CONTENTS

Automat. Comput. Appl. Math.
Volume 21 (2012), Number 2

COMPUTER SCIENCE

Certain Considerations Regarding the Database Complexity

Mircea Petrescu
Department of computers
University "Politehnica" Bucharest
Bucharest, Romania
mircea.petrescu@cs.pub.ro

***Abstract*— In the presented paper, the author used an idea developed within the computational complexity theory [1], trying to emphasize the fact that for defining and manipulating a database some amounts of resources are required. On this basis, a connection was proposed with the Kolmogorov complexity, a central concept in algorithmic information theory. One of the conclusions was that for describing the complexity of a given element belonging to a relational database, the length of the program used in defining this element can be used. In the simple example presented in the paper a relation schema containing 4 attributes was used. For showing the effect on complexity of functional dependencies, the study was completed by adding several dependencies to the description of the used database.**

Keywords- ***complexity; entropy***

I. INTRODUCTION

The general subject of complexity examined from different points of view has been increasingly present in the literature. Much work is constantly devoted to the complexity of interrogation procedures [2] for „information systems", and even for deepening the understanding of the conceptual aspects of the term „complexity" itself. The knowledge of the author is obviously limited, and I must recognize that accordingly to the degree I have been able to inspect the available for me sources the issue of „database complexity" is not very favored by the literature, at least under this name. This is the main reason for the modest attempt contained in this paper, in the intention to open a new way in studying this matter.

In most cases, at least of those having a practical significance, the technical and physical infrastructure complexity of different systems is emphasized, particularly for information systems [3]. More general aspects describing the very broad use of the complexity concepts in science, economics, and social life are presented in [4].

A decisive advancement concerning the philosophy and the mathematical theory of complexity was achieved in 1963-1965, when A.N. Kolmogorov published his basic works on algorithmic information theory, generally known today as Kolmogorov complexity [5, 6]. Accordingly to Kolmogorov's definition, the complexity of an object, such as a piece of text, is the measure of computational resources needed to specify that object. In fact, the basic idea, formulated first by A.N. Kolmogorov, is to measure the complexity of an object by the size in bits of the smallest program for computing it. As G.J. Chaitins expresses it [7], the „algorithmic information theory" is the result of „putting Shannon's information theory and Turing's computability theory into a cocktail shaker and shaking vigorously". Sometimes, this type of complexity is called Kolmogorov-Chaitin complexity or algorithmic entropy. Of course, here one should take into account the information entropy, used by Kolmogorov in [6] (for the variable x, the entropy is $H(x) = \log2(x)$. In [8] an attempt was made to define and calculate the entropy of a database, the basic intention of the author being to find a link between the entropy and response time in using the databases.

II. DATABASE COMPLEXITY

The property of complexity of a database system results easily from the accepted definitions of the concept „database". If we attempt to get deeply into the above concept, we will discover a whole "universe" of specific aspects. Accordingly to the Oxford Dictionary, something is complex if it is composed of "usually several" closely connected parts. Since a database contains a number of "relations", or tables, each of these comprising a number of "tuples", and a tuple is formed by the values of attributes describing a real system, etc., the complexity of a database follows by itself. If we take into account the definition of a relation schema, the fact that the attributes are connected by functional and multi valued dependencies , the scope of complexity of a database becomes almost endless.

III. THE KOLMOGOROV COMPLEXITY IN THE CASE OF DATABASES

Let's assume that we have a relation called Persons, defined in SQL language as:

```
CREATE TABLE Persons
(
person_no            INT (10)        NOT NULL
person_familyname    Char (20)       NOT NULL
person_givenname     Char (20)       NOT NULL
person_address       Char (20)       NOT NULL
);
```

We have here a string of „objects" of the type, for example, „250 popa radu stra", etc. The objects are specified by the „computational resource" represented by the program:

„CREATE TABLEPersons (person_no, person_familyname, person_givenname, person_address) ".

New Semantic-Based Approach for Extracting Parallel Text Data from Comparable Corpora

Eng. Marius Zubac
Universitatea Tehnică
SDL Language Weaver
Cluj-Napoca, Romania
Marius.Zubac@cs.utcluj.ro

Prof. PhD Eng. Vasile Teodor Dădârlat
Universitatea Tehnică
Cluj-Napoca, Romania
Vasile.Dadarlat@cs.utcluj.ro

***Abstract*—Wikipedia can be seen as a multilingual comparable text corpus. This paper describes a new method for parallel documents extraction by mining this site above, followed by segments alignment. The segment-level alignment algorithm is based upon both empirical evidence and lexical and semantic resources and claims superior results. Although the method is applied to extracting parallel data from Romanian and English versions of Wikipedia, the method is general enough to be applicable for other language-pairs as well.**

Keywords: bitext alignment, Wikipedia, WordNet, comparable corpora, semantic methods, lexical resources.

I. THE NEW INTEREST FOR COMPARABLE CORPORA

The interest for *parallel corpora* is well known and continuously proved by the ever growing domain of NLP applications. Good surveys of the domain covering both *bitext* [1] alignment and parallel corpora applications panel are provided by Véronis [2] and more recently by Tiedemann [3] where a special section is dedicated to comparable corpora.

A. Comparable Corpora

While full parallel corpora are still scarce resources the interest shifted in the recent years to *comparable corpora* which in comparison with the parallel ones are easier to find in large quantities, present more diversity and cover more languages including the low density ones. Regarding the term itself there is no general agreement. A 'continuous spectrum' of text parallelism is to be found between parallel and comparable corpora as shown by Fung and Cheung [4]. Nevertheless various degrees of parallelism exist at the word, phrases, clauses and sentences level, parallelism that can be exploited in many multilingual applications while helping the mainstream MT applications. A less restrictive approach is proposed by the Expert Advisory Group on Language Engineering Standards Guidelines (EAGLES, 1996, http://www.ilc.pi.cnr.it/EAGLES96/browse.html) where the following definition is given: "A comparable corpus is one which selects similar texts in more than one language or variety. There is as yet no agreement on the nature of the similarity, because there are very few examples of comparable corpora". Once the interest for comparable corpora set the research spurred in three main directions: a first direction regarded the detection and the extraction of such corpora primarily from the web. A second direction of research was focused on devising new methods for extracting parallel data from within the comparable corpus. Finally the last direction focused on the application level itself. For the domain of Romanian NLP it is important that such corpora exist in a wider variety and larger volume therefore the interest for acquiring such corpora.

