gődel logic from applications point of view
   b. evaluation of different memristor technologies (unipolar, bipolar, metal oxide, ferro-magnetic, etc)
   c. theoretical limits of fuzzy logic memristor circuits
   d. influence of memristor threshold effect on fuzzy computation
   e. composition of logic functions
   f. properties of self-referencing fuzzy logic functions (delay and synchronisation)
2. Fuzzy Prolog like programming languages and corresponding compilers have to be developed to allow programmers write programs in digital computer style to create structures in fuzzy logic networks
3. Applications development
   a. simple programs for architecture benchmarking
   b. evolutionary and genetic programs for more complex brain-like applications

ACKNOWLEDGMENT

This contribution is the result of the project implementation: Centre of excellence for systems and services of intelligent transport II., ITMS 26220120050 supported by the Research & Development Operational Programme funded by the ERDF. Partially supported by grant VEGA 2/0112/11.

REFERENCES

[1] FET Consultations 2009-2010 "Shaping the future". European Union, May 2011, ISBN 978-92-79-19570-9, doi:20.2759/4612

[2] M. Feldman, "D-Wave Sells First Quantum Computer," HPC wire (http://www.hpcwire.com/), May 26, 2011, [retrieved: May, 2012]

[3] http://www.bactocom.eu/, [retrieved: May, 2012]

[4] http://www.cobra-project.eu/, [retrieved: May, 2012]

[5] D. B. Strukov, G. S. Snider,D. R. Stewart,R. S. Wiliams, "The missing memristor found," Nature 453 (2008) 80-83

[6] L. O. Chua, "Memristor – the missing circuit element," IEEE Trans. Circuit Theory 18 (1971) pp .507-519

[7] S. Adee, "HP partners with Hynix to commercialize memristors in an all-purpose memory," IEEE Spectrum, 2010, http://spectrum.ieee.org/semiconductors/devices/memristor-inside, [retrieved: May, 2012]

[8] G. Snider, R. Amerson, D. Carter, H. Abdalla, M. S. Q. J. Léveillé, M. Versace, H. Ames, S. Patrick, B. Chandler, A. Gorchetchnikov, E. Mingolla, "From Synapses to Circuitry: Using Memristive Memory to Explore the Electronic Brain," Computer, vol. 44 No. 2 (2011) pp. 21-28

[9] http://cordis.europa.eu/fetch?CALLER=FP7_PROJ_EN&ACTION=D&DOC=1&CAT=PROJ&RCN=95287, [retrieved: May, 2012]

[10] J. Borghetti, G. S. Snider, P. J. Kuekes, J. J. Yang, D. R. Stewart R. S. Williams, "Memristive" switches enable "sateful" logic operations via material implications, Nature 464 (2010) pp. 873-876

 ISBN: 978-1-61208-217-2

[11] E. Lehtonen, J. H. Poikonen, M. Laiho, "Two memristors suffice to compute all Boolean functions," Electronic Letters 46, No. 3 (2010)

[12] M. Blum, R. Rue, K. Yang, "On the complexity of MAX/MIN/AVRG circuit," Carnegie Mellon Technical Report CMU-CS-02-110, 2002, http://repository.cmu.edu/compsci/150/, [retrieved: May, 2012]

[13] P. Vojtáš, "Fuzzy logic programming. Fuzzy Sets and Systems," 124 (2001) pp. 361–370

[14] S. Guadarrama, S. Muñoz, C. Vaucheret, **"**Fuzzy Prolog: a new approach using soft constraints propagation," Fuzzy Sets and Systems. 144, Issue 1 (2004) pp. 127-150

[15] R. Velik, G. Zucker, D. E. Dietrich, "Towards Automation 2.0: A Neurocognitive Model for Environment Recognition," Decision-Making, and Action Execution," EURASIP Journal on Embedded Systems, Vol. 2011, Article No. 4, doi>10.1155/2011/707410

[16] M. Klimo, O. Such, O.: Memristors can implement fuzzy logic. http://arxiv.org/abs/1110.2074

[17] T. Yamakawa, I Miki, "The Current Mode Fuzzy Logic Integrated Circuits Fabricated by the Standard CMOS Process," IEEE Trans. Computers Vol. 35, No. 2 (1986) pp. 161-167

[18] F. Ueno, Y. Shirai, "Fuzzy Logic Semifinished Integrated Circuit," US Patent No. 4,860,243 (1989)

[19] D. Kim, "An Implementation of Fuzzy Logic Controller on the Reconfigurable FPGA System," IEEE Trans. Industrial Electronics, Vol. 47, No. 3 (2000) pp. 703-715

 ISBN: 978-1-61208-217-2

Simulation of Carry Protocol (c-protocol) for MANET Network

Ahmed Alghamdi, John DeDourek, Przemyslaw Pochec
Faculty of Computer Science
University of New Brunswick
Fredericton, Canada
E-mail: a.alghamdi, dedourek, pochec@unb.ca

***Abstract*— The current protocols used in Mobile Ad Hoc Networks require an end-to-end connected path between source and destination to transmit data. We present a group of protocols (c-protocols) that address the issue of dropping packets each time the end-to-end path breaks. In the c-protocol, rather than being dropped, the packets are allowed to be carried at any node for a time until the carrying node gets connected to another node and retransmits the packets. The simulation shows that the c-protocol allows data to be transmitted and delivered even if the end-to-end connection never occurs. The GPS enabled versions of the c-protocol produced the highest throughput in the simulated network.**

Keywords- store-forward protocls, disturbance–tolerant network, MANET, network simulation.

I. INTRODUCTION

A network is a number of computers or devices that are connected to each other through physical or wireless links. The main advantage of having networks is to be able to share resources and services among the connected devices. In order to have sufficient and beneficial communications, the data transfer between the connected devices is governed by a protocol (a communication rule). There are a large number of protocols that are used in networking. Each protocol has its own services and operates over specific types of network. Wired networks have specific protocols or specific versions of a protocol that would not necessarily work efficiently over wireless networks. These protocols have to be modified or changed in order to serve different types of networks.