B. Acquiring Comparable Corpora

The spectrum of challenges when acquiring comparable corpora is not much different from the one in place for parallel corpora. Well-known sources of comparable text can be found on multilingual newswires agencies like BBC (www.bbc.co.uk/ in 25 languages), AFP (www.afp.com with news in 6 languages), Xinhua (www.xinhuanet.com with editions in 7 languages) or SETimes (www.setimes.com/ in 7 South-East European languages). Unfortunately none of the sites mentioned above provide for Romanian text including SETimes which discontinued some time ago the support for this language. Another important comparable text resource is the Wikipedia site (www.wikipedia.org). This site started in 2001 with English encyclopedic articles reports today articles in over 289 languages and each mainstream language has all over 1 million articles all freely available.

C. Methods, Techniques and Systems in Parallel Text Data Alignment

Extracting the data from the web has a path opened by Resnik's seminal articles [5] and Resnik and Smith [6]. The authors built the STRAND system that allowed them to discover parallel pages in web sites. They used a strong assumption based on empirical evidence that sites have a strong tendency to use the same internal structure and then applied probabilistic generated translated lexicons and structural-matching techniques for the purpose of document alignment. Similar techniques have been applied also by Chen and Nie [7] with their PTMiner system built for Chinese-English web mining where again they applied a strategy for parallel links detection based on html files naming conventions. Ma and Liberman [8] described their BITS system where content-based algorithms have been applied to compute similarity measures between texts. A more recent approach PARADOCS has been described by Patry and Langlais, [9] where authors are not making any assumptions on naming conventions on filenames or URLs. Another interesting research direction is reported in works by Adafre and Rijke [10], Yasuda and Sumita [11], and Mohhammadi and Aghaee [12] where Machine Translation techniques are used to generate candidate pairing documents and then various similarity measures including Dice, Cosine and

A Coincidence-based Statistical Test for Randomness Assessment

Kinga Márton, Alexandru Moldovan, Alin Suciu
Computer Science Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
{ Kinga.Marton, Alexandru.Moldovan, Alin.Suciu }@cs.utcluj.ro

***Abstract*— The quality assessment of random number sequences is a rather difficult task considering that statistical testing can highlight certain deviations from randomness but cannot guarantee perfect randomness. Nonetheless, through extensive analysis and thorough testing one can get a high confidence in the generator but cannot be absolutely sure. This is the reason why input sequences have to be tested from many statistical perspectives, and the present paper introduces such a new perspective in the form of a statistical test based on the index of coincidence between subsequences of the assessed bit sequence and describes several implementation methods together with benchmark results.**

Keywords-statistical test; index of coincidence; performance; population count

I. INTRODUCTION

The quality assessment of random number sequences is a rather difficult task considering two main issues. First, the quality depends on the nature of the entropy source a random number generator uses to extract randomness from. Second, certain properties of random number sequences are statistical and hence the statistical quality can be assessed by various statistical randomness tests. However, the most important problem in connection with randomness is the lack of certainty. By extensive analysis and thorough testing one can get a high confidence in the generator but cannot be absolutely sure.

Based on the nature of the employed entropy source, the wide domain of random number generators can be categorized in three main classes: true random number generators (TRNG), pseudorandom number generators (PRNG) and unpredictable random number generators (URNG).

TRNGs extract randomness from natural physical phenomena (like thermal noise, jitter, radiation, etc.), exhibit the highest level of nondeterminism and irreproducibility but do not necessarily present perfectly uniform distribution and independence of generated values.

PRNGs extract randomness from an initial value, called seed, which is expanded by means of a deterministic (usually recursive) formula, and provide a practical way for generating random sequences using only software methods.

URNGs are practical approximations of TRNGs, extracting the unpredictability induced by the complexity of the underlying phenomenon. URNGs exhibit certain characteristics of both TRNGs and PRNGs since they rely on the behavior of hardware devices like TRNGs are but perform a deterministic sequence of operations like PRNGs do.

A more detailed description of the different categories of random number generators can be found in [4].

The process of statistically assessing the quality of random number sequences usually involves several test suites, specially designed to highlight certain deviations from randomness.

The most widely known and employed batteries of statistical tests are the NIST test suite [5], TestU01 [1], ENT [6], and Diehard [3]. But the problem, as already mentioned, is that no finite amount of statistical tests can guarantee perfect randomness. By using various tests that assess the quality form many different angles and try to rule out sequences that do not exhibit certain statistical properties, we can increase the confidence in the tested generator or reject it due to its proven unsuitability for the given application. Nonetheless there may exist some other statistical tests that may prove that the tested sequence lacks certain other properties that a perfectly random sequence should exhibit

In this context, analyzing the sequence from as many different statistical perspectives as one can afford – with costs in processing time and storage capacity – is essential.

Our present work introduces a novel statistical test based on the counting of bit level coincidences among the subsequences of the assessed sequence. The goal is to inspect whether the index of coincidence between subsequences of different sizes verify the statistical property expected from a perfectly random sequence, namely that the index of coincidence is sufficiently close to 50%.

The rest of the paper is organized as follows. Section 2 provides a meaningful glance into the domain of statistical testing with the aim of placing the proposed test into context. Section 3 presents the proposed test, followed by Section 4 introducing the various approaches for counting the index of coincidence. Experimental results that assess the efficiency of each solution are included in Section 5. Section 6 presents final conclusions and further work.

II. COINCIDENCE COUNTING IN CONTEXT

Coincidence counting is not a new concept, in fact the problem of counting the coincidences between certain sequences of values is of special importance in various fields such as cryptanalysis - useful in both analyzing the natural plaintext language and the ciphertext; in DNA analysis where subsequence matching heavily relies on the index of coincidence, etc.

In the field of statistical randomness assessment the concept of coincidence is related to the general concept of correlation and especially to autocorrelation as opposed to the

Pseudo-Random Number Generation in Prolog

Alin Suciu, Melania Miron, Kinga Márton
Computer Science Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
{ Alin.Suciu, Melania.Miron, Kinga.Marton }@cs.utcluj.ro

***Abstract*—The problem of pseudo-random number generation is important in any programming language and Prolog is no exception. To overcome the lack of flexibility in current Prolog systems, we propose a portable and flexible Prolog library consisting of three well known families of pseudo-random number generators (PRNGs): Linear Congruential Generators (LCG), Quadratic Congruential Generators (QCG) and Cubic Congruential Generators (CCG). We perform a performance (speed) comparison of the proposed generators when running in Sicstus Prolog, Swi Prolog and Yap Prolog.**

Keywords-Prolog; random number generation; performance

I. INTRODUCTION

The task of implementing pseudo-random number generators is important in any programming language, and Prolog, the mainstream logic programming language, is no exception.

Every Prolog implementation offers a set of library predicates which allow the programmer a more or less effective utilization of pseudo-random numbers. However, the problem of generating these numbers is not sufficiently emphasized and is sometimes quite overlooked in several implementations.

In the following we will consider three well known Prolog implementations: Sicstus Prolog[1], SWI Prolog[2] and Yap Prolog [3]. With regard to the random number generators included in these Prolog implementations there are practically two PRNGs employed: Sictus Prolog and Yap Prolog provide a PRNG based on the well known Wichmann Hill algorithm [10] while SWI Prolog offers a PRNG based on the GMP library [2].

A common characteristic in providing access to RNGs by current Prolog implementations is the lack of flexibility (the presence of a single PRNG) associated with a significant level obscurity surrounding the provided PRNG (and particularly its internal state). This is especially intriguing given the large number of PRNG families generally available and widely used today (e.g. LCG, QCG, CCG, etc.). In this context we strongly believe that the programmer should be given a more flexible choice when using PRNGs in Prolog.

Just like the traditional approach for pseudo-random number generation in imperative languages like C or even object oriented languages like Java and C#, Prolog uses the same state-based persistence of the PRNG. The system maintains an internal state of the PRNG which is updated after every generation.

However, since each Prolog implementation uses a certain and fixed PRNG, with a specific fixed representation for the state of the PRNG, the portability of programs initializing and using PRNGs in Prolog is considerably reduced.

For example, in Sicstus Prolog the state of the PRNG is expressed by the functor 'random' followed by four arguments, while in SWI Prolog the state is a number of 6014 decimal digits and furthermore, in Yap Prolog the state is a term having the functor 'rand' followed by three arguments.

To overcome these problems, the aim is to provide a much wider choice for generating pseudo-random numbers by integrating a new Prolog library for pseudo-random number generation. Currently our implementation offers 27 predefined generators: 9 of the most popular generators in the LCG family, 9 of the most popular generators in the QCG family and another 9 generators from the CCG family. Additionally, the programmer has the option of choosing a general generator from each family, in which case he or she has to provide the right parameters for the generator. The internal state is clearly visible and easily available for save and/or restore for all these PRNGs.

All the benchmarked Prolog implementations basically offer a predicate to obtain one single random number; of course one can create a predicate that repeatedly calls the PRNG and builds a list, but this will imply saving and restoring the state of the PRNG just as many times. A more effective way would be to have a predefined predicate that offers a list of random numbers starting from a given seed. Our implementation is designed to fill this need as well.

Consequently we define two modes of operation for each PRNG:

- batch mode – for requesting a list of numbers,
- interactive mode – for requesting a single number (possibly repeatedly).

The interactive mode closely resembles what is currently offered by the Prolog implementations and it implies retrieving and saving the state of the generator after each generated number (assert / retract for side effects).

On the other side, the batch mode, which looks like a batch of requests for random numbers is being answered by returning a list of numbers, and does not imply a persistent state therefore it can be used for a more efficient generation of large amounts of random numbers.

Additionally, for each generator the execution time (runtime, walltime) can be measured using a very simple and straightforward predicate. This facilitates the performance comparison between implementation in Sicstus Prolog, Swi Prolog and Yap Prolog.

The rest of the paper is organized as follows. Section 2 presents Linear Congruential Generators including their Prolog implementation, followed by Quadratic Congruential Generators and Cubic Congruential Generators presented in Section 3 and 4. Experimental results are provided in Section 5 and Section 6 presents the final conclusions and further work.

Image Enhancement in Daytime Fog Conditions

Mihai Negru
Computer Science Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
Email: Mihai.Negru@cs.utcluj.ro

Sergiu Nedevschi
Computer Science Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
Email: Sergiu.Nedevschi@cs.utcluj.ro

Radu Ioan Peter
Mathematics Department
Technical University of Cluj-Napoca
Cluj-Napoca, Romania
Email: Ioan.Radu.Peter@math.utcluj.ro

***Abstract*—The images captured in fog conditions have degraded contrast, that makes current image processing applications sensitive and error prone. We propose in this paper an efficient image enhancement algorithm suitable for daytime fog conditions and based on the Koschmieder's model. Using this mathematical model together with an original inference of the atmospheric veil induced by the fog we are able to recover the original fog-free image. A quantitative and qualitative evaluation is performed on both synthetic and real camera images. Our algorithm is suitable for both color and gray scale images and is able to perform image enhancement in real time.**