With the presence of the Internet, having a home or an office network connected to the world is relatively a simple and easy task, especially these days with the huge improvement in the Internet services. However, this is not the case in every network. There are some networks that have to operate in extreme environments. These types of networks have to be treated in different ways to achieve the most efficient type of communication. Examples of such networks are Terrestrial Mobile Networks, Exotic Media Networks, Military Ad-hoc Networks and Sensor/Actuator Networks [1] [2]. In Terrestrial Mobile Networks, the mobility of the nodes and the change in the strength of the signal may cause the network to be partitioned. Due to the large distance in the case of the Exotic Media Networks, there is a high latency in delivering a message. The Military Ad-Hoc Networks usually operate in hostile areas where mobility, danger and other environmental factors would cause the disconnections to repeatedly occur in the network. Generally, there are different types of problems that occur in such networks. For example, high latency and low data rate are common problems that need to be dealt with in order to have a satisfactory connection. Another problem is the repeated disconnections that prevent the network from having normal end-to-end connections which lower the data transmission rate to unacceptable levels. These problems could be solved by some special architectures like MANETs (Mobile Ad Hoc Networks) or DTNs (Disturbance-Tolerant Networks).

In order to exchange messages or packets between any two nodes, the existing MANET networks require an end-to-end direct path; otherwise messages are dropped when the connection breaks. In order to overcome this problem, messages may need to be carried by the intermediate nodes for a period of time until the nodes get reconnected. Then, the messages are retransmitted again. Our objective is to test such a scenario by simulating different versions of data carrying protocols.

We start by presenting a literature review in Section II. We review Ad Hoc, MANET and DTN networks. In Sections III and IV, we review in details the protocols used in MANETs and DTNs respectively. In Section V, we describe our proposed solutions. In Section VI, we describe the simulation environment. In Section VII, we present the results. Finally we conclude the paper and give some ideas for future work.

II. STATE OF THE ART

We start with a review of different types of mobile networks.

A. Ad Hoc

An Ad Hoc network is a type of local area network (LAN). Each individual device in this network can communicate directly with any other device in a peer-to-peer style. This style eliminates the involvement of a central device that acts as a base station or a router. Ad Hoc networks operate with the absence of a fixed infrastructure. The nodes in Ad Hoc networks can be hosts as well as

 ISBN: 978-1-61208-217-2

routers which allows a message to be transmitted form node to node through the network until it reaches its final destination. The setup of an ad-hoc network is easy and simple and almost all operating systems support this type of network.

B. MANET

The MANET (Mobile Ad Hoc Network) is a type of Ad Hoc network with mobile nodes moving around. The mobile nodes in a MANET change locations and configure themselves on the move and use wireless channels to be able to communicate with each other [1]. The wireless medium could be a Wi-Fi medium, a cellular medium or a satellite medium. Each node could play the role of a source, a router and/or a destination [3].

A simple example of MANET topology could, for example, contain two nodes far away from each other that behave as a source node that sends messages and a destination node that receives the messages. Between these two nodes, there are a number of mobile intermediate nodes that act as hosts as well as routers [3]. Each intermediate node has a limited range in which it can make a connection to a neighbour node (i.e., two nodes that have a range of one meter have to be at most one meter away from each other in order to establish a connection and exchange messages). The message to be sent is then transmitted between those nodes until it reaches the destination. Due to the movement of the nodes and whether or not they are connected to each other, a direct path between the source and the destination is not always guaranteed.

C. DTN

A Disturbance-Tolerant Network (DTN) (also called Delay-Tolerant Network) is a network architecture that could be applied to operate over long distance communication. In this sort of communication, there are challenges that occur during a communication session. Some of these challenges include node mobility, lack of end-to-end paths, the limited transmission range associated with each node and other challenges. Due to these challenges, the existing TCP/IP routing protocol will not work efficiently and has to be modified or changed to work better with the characteristics of DTNs. Zhang (2006) categorized the routing protocols associated with DTNs into different categories: deterministic, stochastic, model-based, control movement and coding-based approaches [4]. Routing in DTNs has a direct relation with the work and development of the c-protocol discussed in this paper.

III. PROTOCOLS USED IN A MANET

Many protocols have been proposed to work in MANETs. Each one of these protocols has specific properties and structure to deliver a solution to a problem facing MANETs. A wide range of these protocols have been categorized into three major categories. These categories are Proactive Routing Protocols, Reactive Routing Protocols and Hybrid Routing Protocols [3] [5].

In proactive routing protocols, each node in the MANET network stores a table holding information about the other nodes in the network. Depending on the implementation of the protocol, the table might hold information about every other node in the network or some selective nodes [5]. These tables are updated periodically or whenever the topology of the networks changes [3] [5] [6]. The protocols in this category differ in the way they update the table/s and the information kept in them. Examples of protocols that fall under this category are DSDV (Destination Sequenced Distance Vector), WRP (Wireless Routing Protocol), GSR (Global State Routing), FSR (Fisheye State Routing), STAR (Source Tree Adaptive Routing) and DREAM (Distance Routing Effect Algorithm for Mobility).

Instead of updating the tables of all the nodes in the network, in Reactive Routing Protocols, updates are only performed on the nodes that need to send data at a specific time. This is called On-demand routing [3] [5]. This means the route from the source to the destination is determined upon sending. Usually the source floods packets into the network to determine the best route to the destination. The packets flooded are small packets known as route request packets (RREQ) [3]. Based on the acknowledgment/response/reply resulting from sending (RREQ), the best route is chosen to deliver the data. Reactive/On-demand routing is further categorized into two categories known as hop-by-hop routing and source routing [3] [5]. The difference between these two categories occurs in the header of the sent packets. In source routing, the full information of the address is stored in the packet header. In hop-by-hop routing, only the addresses carried by a packet are the final destination address and the next hop address. The source routing is reported to be inefficient due to the overhead resulting from carrying too much information in the packets headers [8]. Examples of Reactive/On-demand protocols are AODV (Ad Hoc On-demand distance vector), DSR (Dynamic Source Routing), ROAM (Routing On-demand acyclic multi-path), LMR (Light-weight Mobile Routing), TORA (Temporally Ordered Routing Algorithm) and ABR (Associativity-Based Routing).

Hybrid Routing protocols adopt a mix of the first and second categories' properties [5]. Examples of this category are ZRP (Zero Routing Protocol), ZHLS (Zero Based Hierarchical Link State), DST (Distributed Spanning Trees based routing protocol) and DDR (Distributed Dynamic Routing).