***Keywords*—daytime fog, image enhancement, contrast restoration, median filter, atmospheric veil**

I. Introduction

The visibility in images can be downgraded due to different natural phenomena such as haze, fog, mist, rain, etc. In such situations the visibility distance is decreased because of the absorption and scattering of light by the atmospheric particles. The light emanating from objects in the captured scene is attenuated by scattering along the line of sight of the camera. Images of outdoor scenes, captured during fog conditions, are drastically degraded. This weather phenomenon is especially dangerous in driving situations, because drivers tend to overestimate the visibility distance while traveling in fog conditions and drive with excessive speeds [1]. Due to the presence of fog, the visibility distance decreases exponentially, thus making fog one of the most dangerous weather condition for driving. Some of the negative effects of fog on the quality of the image are the loss of contrast and the alteration of the natural colors from the image. In addition the scattering effect of the transmitted light causes additional lightness in parts of the image [2]. These effect is called air-light or atmospheric veil. In order to overcome these impediments we must either adapt the operating parameters of the camera or try to detect the presence of fog and remove its effects from the images. In this work we focus on the second approach, namely we are dealing with restoring the contrast and enhancing the quality of the original foggy image.

Extensive research has been carried out in the field of fog detection and visibility estimation in fog conditions. Some methods [3], [4] use Gabor Filters, at different frequencies, scales and orientations in order to detect and classify the fog conditions. Their method is suitable for both day time and night time fog detection. Other approaches try to estimate the visibility distance by computing the position of the horizon and inflection point lines in the image [1], [5]. The classification of the fog density is done based on the obtained visibility distance. A similar method is presented in [6]. Fog detection is based on the computation of the vanishing point; the road lines are taken as reference lines in order to compute the vanishing point. After the vanishing point is found a segmentation of the road and sky is performed. All the above methods require only one input image. The authors in [7] and [8] fuse the information from an in-vehicle camera with a millimeter wave radar in order to classify the fog density and estimate the visibility range. They detect the preceding vehicle and compare the image area found with a fog free reference image. Then the distance obtained between the original image and the fog free image is used together with the distance measured by the millimeter wave radar in order to compute Koschmieder's atmospheric extinction coefficient [9]. But the accuracy of the method strongly depends on the computed distance between the original and the fog free reference image. Another drawback is the lack of confidence when there is no vehicle in front of the ego vehicle, so the fog conditions can not be inferred.

If further real time image processing is needed on the acquired images than a contrast restoration procedure must be applied. Several algorithms were proposed in literature for restoring the contrast of the images. These methods can be categorized in two groups: model and non-model based enhancement techniques. Non-model based methods perform image enhancement relying only on the information obtained from the image; such as histogram equalization or adaptive histogram equalization [10], approaches based on Retinex theory [11]. Unfortunately, these methods do not maintain color fidelity and are not suitable for real time computer vision.

Model based contrast restoration techniques can be further divided in two categories: with given depth and unknown depth. When the depth is supposed to be known, this information can be used to restore the original contrast of the image. The authors in [12], [13] and [14] studied different haze removal approaches based on given depth information. The depth is inferred by using the altitude, tilt and position of the camera [12], through the manual approximation of the sky area and vanishing point in the captured image [13] or by approximating the geometrical model of the analyzed image scene [14]. Because the depth information is provided by the user in all these above mentioned approaches and because the obtained depth information is erroneous and unreliable, these methods are not feasible for real world applications.

An Analysis of Datacenter Network Topologies for Running Mixed Workloads

Ioana Giurgiu*
IBM Zurich - Research
Rüschlikon, CH-8803 Switzerland
Email: igi@zurich.ibm.com

***Abstract*—The problem of resource allocation for complex, networked workloads in datacenters is becoming ever more important. Lately, several topology designs have been proposed to address some of the difficulties of efficiently placing such applications within the datacenter, although designing smart, optimized algorithms is still crucial for a complete solution. In this paper, we want to shed some light on how these different architectures affect the quality of resource allocation for mixed workloads, as well as how scalable and versatile they are. First, our study compares three popular datacenter topologies from the perspective of their specifications (i.e., link redundancy, server-to-switch ratio, customization). Second, we consider the resource allocation of mixed workloads and evaluate each topology's performance in terms of placement time and quality, network utilization, application drop rate and maximum achievable load.**

I. Introduction

Typical Infrastructure as a Service (IaaS) providers offer today virtualized building blocks, that vary in virtual machine, storage volume and network capacities, to allow developers to easily deploy their applications and services. On the one hand, while the diversity of virtual instances available for renting matches a large variety of workloads, the basic placement solutions are still treating such instances as being independent from each other. On the other hand, the exploding complexity of modern applications triggers the necessity of breaking them into multiple virtual instances that exchange data according to specific communication models. The back end database access of a 3-tiered application, for instance, dictates the requirement of a virtualized block with high storage and bandwidth capacities. In other applications or for security and robustness reasons, the storage volume might be required to be placed on a different rack than the computational instances, thus introducing additional locality constraints on the virtualized blocks.

Thus, it becomes clear that there exists a discrepancy between what cloud operators offer and what users actually need. No current placement mechanisms are yet able to handle these computational and networking constraints governing how virtual instances should be deployed and connected. In fact, users have little or no control with respect to the layout of their applications on the physical infrastructure. As a result, it is impossible for an application developer to receive guarantees, such as maximum placement time of a virtual instance, or maximum latency between virtual machines or other crucial information for the specific application. Such policies reflect constraints on the available resources and set the expectations on how instances respond to various stimuli in the environment. For example, a high availability policy associated with a collection of virtual instances specifies that these instances should continue running even in the presence of failure at the infrastructure level.