IV. PROTOCOLS USED IN DTN

Due to the repeated end-to-end connection loss, routing in DTN is challenging. A store-and-forward approach is used often in such networks. In Store-and-forward, a message is stored in an intermediate node until the node

 ISBN: 978-1-61208-217-2

sees an opportunity to retransmit the message. This gives the DTNs the advantage of delivering the message without the need of an end-to-end connection [7]. The routing protocols in a DTN are designed to overcome the problem of repeated disconnections. There are two categories of DTN protocols: Deterministic Routing Protocols and Dynamic or Stochastic routing protocols [7]. In Deterministic DTNs, the future topology of the network is known or could be predicted simplifying finding a route.

In Dynamic DTNs, the topology is not known. Dynamic Routing Protocols differ in the way they make decisions to which node a message is forwarded. A simple routing algorithm is called Direct Delivery where a node retransmits a message only if it gets in range with the destination [7]. Another routing algorithm is the First Contact algorithm [7]. In the First Contact algorithm, each node retransmits a message to a randomly-selected, in range, node. The decision made to choose a random node is not efficient since this randomly-chosen node might not be moving towards the destination. Epidemic routing is another approach where each node sends the message to be delivered to each other node in range (flooding). A node accepts a message only if it does not already have another copy of the same message in its buffer.

In Dynamic Routing, because of repeated transmissions, a lot of storage space is wasted. This raises the need of having a recovery scheme to deal with the copies of the data left in the network after a message is delivered. One solution is to introduce a life time parameter where a message is discarded if it has been carried for a period exceeding its life time. This life time scheme is optimal since the message would not reach the destination if the life time is too short. If it is too long, the storage capacity would be wasted. Another recovery scheme introduces acknowledgments that are flooded into the network once a message is received at the final destination. Each node in the network receives such acknowledgments. Then, it deletes the corresponding message stored in its buffer. These acknowledgments could be used as a way to guarantee successful delivery.

V. PROPOSED SOLUTIONS

Different versions of the c-protocol (also known as a store-and-forward-protocol) that do not relay on maintaining the end-to-end connections for a successful data transfer are proposed and simulated in this paper. The difference between these versions is in the algorithms that are used to forward the packets. The common property, which all versions share is the ability to carry/store a message for a while until a reconnection occurs.

These versions are; (1) First hop in the list routing (FLR), (2) closest hop routing (CHP) and (3) farthest hop routing (FHR). By introducing a GPS location (Global Positioning System), so that the distance to each node in the topology is known, (4) the closest to the destination routing (CGPS) and (5) forwarding to the hop that has the best next location to the destination (NGPS) are proposed. One last version of the c-protocol is simple flooding. In order to understand the underlying implementation of each version, a brief discussion is mandatory.

In First in the list routing, the node listed first in the next hop table, by the node currently carrying a packet (the carrying hop), is the one the packet is forwarded to. In closest hop routing, the distance between each connect node is calculated and the packet is forwarded to the one that is closest to the carrying node (in other words, the one having the strongest transmission signal). In farthest hop routing, the packet is forwarded to the farthest node from the carrying node (in other words, the one having the weakest transmission signal).

In GPS enabled routing, the current position of the destination is known to all the nodes in the network. In closest to destination routing, the distance between every connected node and the destination is calculated and sent to the carrying hop. The packet then is forwarded to the closest node to the destination. Since there is movement involved in the network, it cannot be guaranteed that the closest node to the destination is not moving away from the destination. To overcome this issue, forwarding to the closest *next location* to the destination is proposed. (i.e., next location is the location that a node is moving towards. Once the next destination is reached, the node changes direction) Rather than sending the packets to the closest current location of the node to the destination, they are forwarded to the node that has the closest next location to the destination.

In the simulation, the connectivity between the nodes is stored and maintained as a matrix of 0s and 1s, which means not connected and connected respectively. The movement of the nodes is considered random in this work. The way the nodes move is by generating a random X and Y coordinate (treated as the next location and bounded by the network area) and then move at a constant speed towards this next location. After the next location is reached, it is set to be the current location and a new next location is generated. The movement pattern used in the simulation is the same pattern used by the SetDest utility supplied with Network Simulator 2 (NS2) [8].

VI. SIMULATION ENVIRONMENT

When designing any protocol, a set of requirements has to be specified such as: guaranteed delivery, in-order delivery, packet duplication, etc. In this particular type of network topology, the movement pattern and density of the intermediate nodes plays a big role in designing the c-protocol. For testing, the implementation of the actual mobile nodes could be expensive and time consuming. We considered using an existing simulator (like NS2) but require a significant effort to add a new protocol [8]. Instead, a customized JAVA simulator was used to simulate

 ISBN: 978-1-61208-217-2

the network and to develop the c-protocol. The JAVA simulator provides a controlled environment for testing the c-protocol and provides a full control over the parameters and the algorithms used by the c-protocol.

A. *Assumption*

To simplify the simulation, some assumptions have been made. To be able to accurately monitor the data flow, it was decided that there was only one source sending and one destination giving only one data flow. To better visualize the network as well as to better understand how the intermediate nodes move, the source and the destination were assumed to be stationary with all other nodes mobile. Another assumption is related to the forwarding mechanism. It is assumed that packet transmission and delivery takes exactly 200 ms for every node (200 ms forwarding cycle).In 500 byte packets, this corresponds to the packet transmission rate of 20 kbps (assuming zero propagation time). In each forwarding cycle, only one packet is sent from each node if the node has a packet to transmit. Another assumption is regarding the type of the data flow and the data generator. The data generator is assumed to generate CBR traffic (Constant Bit Rate traffic), with no acknowledgements required, at the source with fixed interval (one packet every 800 ms) between each packet generated. This corresponds to a generation rate of 5kbps. We implemented a limit on time for carrying a packet in a node: A packet is carried in the node no longer than 10 seconds. Then it is dropped. Buffers were introduced in each node buffering no more than 50 packets at a time. If the buffer is full, it is assumed that the node is not going to receive any more packets until the buffered packets get resent or dropped due to the effect of TTL. In case of the buffers in the sending and receiving ends, a drop tail queue was introduce at the source which allows dropping the new generated packets if the buffer is full. At the receiving end, the buffer size is assumed to be infinity due to the fact that the receiving end is the final destination of the packets.

Table 1 contains the parameters used in the simulation along with their values.

Table 1 The values of the parameters used in the simulation

Parameter	**Value**
Network area	400m x 400m
Number of nodes	N = 7(Low density)
Transmission range	50m
Velocity	1 m/sec
Packet generation interval	800 millisecond
Forwarding cycle	200 millisecond
Location update	10 millisecond
End-to-end connection	Updated every 1 millisecond
Throughput	Calculated every 5 sec
Buffer Size	50 packets
Packet size	500 bytes
Carry time	10 sec
Total simulation time	180 seconds

VII. RESULTS

Two simulation scenarios were used in testing. One with N=7 mobile nodes and the other with 50.

Figure 1 shows the end-to-end connectivity between the source and destination during the simulation captured from the first simulation scenario (N=7).

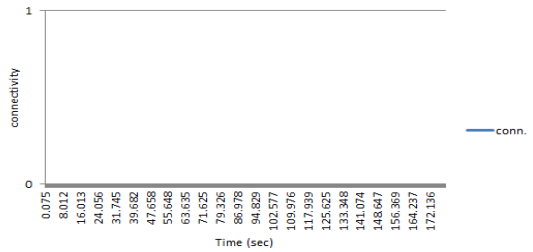


Figure 1: End-to-end connectivity N=7.

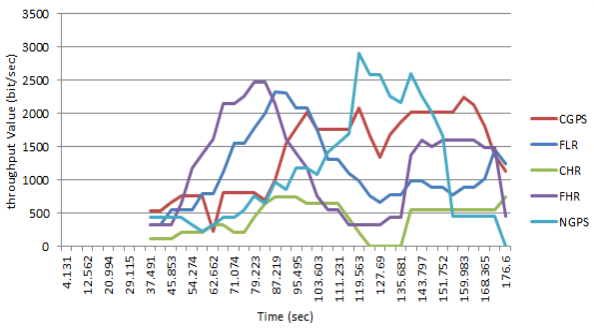


Figure 2: Throughput N=7.

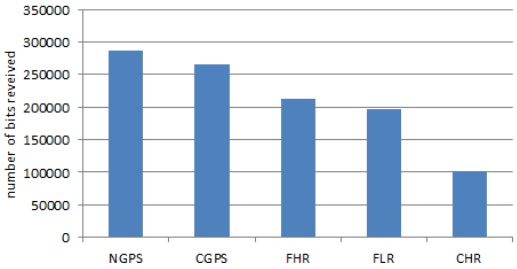


Figure 3 Total number of bits received N=7.

The graph shows that, in this experiment, the source and the destination are never connected by a closed path. Lack of end-to-end path in conventional MANETs prevents the data from being delivered. This issue has been addressed by introducing carrying feature implemented in c-protocols.

Figure 2 shows the throughput resulting from implementing different versions of the c-protocol. Although, in this experment there was no end-to-end path between the

 ISBN: 978-1-61208-217-2

source and the destination, the c-protocol allowed data to go through and be delivered to the destination. GPS enabled versions, CGPS and NGPS, allow for a larger amount of data to be delivered than other versions.

Figure 3 shows the total number of bits successfully received at the destination as a result of using different versions of c-protocol. Again, the graph shows that the GPS enabled versions have the highest delivery.

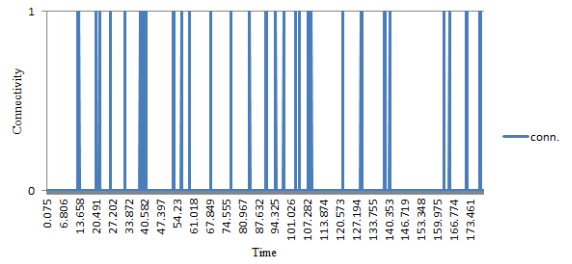


Figure 4 connectivity when N = 50

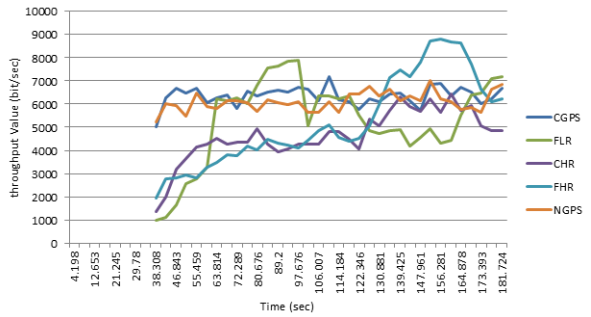


Figure 5 Throughput when N = 50

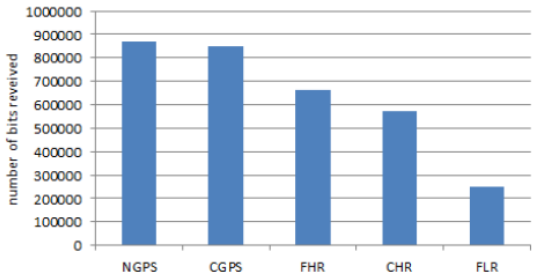


Figure 6 Total number of bits received N=50.

Figure 4 shows the end-to-end connectivity between the source and the destination in the second experiment when the number of nodes is 50 (N=50). The end-to-end connectivity occurs only periodically during this experiment.

The total time of end-to-end connectivity recorded in this experiment was 3600 ms. If we use 5kbps generation rate we can estimate the maximum number of bits (that can be potentially) delivered over the 5kbps connection is 18,000 bits. This would be the maximum throughput achievable using a conventional MANET.

The throughput for the c-protocol is shown in Figure 5. The total number of bits delivered is in Figure 6. For NGPS protocol, in our simulation we recorded almost 900,000 bits received in the course of the experiment.

The results show that introducing the c-protocol in MANET offers a significant advantage over conventional end-to-end protocols used currently in MANETs.

VIII. CONCLUSION

Currently in MANETs, it is essential to have an end-to-end connection in order to deliver packets; without end-to-end connectivity no packets would be delivered. C-protocol addresses this problem by introducing the carry (store/forward) mechanism that allows packets to be delivered even with the absence of an end-to-end connection.

There are two different types of protocols introduced in our work. They include the c-protocols that work without the need of using GPS location of the final destination and the c-protocols that make use of such a GPS location. Both types of c-protocols are reported to be able to deliver data to the destination even without end-to-end connection. The c-protocol versions that use GPS result in largest throughput in the simulated network.

IX. FUTURE WORK

The future work will include investigating the role of mobile node characteristics in a MANET, like the buffer size and the data retention time of the node, on the throughput of a MANET. We also plan to investigate other performance characteristics of the c-protocol like the propagation delay.