Previously, we have proposed an efficient placement mechanism that is able to allocate resources for such applications (i.e., composed of multiple virtual instances in a graph manner), while managing their capacity demands and locality constraints, as well as ensuring resource balancing at the datacenter level [5]. This placement problem is well known in the community as NP-complete, since it can be reduced from a theoretical perspective to graph isomorphism. Its complexity arises from its combinatorial nature, which results in a search space explosion as the size of the datacenter increases. Given the variety of datacenter architectures proposed in recent years [7], [13], [9], [8], [3], [1] and of applications, we believe several fundamental questions remain unanswered: *In which ways do different datacenter architectures affect the quality of placement mechanisms? How do they scale when various application patterns are considered?* In this paper, we compare and contrast several datacenter modern architectures and choose three, namely balanced tree, Portland and BCube, that are representative within their classes. Additionally, we propose a rich set of application workloads, with CPU, memory, network and availability constraints. Finally, we conduct an empirical study to understand the scalability and versatility features of each datacenter architecture in the face of allocating resources for such mixed workloads.

The remainder of this paper is organized as follows. Section 2 provides a comparison between three popular datacenter architectures. We formulate the placement problem in Section 3. Section 4 presents our experimental study on mixes of application workloads. Finally, in Section 5 we provide an overview of the current state of the art and conclude in Section 6.

II. Datacenter Architectures

We start with a high-level classification of datacenter architectures based on which we narrow our study to three specific ones. The various proposals for next-generation datacenter topologies [1], [3], [13], [7], [8], [9] similarly target the support of high bisection bandwidth between large numbers of servers cost-effectively. However, their approaches in achieving this are quite different. A key dimension along which these models differ is the type of hardware used to forward or process network traffic, as discussed in [14]. The most traditional architectures are **switch-only**, in which the packet forwarding is implemented exclusively using switches. Examples of such topologies are the balanced tree, or the more recent VL2 [7], PortLand [13] and the model proposed

* Work done while being at ETH Zurich.

Methods to Protect and Isolate Application Traces Left on a Computer

George Pecherle
Faculty of Electrical Engineering and Information Technology
University of Oradea
Oradea, Romania
Email: gpecherle@gmail.com

Cornelia Győrödi
Faculty of Electrical Engineering and Information Technology
University of Oradea
Oradea, Romania
Email: cgyorodi@uoradea.ro

Robert Győrödi
Faculty of Electrical Engineering and Information Technology
University of Oradea
Oradea, Romania
Email: rgyorodi@uoradea.ro

***Abstract*—Almost all applications leave traces of user activity on the computer and sometimes this can be confidential information that no one wants to fall into the wrong hands. Traditional software systems wipe these traces beyond recovery, however finding where these traces are saved can be a difficult work and it may leave out traces not detected by classic algorithms. Our proposed method implements another way, of running applications in a protected and isolated environment (like a portable USB flash or a virtual machine). This way, application traces are isolated in their running environment and if needed, this environment can be easily destroyed and/or rebuilt to its initial (clean) state.**

***Keywords*—security, privacy, application traces, data wiping, virtual machines**

I. Introduction

Even though there are many solutions for destroying confidential data, they are not able to get into action exactly in the moment they are needed, for example when data is in danger of being accessed by others.

A phenomenon that is starting to bring more and more attention lately is the identity theft. Essentially, this means that one person tries to pretend to be another one, then performs different actions using that person's identity. These actions can lead to big losses, especially of financial nature, because each time an online transaction takes place, there isn't any face to face interaction between the involved parties. Because of this, anyone can pretend to be someone else, if they know and/or posses the right information.

One of the major problems with software tools that have to locate confidential information in a computer system is the high level of data dispersion. Each application from the system (Web browsers, media players, etc.) hides its confidential information (usually records of user's activity with that application) in different locations, making it really difficult, sometimes even impossible, to find their exact location.

A fast but impractical solution is destroying the entire system that will permanently wipe the confidential information and also remove the operating system and programs. A complete system reinstall will be needed after the wipe process.

Because of this, it would be ideal, if there were a possibility to run the applications and to save the data in a protected environment, isolated from the rest of the main system. If needed, this environment could be self-destroyed or restored to its initial state (the one without the confidential data), without affecting the functionality of the main system.

The idea came from a method used by major antivirus applications to test potentially infected files: the files are run/opened in a protected environment, called "sandbox", to test their effect on the main system, without affecting it. A good example is SafeRun from Kaspersky Internet Security [1].

For this purpose, we have experimented two methods with specific settings and steps to follow, that will help achieve a safe and protected environment for saving confidential data:

- A virtual machine
- A USB flash drive, with portable applications

The virtual machine is a more flexible environment, due to its capability to execute almost any applications from within, just like a real computer.

The USB flash drive with portable applications has a reduced flexibility because of the limited number of applications that can run on it (usually a Web browser, an email client, an Office suite like OpenOffice, various audio-video applications, chat etc.). On the other hand, it is ultra portable, this being the main reason it was created for.

Next, we will demonstrate how to adapt each of these solutions to our problems, following three simple steps:

- Configuring the working environment
- Protecting the working environment against unauthorized access, during its use
- Destruction/self-destruction of the working environment

II. Virtual Machine For Isolating Confidential Information

A virtual machine is a software implementation of a computer, capable of executing programs, the same as a physical machine [2]. Due to its great isolation, against the main system, we will use this environment to control confidential data storage.

For this experiment, we used Windows Virtual PC, on which we installed Windows XP Professional. After the initial settings of the virtual machine (RAM, size of the virtual hard disk, etc.), two files will be created: a file with the VMC extension (Virtual MaChine file, which stores data of the virtual machine, like hardware configuration) and a VHD file (Virtual Hard Disk file, that stores all the data from the virtual hard drive) [5].

Because both files are saved in a location specified by the user, all the data on the virtual hard disk will be isolated in that VHD file, that will help the self-destruction of the data contained in it.

Metaprogramming can be Easy, Fast and Fun

A Plea for Hyper-Metaprogramming

Lucian Radu Teodorescu
Technical University of Cluj-Napoca
lucian.teodorescu@cs.utcluj.ro

Rodica Potolea
Technical University of Cluj-Napoca
rodica.potolea@cs.utcluj.ro

***Abstract*—Compile-time metaprogramming is proven to be a very useful technique in areas like code generation, code optimization or generic programming. A metaprogramming system tends to be more powerful when it has the ability to perform computations at compile-time. However, metaprograms that perform computations at compile-time are usually hard to understand, as they are written with different syntax and they use different idioms compared to regular programs. Moreover, most of the time, the duration of the compilation rapidly increases with the complexity of the metaprogram. The C++ template metaprogramming system, a prominent example of compile-time metaprogramming, is especially known for these problems.**

Hyper-metaprogramming is a concept based on static metaprogramming that aims at providing an easy way to deal with compile-time computations. By its definition, it guarantees that one can write arbitrary complex metaprograms with (almost) the same syntax and the same abstractions as found in traditional run-time code.

The present paper investigates the extent to which hyper-metaprogramming is suited for performing compile-time computations by analyzing two aspects: the ease of writing metaprograms and the increase in compilation time resulted by the execution of metaprograms. As a case study, we employ the same N-Queens problem that Dubrov used to test the compilation speed of C++ template metaprogramming. We compare hyper-metaprogramming with C++'s metaprogramming, and we also evaluate the compile-time execution of a metaprogram in comparison to the run-time execution of the same algorithm.

We prove that, by using hyper-metaprogramming, writing metaprograms is as easy as writing regular run-time programs. These metaprograms are fast to execute, significantly faster than the ones that use C++ template metaprogramming. Furthermore, the execution times for metaprograms are comparable with the ones for run-time code execution.

***Keywords*—Hyper-Metaprogramming, Static Metaprogramming, C++ Templates, `constexpr`, Sparrow**

I. Introduction

It is said that a good workman is known by his tools. The same goes for programming: a good programmer needs to have good development tools. Moreover, a programmer would want to create his own tools. Here is where metaprogramming comes into play: metaprogramming is the writing of programs that generate or manipulate other programs. All along the history of programming, metaprogramming techniques were widely used to ease the work of programmers. A common example of metaprogramming is the use of macros in the C language; the programmer writes some macro definitions that, when used, get expanded into additional code.

If the metaprogramming techniques are applied during the compilation processes like in the C macros example, we say we have *compile-time metaprogramming*, or *static metaprogramming*. Usually, static metaprogramming is more appropriate to static languages that involve the use of compilers. With compile-time metaprogramming, the programmer instructs the compiler (more generally, any tools used to generate the target program) to perform additional actions, like generating new code. Furthermore, in some cases, the programmer may instruct the compilation system to perform complex computations. As a simple example, the user may want to precompute the values for the sin function for all sexagesimal angles at compilation time, before the program starts.

The template metaprogramming from C++ [4] is a system that allows computations to be performed during compilation, by carefully arranging templates. This system was even proven to be Turing complete [33]. To use this feature for a computation, the user must encode the input data as types and integral constants, write a series of templates that manipulate these data, and instantiate the templates with the given data; the results are also given as types and/or integral constants. The execution of the computation follows a purely functional model.

Nevertheless, in practice, template metaprogramming is cumbersome to use [10], [18], [4]. First, writing metaprograms is difficult: the syntax is different than the one of traditional run-time programming, the idioms to be used are different, and even the programming paradigm is different; the C++ langugage was simply not designed for template metaprogramming [23]. If programmers, however, are able to get over these inconveniences, they will encounter a second wave of problems: compilation time increases rapidly with problem size, and compilers have limits regarding the number of templates that can be instantiated, reducing, therefore, the number of computations that can be actually performed [10], [4], [30].

By contrast, Lisp got metaprogramming right. By using dynamic metaprogramming (the manipulation of the program takes place during run-time), it treats code as data, giving the programs the ability to analyze, manipulate or generate other programs. The syntax is the same, the abstractions to be used are the same, and the execution of the code is performed in the same manner. Racket, a programming language derived from Lisp, can execute any code at different moments, even at compile-time [11].

As static languages play an important role in the software industry, we would want, similarly to Lisp and Racket, a clean metaprogramming system in static languages that can execute arbitrarily complex computations during compile-time. The syntax of writing metaprogramming should be simple enough, the idioms and the programming paradigm should be similar to the one the language already provides, and the compilation

Localization and Detection of Pedestrians in Infrared Traffic Scenes

Raluca Brehar, Sergiu Nedevschi
Technical University of Cluj-Napoca
Computer Science Department
Raluca.Brehar@cs.utcluj.ro ; Sergiu.Nedevschi@cs.utcluj.ro

***Abstract*—The information provided by infrared sensors can be useful for pedestrian detection in all types of weather conditions and both at day time and during night. We describe an integrated framework for pedestrian detection based on a monocular infrared sensor. The method comprises two main original contributions: the region of interest reduction and the combination of multi-scale histogram of oriented gradient based cascades for the detection task.**

The region of interest reduction method is based on the analysis of vertical edges of the IR image. We propose a set of filters that remove uniform areas such as the sky or the road and we also try to remove highly cluttered regions such as heated buildings. We also use a scan window reduction method that based on the geometry of the scene provides suitable scan window dimensions for different positions of the scan window.

For the classification task we train eight boosted cascades, each working with a specific size of training images. Hence we follow the so called approach of one image scale and multiple detector sizes. For each size we use an aspect ratio of 0.5. Each cascade is trained until it reaches specific false positive and true positive rates. Negatives' bootstrapping is applied for each stage of the eight cascades.