X. ACKNOWLEDGMENTS

This work is sponsored and funded by the Ministry of Higher Education of Saudi Arabia through the Saudi Arabian Cultural Bureau in Canada.

REFERENCES

[1] Fall, K. (2003) A Delay-Tolerant Network Architecture for Challenged Internets. Karlsruhe, Germany.*SIGCOMM'03*

[2] Yu, W., Wu, C., and Hu, X. (2010). Spray and Routing for Message Delivery in Challenged Networks, *Embedded and Ubiquitous Computing (EUC), 2010 IEEE/IFIP 8th International Conference on*, pp.472-475

[3] Kulkarni, N.S., Gupta, I., and Raman, B. (2009). On Demand Routing Protocols for Mobile Ad Hoc Networks: A Review, *Advance Computing Conference, 2009. IACC 2009. IEEE International* , pp.586-591.

[4] Zhang, Z. (2006). Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges, *Communications Surveys & Tutorials, IEEE* , vol.8, no.1, pp.24-37.

[5] Abolhasan M., Wysocki, T., and Dutkiewicz, E. (2003). A Review of Routing Protocols for Mobile Ad Hoc Networks. Elsevier B.V.

[6] Latiff, L. A., and Fisal, N. (2003). Routing Protocols in Wireless Mobile Ad Hoc Network - a review, *Communications, 2003. APCC2003. The 9th Asia-Pacific Conference on*, pp. 600- 604

 ISBN: 978-1-61208-217-2

[7] Socievole, A.; De Rango, F.; and Coscarella, C.(2011). Routing approaches and performance evaluation in delay tolerant networks, *Wireless Telecommunications Symposium (WTS)*, pp.1-6.

[8] Ekram, H., and Issariyakul, T.(2009). Introduction to Network Simulator NS2. Springer.

[9] Agrawal, D.P., and Zeng, Q. (2011). Introduction to Wireless and Mobile Systems.[Third Edition]. Brooks/Cole.

[10] Balakrishnan, M., Birman, K., Pleisch, S., and Renesse, R. (2006). MISTRAL: Efficient Flooding in Mobile Ad-hoc Networks. [Electronic Version]. MobHoc, Florence, Italy.

[11] Basagn, S., Conti, M., Gioradano, S., and Stojeneoric, I. (2004). Mobile Ad Hoc Networking. Willy-IEEE Press.

[12] Dai,F., Yang, S., and Wu, J. (2007). Logarithmic Store-Carry-Forward Routing in Mobile Ad Hoc Networks, Parallel and Distributed Systems, IEEE Transactions on , vol.18, no.6, pp.735-748.

[13] Seneviratn, A., and Sarikaya, B. (1998). Cellular networks and mobile internet, *Computer Communications*, 21(14), 1244-1255.

[14] Zhang, J., Li, W., Cui, D., Zhao, X., and Yin, Z. (2009). The NS2-Based Simulation and Research on Wireless Sensor Network Route Protocol, *Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09. 5th International Conference on* , pp.1-4, 24-26.

 ISBN: 978-1-61208-217-2

Using Symbolic Substitution Logic as an automated design procedure for QCA arithmetic circuits

Dietmar Fey, Bruno Kleinert
Department Computer Science 3
Friedrich-Alexander-University Erlangen-Nuremberg
D-91058 Erlangen, Germany
dietmar.fey@informatik.uni-erlangen.de, bruno.kleinert@informatik.uni-erlangen.de

***Abstract* — The paper presents a work-in-progress for an automatic synthesis procedure of an optical computing design principle onto future nanocomputing circuits based on Quantum-dot Cellular Automata (QCA). The goal of this work is to provide a contribution for the elimination of the lack of automatic design procedures for regular built-up QCA arithmetic circuits, which are built-up mostly manually so far. Furthermore by means of such a synthesis procedure a lot of designs made for optical computing arithmetic circuits can be directly transferred to QCA circuits. In this sense, this work is a contribution to ease the automatic synthesis of future arithmetic nanocomputing circuits.**

Keywords - optical computing; symbolic substitution logic; quantum-dot cellular automata, nanocomputing

I. INTRODUCTION

Symbolic Substitution Logic (SSL) was invented by Brenner et al. [1] in 1986 as a new method for the design of optical computing circuits. It was exactly tailored to the constraints and possibilities of a high-dense pixel parallel processing offered by optical hardware. The idea behind SSL is to search for a certain binary pattern within a binary pixel image and to replace the found patterns by another pattern. This substitution process can be exploited to realize a digital arithmetic in a highly parallel manner. The key features of SSL are characterized by their strong regularity concerning the pixel processing and the focusing on operating on elementary binary information cells, namely pixels, arranged in a grid structure.

In particular this situation is also given in Quantum-dot Cellular Automata (QCA) [2]. QCA is one of the promising nanotechnologies besides carbon nanotube field effect transistors and further nano device technologies based on tunneling effects that are considered as candidates for a new device technique to realize logic circuitry in the post CMOS area. Analogue to an optical computing scheme like SSL QCA are characterized by a highly dense implementation of binary information cells and a regular information flow. Whereas the elementary binary information cell in SSL was a pixel, which is either bright or dark, the binary information cell in QCA corresponds to two electrons, which are arranged in two distinguishable directions in a four dot quantum cell.

In literature, a really large number of proposals for QCA arithmetic circuits can be found, which have been developed largely manually (e.g. [3],[4]). However, there is still a lack of design methodologies that can be used for an automatic design process of arithmetic circuits based on QCA. There is an exception presented in [5], which proposes a methodology how to convert Boolean sum-of-products in an algorithmic way to QCA logic, in particular to QCA majority gates, which is the basic gate structure in QCA (see Section III). However, most of the QCA arithmetic circuits are still developed in a time consuming try-and-error process by hand.

On the other side there was a lot of research in the 1980s and 1990s in the Optical Computing community on SSL (e.g. [6],[7]), which brought numerous proposals for digital optical computing circuits based on the basic SSL logic building block, the so-called SSL rule (see Chapter II). Due to this fact and the similarities given in the kind how elementary information is handled in QCA and SSL, we present in this paper on-going research on developing strategies how SSL rules can be used for an automatic mapping process onto QCA circuits, which can be used in future design tools.

The rest of the paper is organized as follows. In Section II, we present the basic principles of digital optical computing based on SSL. In Section III, we explain nanotechnology information processing based on QCA. In Section IV, we present the mapping process between SSL rules and QCA cells for the example of one stage of a bit-serial QCA adder deduced from a SSL adder. Section V concludes and points out the remaining steps of this work.

II. OPTICAL COMPUTING WITH SSL

SSL [6],[7] has drawn a lot of attention during the 1980s and 1990s as a method for exploiting the space invariance of regular optical imaging systems for the set-up of digital optical hardware. The base of information processing in a SSL is the implementation of a so-called SSL rule. An SSL rule depicts a pattern substitution process and consists of two parts, a left-hand side (LHS) and right-hand side (RHS) pattern, (see Fig. 1). By a corresponding optical hardware each occurrence of the LHS pattern is searched within a binary image and is replaced by the RHS pattern. Fig. 2 shows schematically a possible optical set-up for the search process as it was frequently realized in SSL hardware demonstrators. The principle processing works as follows.

For each switched-off pixel, i.e., a black pixel, in the LHS of an SSL rule a copy of the image is produced, e.g., by a beam splitter. Furthermore a reference point is defined within the LHS pattern, e.g., the lower left corner pixel. Each of the copies is reflected, e.g., by tilted mirrors, in such a way that the copies are superimposed and pixels, which have

 ISBN: 978-1-61208-217-2

the same relative position to each other as defined in the LHS pattern, meet at the same location.

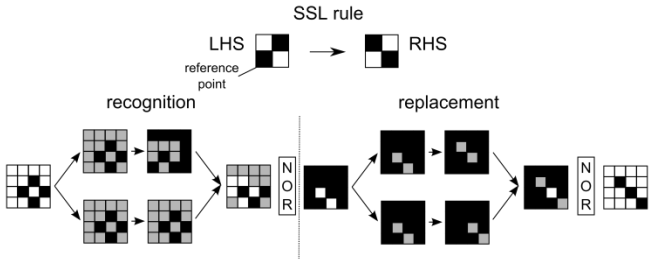


Figure 1. Principle of SSL.

For the example of Fig. 1, this means that one copy of the image is not tilted since it corresponds to the set pixel in the LHS pattern, which is already located in the reference point. Whereas the other copy is shifted by the tilted mirror, such that each pixel in the copy of the input image is shifted one pixel position down and left. At each position, where two dark pixels meet, an occurrence is given of the LHS pattern in the original input image. The superimposed image is mapped onto an array of optical threshold detectors. Each detector operates on one pixel of the superimposed image as a NOR device. The detector output is used for switching on a LED or laser diode. As a result, one gets a high light intensity at each pixel position, which corresponds to the occurrence of the LHS search pattern in the input image. We denote this new image as detector output image.

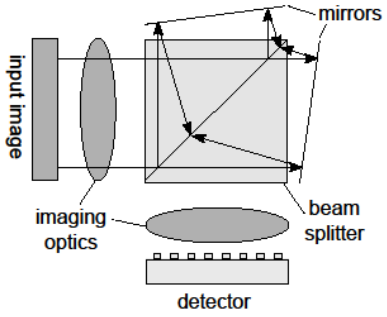


Figure 2. Implementation of SSL with optical hardware.

The recognition step is followed by a replacement step, which works analogue to the recognition step but in opposite direction. For each switched-on pixel in the RHS pattern a copy of the detector output image is again produced by optical beam splitter hardware in such a way that the copies are shifted towards the switched-on pixel in the RHS pattern. This means for the example of Fig. 1 that two copies from the detected output image are generated and each of the copies are shifted one pixel up resp. right before superimposing the copies. Once again, the superimposed image is mapped onto a pixel-by-pixel operating NOR detector and LED/laser diode array. The reproduced output is a new image, in which each occurrence of the LHS pattern in the original input image is substituted by the corresponding RHS pattern.

Implementing appropriate SSL rules by splitting the input image into multiple optical recognition and replacement paths, which are applied simultaneously and joined at the end, have been used for the proposing and realizing of digital optical computer arithmetic circuits. Fig. 3 shows this schematically for an optical ripple carry adder based on SSL. A large number of further arithmetic circuits using SSL or similar techniques like optical shadow logic [8] have been published in the past for optical adders, multipliers or image processing tasks. All these proposals can be used to transfer them to QCA due to the similarities between SSL and QCA we outlined above.

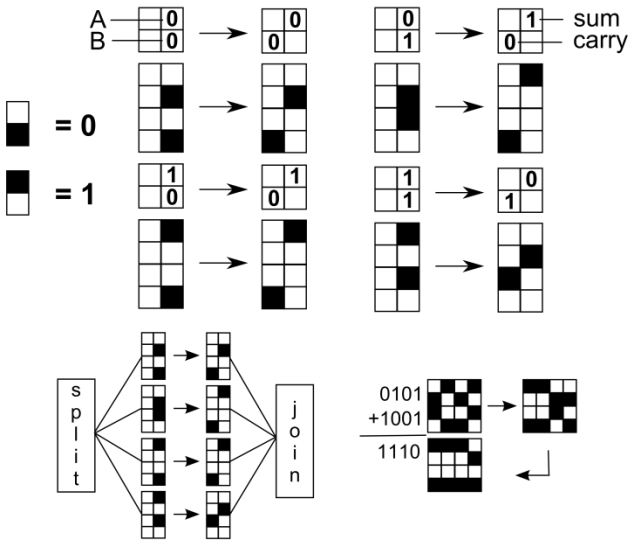


Figure 3. Realization of a ripple carry adder with SSL. For reasons of improved robustness a dual rail coding is used for 0 and 1.

III. NANOCOMPUTING WITH QCA

The elementary information cell in a QCA is a kind of container that groups a few quantum dots, at which charged particles, like e.g., electrons, are fixed (see Fig. 4). Mostly a QCA cell consists of four dots, in which two electrons are grouped in opposite order. Consequently, the cell knows exactly two polarizations orders, which are assigned to the binary values 0 or 1. Due to quantum mechanical rules it is possible that a cell can switch between the two states by tunneling of the charged particles between the dots. Concerning the two different particle arrangements one distinguishes between type 1 and type 2 cells.

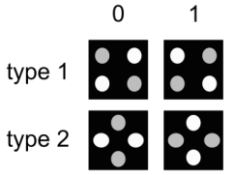


Figure 4. Binary coding in QCA cells. White circles correspond to empty qauntum dots, gray ones represent dots occupied with electrons.