Another contribution is the setup of a database of infrared pedestrians captured in traffic scenes. Our dataset contains more than 3000 pedestrian models that are used for training the classifiers. A separate sequence has been annotated in order to validate accuracy of the detection of the framework. For each test image we scan the regions of interest with the multi-scale ensemble of cascades and avoid the multiple resize of the image. This leads to an improved execution time.

***Keywords*—Pedestrian detection, night vision, autonomous driving.**

I. Introduction

Pedestrian detection in automotive applications is an extremely active field of research. Most approaches rely on stereo vision for detecting pedestrians at daytime. A limitation of the visual spectrum is encountered at night when stereo vision is not feasible. For night vesion infrared sensors have been used successfuly because they capture the heat emitted by objects. The infrared sensors can also be used at daytime and enhance the accuracy of stereo vision based approaches by sensor information fusion.

The infrared pedestrian appearance is different from the look of a pedestrian in the visual field but the challanges that must be faced by a pedestrian detector hold due to the high variety of appearance given by clothing, accessories, body part positions, viewing angle (front, lateral, rear) and actions (walk, stand, run) performed by pedestrians.

Another difficulty of an IR pedestrian detector is given by the fact that an IR image looks different at summer and at winter. In winter the pedestrian face appears as being lighter while the body is insulated by warm clothes and it is darker. In autumn and spring when temperatures are bellow 25 degrees the pedestrian head and body is more nicely visible because the clothes are less insulating than in winter. Usually the cars and heated buildings are also clearly visible in those conditions, while the road, the sky or other cold objects appear to be darker in the IR image. In summer when temperatures are very high (above 30 degrees) the road and the sky are hotter and appears lighter than the buildings. Depending on the environment temperature the pedestrians might be darker than the background in hot summer days.

To overcome the difference between appearances in summer and winter, a polarity inversion algorithm can applied on IR images taken in hot summer days such that the pedestrians are lighter than the environment.

Knowing these constraints and difficulties in IR pedestrian detection, we bring contributions in two important taks of a generic pedestrian detector:

- The region of interest (ROI) generator that highlights areas having a high probability of containing a pedestrian, hence its generates pedestrian hypotheses.
- The clasifier that refines the pedestrian hypotheses of the ROI generator and keeps only those hypotheses that are very likely to be pedestrians.

The contributions brought by the proposed region of interest generator come from the following considerations:

1) Pedestrians have a certain distribution of vertical edges: making use of vertical edge information we remove uniform areas such as the sky or parts of the road or buildings, some of the vegetation, and we enhance the areas having a high density of connected vertical edges.
2) In an automotive setup closer pedestrians have a large height opposed to far pedestrians that are smaller. We determine in each region of the image the dimension of the scanning windows that must be retained by the region of interest generator.
3) The pedestrian head and legs appear as light spots in the image. We apply a combination of morphologic operations and adaptive thresholding to keep as much as possible of the pedestrian body in the region of interest.

For the classification part we bring contributions by the usage of multi-scale boosted cascades: we use eight different models trained for eight dimensions of the train data. By the sizes of the models we try to cover the entire height dimension of infrared pedestrians captured by the IR sensor that provides images having a resolution of 320×240 pixels.

We also create a dataset of pedestrian models in infrared images (as we have not found simmilar datasets in literature to be freely available and no benchmarking procedures exists for IR pedestrian detection). The dataset is created from

Bayesian Belief Network for Arteriosclerosis

Călin CENAN
Department of Computer Science
Technical University Cluj-Napoca
Cluj-Napoca, Romania
Calin.Cenan@cs.utcluj.ro

Abstract— **Data mining is concerned with extracting knowledge from databases using machine-learning techniques. We consider that the increasing number of distributed database dispersed over many sites makes necessary to adopt new techniques to improve the overall system response even in the data-mining area. In this paper we present a methodology for discovering knowledge using a Bayesian belief network representation with practical exemplification in the diagnosis of the arteriosclerosis cardiovascular disease. The paper considers the problem of learning the structure and parameters of such a Bayesian belief network used for medical diagnosis. We will propose also an architecture in which machine learning agents use distributed datasets in solving such a knowledge discovering problem.**

Keywords: Bayesian belief networks, machine learning, distributed systems, medical diagnosis

I. INTRODUCTION

Companies are concerned in increasing the flexibility of developing applications using standards to achieve interoperability, and to manage their infrastructure resources (processors, networks, storage, applications) efficiently by taking advantage of new business models and system management techniques, including distributed databases. Enterprises are adopting new approaches to distributed computing to meet earlier and nowadays needs.

In the medical field the research results and the cures for critical diseases do not spread fast and wide enough compared to increasing number of cases. Considering these we hope that a distributed database approach can improve medical results. We can conclude that this practical exemplification suits well the growing needs for the healthcare domain, that the world gathered information can lead to better treatments, to a decreasing number of fatalities, and so to a healthier society.

In this paper, we look at such an application based on an available medical data set that asserts the risk for arteriosclerosis cardiovascular disease. The aim of our study was to identify arteriosclerosis risk factors prevalence in a population generally considered to be the most endangered by possible arteriosclerosis complications, i.e. middle aged men. Our method of work will try to discover knowledge, represented as Bayesian belief network, from distributed databases or data sets as we imagine that are in the reality.

Considering the problem of learning the paper will shortly introduce the knowledge representation method, the Bayesian belief networks, and then the possible distributed sources of data. Networks of this form are frequently used models for expert systems and include the well-known Quick Medical Reference (QMR-DT) model for medical diagnosis [30]. After these we will present some results obtained during the learning process. Maybe these results will generate some interest even in the medical domain from where we obtained out data sets. In the final part we will present the proposed distributed architecture for learning and the results and performance of learning in these situations.

II. DISTRIBUTED DATABASES

Distributed problem solving is a subfield of distributed artificial intelligence in which the emphasis is on getting agents to work together in order to solve the problem. Due to an inherent distribution of resources (knowledge, capability, information, and expertise) among various participants in such a distributed problem-solving system agents are unable to accomplish their own tasks alone. Or at least can accomplish its tasks better, more quickly, completely, precisely, or certainly when working with others. Solving such problems demands both group coherence (agents need to want to work together) and competence (agents need to know how to work together).