A QCA cell serves not only as an information storage cell but also as a transport cell since neighbored QCA cells interchange by Coulumb forces. This means that a cell, which is fixed to a certain polarization, transfers its state to a neighbored cell because this arrangement shows the mini-

 ISBN: 978-1-61208-217-2

mum electrical field energy between neighbored charged particles. Consequently, a QCA wire can be built up, in which information is transported not by an electric current flow but by subsequent reordering of the quantum states in neighbored QCA cells. Due to the fact that no current is flowing and due to the extreme small dimensions of a QCA cell this technology offers very low power dissipation. Besides information transport one needs also logical gates to realize computing circuits. QCA logic utilizes an inverter and a so-called majority gate for this purpose. Fig. 5 shows an inverter built with cells of type 1. In both circuits, the output cell adopts the opposite state of the input cell state, again due to Coulombic forces. In contrast to CMOS circuits, QCA gate logic is not based on the switching of parallel and serial connected transistors but on the states of the cells surrounding a certain QCA cell, serving as output cell of the gate. The majority of the states in these surrounding cells determines the state of the output cell. In Fig. 5, a 3-input majority QCA gate is shown. The output cell adopts the same state, which at least is stored in two of the three neighbor cells. By fixing one of the inputs to a certain polarization 2-input AND, OR, NAND and NOR gates can be built-up.

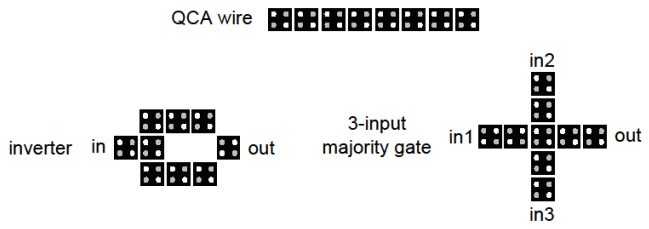


Figure 5. QCA logic building blocks.

Based on these three building blocks, QCA wire, QCA inverter and QCA majority gate, various proposals exist in literature for different typical digital circuits like adders, multipliers, shifters, multiplexers, flip-flop memories and registers, which have been found in a more or less try-and-error procedure. A very impressive collection of computer arithmetic QCA adder and multiplier circuits can be found in the work made by Hänninen [9]. The solutions proposed in this work are distinguished by their regular set-up that helps to realize QCA cells in the future. This is an important feature since QCA technology has a long-term perspective concerning its realization with real hardware.

Also design tools, which support the automatic synthesis of regular built-up QCA circuits, will encourage and give hints to device technologists how QCA technology should develop in the best way. In this sense, we propose to use optical computing SSL design procedure as a design entry point for the systematic design of nanocomputing QCA logic. How this mapping can be done is presented in the next section.

IV. MAPPING SSL RULES TO QCA LOGIC

The procedure to map SSL logic to a regular built-up QCA layout is subdivided in three steps. These steps correspond (i) to the core of the logic circuitry, namely the synthesis of a SSL rule into an equivalent QCA circuit, (ii) the realization of the splitting process, because we want to realize systems, which apply multiple SSL rules simultaneously, and (iii) the realization of the join at the end of the recognition-substitution stages. We will demonstrate the generic approach for these mapping steps in the following subsections without loss of generality on the example of the ripple carry adder from Fig. 3. Furthermore, we will use this example also to show generic applicable optimization measures for mapping SSL rules, which saves otherwise necessary QCA logic resources.

A. *Mapping the Split stage to QCA cells*

As shown in Fig. 3, the applying of multiple SSL rules starts with a Split function. The mapping of the Split stage onto QCA logic can be done in a straightforward manner. Producing copies of input cells can be simply done with branches of QCA wires running orthogonally to the input QCA wires. If one has to copy more than one input, as for example for the LHS rules in a ripple carry adder, one has to observe that crossing branches can interchange without conflicts. This can be done by crossing lines between QCA cells of type 1 and type 2. Converting between these two types can be realized with QCA cells, which are shifted about one half cell height (see Fig. 6, part Split).

B. *Mapping SSL rules to QCA cells*

The mapping of SSL rules onto equivalent QCA layouts is divided in two substeps, (i) the mapping of the recognition step, and (ii) the mapping of the replacement step. The recognition of a LHS of an SSL rule is mapped to an equivalent QCA majority gate realizing an appropriate AND gate. The number of inputs of this AND gate depends on the given number of values in the LHS. For example, the number of relevant inputs for the rules of the ripple carry adder is two. This means that a three-input QCA majority gate can be used, if one of the three inputs is fixed to 0 (see Fig. 6). For rules with a higher number of input values an appropriate majority AND gate has to be used. A lot of solutions for QCA gates with more than 3 inputs can be found in literature, e.g., in [10] an optimized solution for a 5-input majority gate is presented. If the value in the LHS is 0, then an inverter has to be included in the path of QCA cells that leads the input value corresponding to the LHS entry to the input of majority gate. The output of the majority cell is exactly 1, if the LHS pattern is detected. In this sense the majority gate works analogous to the photo detector NOR device used in SSL (see Fig. 1).

The following explanations correspond to the replacement stage in SSL. If the output of the majority gate is 0, then 0's are produced for all 1 values in the RHS of the corresponding SSL rule. If the output of the majority gate is 1, i.e., the LHS pattern was detected, a 1 is produced for each value 1 given in the RHS by an additional majority gate operating as an OR gate (see Fig. 6, majority gate in replacement part with one input fixed to 1). If the RHS is 0 no majority gate is necessary since we will work with wired-OR buses in the Join stage that carry already the 0 value, which is possibly inserted by the replacement stage located at the lowest position () in the wired-OR bus (see rule 2 in Fig. 6).

 ISBN: 978-1-61208-217-2

C. Mapping the Join stage to QCA cells

As just mentioned the principle of the Join stage in SSL is the realization of an optical wired-OR. The same idea is pursued for the equivalent QCA logic. If a 1 has to be inserted in the wire due to a relevant 1 from a RHS, which is output from a replacement stage, this can be done with 3-input majority gates with one input fixed to 1 (see Fig. 6,wired-OR bus in block rule 1). In this case, a 1 is only injected in the wire if the output of the attached recognition stage to the wire is 1 or the third input coming from the wire is already 1. This functioning corresponds exactly to a wired-OR bus. A logical 1 is injected if a LHS was found and a 1 in the corresponding output of the RHS is given. If the detected rule requires a 0 in the RHS this is automatically given by the fixed injection of a 0 in the QCA wire by the lowest replacement stage attached to the wired-OR bus. If the rules are not in conflict, i.e., only the LHS of exactly one rule was found, then only the output of the RHS belonging to the LHS is injected. This can be either a 0 or a 1. If it is a 0 an explicit injection is not necessary. This causes that rules, which have only 0's in the RHS, must not be implemented if it is secure that exactly one of the rules is always valid. This is given for the case of the ripple carry adder. Therefore, the rule corresponding to (A,B)=(0,0) has not to be implemented with corresponding QCA cells. If it is the only rule that holds, then the corresponding 0's in the output are already on the QCA wires. Utilizing this a priori knowledge the requirements to QCA hardware can be optimized during the synthesis process from SSL logic to QCA logic.

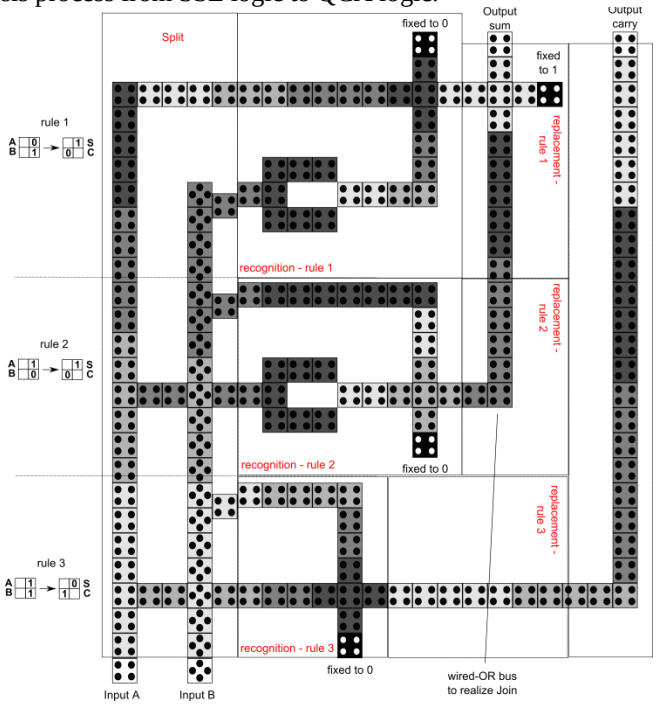


Figure 6. Result of the mapping process of SSL logic onto QCA logic for the ripple carry adder. To synchronize the changing of QCA cell states' four different clock zones have to be defined. In the figure these four clock zones are marked with a different gray level in QCA cells.

V. CONCLUSION AND FURTHER WORK

We have presented a generic design procedure for mapping digital optical computing circuits based on SSL onto nanocomputing QCA circuits. This will form both the base for future design tools for compact, regular built-up QCA circuits and supports the direct mapping of optical computing circuits to QCA technology. For example, we intend to map an integer arithmetic unit based on SSL, designed by us [11], onto a complete QCA integer unit, which does not yet exist so far. In addition, we have to verify the schematically shown QCA circuit of Fig. 6 by simulation with the QCA designer tool [12], the standard for simulating QCA layouts. So far, we have only verified parts of the circuit. Furthermore the insertion of an exact clocking scheme for the QCA cells has to be considered in the synthesis procedure. Nevertheless, the basic step for an automatic synthesis of SSL arithmetic circuits to QCA layouts is established.

REFERENCES

[1] K.-H.Brenner, A. Huang, and N. Streibl, "Digital Optical Computing with Symbolic Substitution", Appl. Opt. **25**, No. 18, pp. 3054 - 3060, (1986).

[2] C.S. Lent, P. Tougaw, W. Porod, and G. Bernstein, "Quantum cellular automata" Nanotechnology, vol. 4, 1993, pp. 49-57.

[3] V.A. Mardiris and Ioannis G. Karafyllidis, "Design and simulation of modular 2^n to 1 quantum-dot cellular automata (QCA) multiplexers," Int. J. Circ. Theor. Appl., 2010, vol. 38, pp. 771–785, doi: 10.1002/cta.595.

[4] F. Bruschi, F. Perini, V. Rana, and D. Sciuto, "An efficient Quantum-Dot Cellular Automata adder," Design, Automation & Test in Europe Conference & Exhibition (DATE), March 2011 , pp.1-4.

[5] R. Zhang, K. Walus, W. Wang, and G.A. Jullien, "A method of majority logic reduction for quantum cellular automata," IEEE Trans. on Nanotechnology, vol.3, no.4, Dec. 2004, pp. 443- 450, doi: 10.1109/ TNANO.2004.834177.

[6] Ahmed Louri, "Parallel implementation of optical symbolic substitution logic using shadow-casting and polarization," Applied Optics, Vol. 30, Issue 5, pp. 540-548 (1991) http://dx.doi.org/10.1364/AO.30.000540.

[7] K.-H. Brenner, W. Eckert, and C. Passon, "Demonstration of an optical pipeline adder and design concepts for its microintegration," Optics & Laser Technology, Vol. 26, No. 4, 1994, pp. 229 – 237.

[8] Y. Ichioka and J. Tanida, "Optical parallel logic gates using a shadow-casting system for optical digital computing," Proceedings of the IEEE , Vol. 72, No. 7, July 1984 , pp. 787 - 801 , doi:10.1109/PROC.1984.12939.

[9] I. Hänninen, "Computer Arithmetic on Quantum-Dot Cellular Automata Technology," Ph. D. thesis, Tampere University of Technology, 2009.

[10] R. Akkela and M.D. Wagh, "A Five-input Majority Gate in Quantum-dot Cellular Automata", NSTI-Nanotech 2011, Vol. 2, pp. 13-16, 2011.

[11] D. Fey and K.-H. Brenner, "Digital optical arithmetic based on systolic arrays andsymbolic substitution logic," Opt. Comput. 1, 153–167 (1990).

[12] K. Walus, T.J. Dysart, G.A. Jullien, and R.A. Budiman, "QCA Designer: a rapid design and Simulation tool for quantum-dot cellular automata," IEEE Trans. on Nanotechnology, Vol. 3, No. 1. (2004), pp. 26-31, doi:10.1109/ TNANO.2003.820815

 ISBN: 978-1-61208-217-2