A database that consists of two or more data files located at different sites on a computer network it is a distributed database. Because the database is distributed, different users can access it without interfering with one another. However, the database management system must periodically synchronize the scattered databases to make sure that they all have consistent data [14].

In a distributed database system, the database is stored on several computers. The computers in a distributed system communicate with one another through various communication media, such as high-speed networks and they do not share main memory or disks. The computers in a distributed system are referred to by names, such as sites or nodes. The main differences between shared-nothing databases and distributed ones are that distributed databases are typically geographically separated, are separately administered, and have a slower interconnection.

There are several reasons for building distributed database systems, including [21]:

- Sharing data. The major advantage in building a distributed database system is the provision of an environment where users at one site may be able to access the data residing at other sites.
- Autonomy. The primary advantage of sharing data by means of data distribution is that each site is able to retain a degree of control over data that are stored locally. In a centralized system, the database administrator of the central site controls the database. In a distributed system, there is a global database administrator responsible for the entire system. A part of these responsibilities is delegated to the local database administrator for each site. Depending on the design of the distributed database system, each administrator may have a different degree of local autonomy. The possibility of local autonomy is often a major advantage of distributed databases.

Efficient Evaluation of Hyper-Rectangular Blocks of Update Operations Applied to General Data Structures

Mugurel Ionuţ Andreica*, Andrei Grigorean†, Andrei Pârvu* and Nicolae Ţăpuş*
*Computer Science Department
Politehnica University of Bucharest,
Splaiul Independenţei 313, sector 6, Bucharest, Romania, RO-060042
Email: mugurel.andreica@cs.pub.ro, andrei.prv@gmail.com, nicolae.tapus@cs.pub.ro
†Faculty of Mathematics and Computer Science,
University of Bucharest,
Str. Academiei 14, sector 1, Bucharest, Romania, RO-010014
Email: andrei.grigorean@gmail.com

***Abstract*—In this paper we present novel solutions for the following problem: We have a general data structure DS and a set of update operations organized into a D-dimensional cube of side N (thus, there are N^D update operations). We are interested in efficiently evaluating range queries of the following type: compute the result of applying all the update operations within a hyper-rectangular block of the D-dimensional cube to DS (considering that DS is initially empty). The result of applying the updates consists of computing some aggregate values over the data structure. We consider that the order of applying the updates is irrelevant (i.e. the update operations are commutative) and that the aggregate results corresponding to a block of updates cannot easily be computed by combining the results of a set of sub-blocks whose disjoint union is B. However, the results can be efficiently maintained after each update operation, if the operations are performed sequentially in any order.**

***Keywords*—data structures, hyper-rectangular blocks of updates, sequence of updates, range query, block partitioning.**

I. Introduction

Applying a large number of update operations to a data structure and afterwards computing an aggregate result is an important scenario which has not received sufficient attention in the scientific literature so far, particularly when the results of different sets of updates cannot be easily aggregated. In this paper we address the situation in which the update operations are placed in the cells of a D-dimensional cube of side length N and we want to efficiently evaluate the result of the application of a subset of these update operations on an initially empty data structure. The subset of update operations consists of a hyper-rectangular block of the D-dimensional cube containing the update operations. We will consider both the online and the offline case and we will also place emphasis on scenarios where D is small (e.g. when $D = 1$ the update operations are placed in a sequence and we are interested in applying contiguous subsequences of update operations to the data structure). Our results hold for any type of data structure and any type of updates specific to it. In this paper we will present solutions which are capable of evaluating the result of the applications of the update operations without actually applying each update operation on the data structure each time. In order to achieve this we first need to preprocess the D-dimensional cube of update operations and precompute multiple values.

The rest of this paper is structured as follows. In Section II we define the problem statement clearly. In Section III we present a solution for the online case (i.e. the queries are answered as they come, one at a time). In Section IV we improve the memory requirements of the solution presented in Section III for the case $D = 1$ and all the queries are available offline. In Section V we discuss several applications of our solutions. In Section VI we present experimental evaluation results for the solutions proposed in this paper. In Section VII we discuss related work and in Section VIII we conclude and discuss future work.

II. Problem Statement

We consider that we have a data structure DS on which we can apply certain update operations. The update operations are placed in the cells of a D-dimensional cube of side length N. We will denote by $Op(c(1), \ldots, c(D))$ the update operation located in the cell $(c(1), \ldots, c(D))$ of the cube ($1 \leq c(i) \leq N$, $1 \leq i \leq D$). The data structure is capable of efficiently maintaining some aggregate result values after applying each update operation. The result values are independent of the order in which a given subset of update operations are applied (i.e. the update operations are commutative).

We are interested in efficiently answering queries of the following type: Given a hyper-rectangle $\prod [l(i), h(i)]$ ($1 \leq l(i) \leq h(i) \leq N$, $1 \leq i \leq D$), apply all the update operations $Op(c(1), \ldots, c(D))$ with $l(i) \leq c(i) \leq h(i)$ ($1 \leq i \leq D$) to an initially empty data structure and return the result values maintained by the data structure after applying all the operations.

III. Online Solution

We will consider that the side of the cube in each dimension is split into groups of size K (except possibly for the last group, which may contain fewer than K elements). We will define $G(i) = (i-1)/K + 1$ as the group to which the coordinate i belongs. We will now consider the D-dimensional cube CG of side length $(N + K - 1)/K$, where $CG(c(1), \ldots, c(D))$ is a sub-cube of the original cube consisting of the operations $Op(c'(1), \ldots, c'(D))$ such that $G(c'(i)) = c(i)$ ($1 \leq i \leq D$). Note that we consider integer division throughout this paper.