International Journal of Computer Applications (0975 8887)
Volume 101 - No. 6, September 2014

Intelligent Linear Data Structure with Self Performance Optimization Capacity: Application on Big Data

[1]Smail TIGANI, [2]Mouhamed OUZZIF, [3]Abderrahim HASBI and [4]Rachid SAADANE
[1]RITM/ESTC, National High School of Electricity and Mechanics. Casablanca, Morocco
[2]RITM/ESTC, High School of Technology. Casablanca, Morocco
[3]RSE/EMI, Mohamadia School of Engineering. Rabat, Morocco
[4]SIRC/LaGeS-EHTP, Hassania School of Labor Works. Casablanca, Morocco

ABSTRACT

The access to relevant information from a big data container is gaining immense significance. This depends on storage technics and the organization level. This work proposes an intelligent linear data structure with an integrated cognitive agent reorganizing periodically the data structure content. The reorganization is based on a confidence interval of a random variable estimated by the agent. This random variable represents the demand frequency for each element. The cognitive agent studies the client behavior and puts most popular data in the beginning of the array in order to be found quickly. That increases considerably the search algorithm performance and solves by that one of most problems of the big data field. Models and algorithms in this work are implemented with Java programming language and simulated and that proves the reliability of the approach.

Keywords:

Data Structures, Big Data, Statistic Modelling, Algorithms and Computational Complexity, Auto-adaptive Systems, High Performance Systems.

1. INTRODUCTION

The Web 2.0 described by D. Dougherty (2003) allows internet end-users to interact with web pages content and exchange millions of comments, pictures and also massive data over social networks and personal pages... By that we mean that Modern information systems needs processing a huge quantity of data and manage an important network traffic : it is the Big Data. The big quantity of data presents an asset for inference algorithms by increasing precision and in the same time harms the performance witch is a very important aspect that many researches are focused on and still up to now. Z. Yang and al (2010) talks, in (1), about one of the most important aspect of information system performance : it is the availability of data and servers hosting those data.

This paper presents a new linear data structure having the ability to reorganize the data is self in order to optimize the performance of operation research of data on it. By self reorganize we means, the determination of most popular data subset and put them on first ranges of the list to be found quickly. The estimation of popular and non popular data is done by un internal agent making the perception of the client behavior and then estimate some statistical parameters on which the decision making relies.

The content of his paper is organized as follows : the section 2 introduces some related works and some critics with discussions, while section 3 presents the statistical models. Algorithms design and computational complexity analysis are reported to the section 4, while simulation results and curves with comments are presented in section 5. Finally, Section 6 presents concluding remarks followed by discussion of future work.

2. RELATED WORKS

In (3), R. Gupta (2014) talked about the evolution time line of intelligent information technologies from the data mining fields moving to the web mining arriving actually (2014) to the big data technology. By big data we means all technics allowing the processing of very huge quantity of data and the extraction of useful information keeping performance aspects. That comes to design intelligent data container able to assure the availability of data as described by Z. Yang and al (2010) in (1). One of works touching to the all those sides are works of I. Ioannidis and al (2003) who described in (2) the idea of **adaptive data structures** providing performance guarantee as an IP addresses lookups application.

K. Greer and al (2009) introduces the idea **self-organising infrastructure** of services in (5) and, in the same year, D. Lefebvre presents in (7) some challenges for **adaptive systems**. The discussed last works allows us to think about the incorporation of the auto-adaptive aspect in a linear data structure in order to reorganise the content of it and then optimize some performance indication to define posteriorly.

The organization of the content of the data structure must be done based on some criteria, this is the reason of the choice of the confidence interval of the random variable representing the average of demand of each element on the data structure. M. Tanusit present in (8) the technics of estimations of a two-sides confidence intervals for a poisson means. In this case, the judgment that the means of demand follows Poisson law seems to be impossible. Knowing the average $\overline{X}$ denoted X^{e_i} and the variance of the sample denoted s_n, that allows to build the confidence interval using works of S. Niwitpong and al (2013) found in (9) and statistical reference (11) of L. Lebart and al (1995).

3. MATHEMATICAL MODELLING

Designed algorithms in this work are based on some mathematical models. This section focuses on statistical modelling aspects and explains also the classification criteria of data in the array.

3.1 Basic definitions

Let consider a finite state $\mathbb{E}$ containing all data in the linear data structure. In theory, the data is represented with elements $e_1 ... e_m$ and, in practice, with a Java class introduced in the appendix 2. Formally :

$$\mathbb{E} = \{e_i\}_{i=1}^{i=m} \tag{1}$$

Let consider a random experience repeated n times : the experience is looking for an element from $\mathbb{E}$ defined previously. Let X^{e_i} be the random variable representing the average of the demand of the element e_i during all the process and m_n the mean of X^{e_i}. Mathematically :

$$m_n = \frac{1}{n}\sum_{k=1}^{n} \mathbf{1}_{\{e_i = e_k^*\}} \tag{2}$$

With e_k^* is the researched element at the instant k. The function $\mathbf{1}_{\{e_i=e_k^*\}}$ is equals to 1 if $e_i = e_k^*$ and equals to 0 if not. Let's call also m_n and s_n the average and the standard deviation, respectively, of the variable X^{e_i}.

3.2 Probability Law

This subsection discuses the probability law of the variable X^{e_i}. Based on this low, the confidence interval of the previous variable is build with a fixed confidence rate : 90%, 95%... S. Niwitpong and al (2013) affirms in (9) that with unknown standard deviation σ. That comes to use a random variable with Student law based on non-biased estimator s_n. Formally :

$$\frac{X^{e_i} - m_n}{s_n/\sqrt{n}} \backsim T^{n-1} \tag{3}$$

With T^{n-1} is the Student distribution with $n-1$ degrees of freedom.

3.3 Confidence Interval Estimation

The confidence interval of X^{e_i} denoted $\zeta_{95}(X^{e_i})$ with $LB(\zeta_{95}(X^{e_i}))$ is the left bound of the interval $\zeta_{95}(X^{e_i})$ and $RB(\zeta_{95}(X^{e_i}))$ the right one. See the proof on appendix 1. Formally :

$$\zeta_{95}(X^{e_i}) = \left[\underbrace{m_n - 1.96\frac{s_n}{\sqrt{n}}}_{LB(\zeta_{95}(X^{e_i}))}, \underbrace{m_n + 1.96\frac{s_n}{\sqrt{n}}}_{LB(\zeta_{95}(X^{e_i}))}\right] \tag{4}$$

3.4 Classification Criteria

In order to optimize the time of research in the linear array, The internal agent reorganizes periodically the content of the array. The organization will be done by moving most demanded class of data to the beginning of the array, and the less demanded at the end.
The CI[1] of the average of demand allow the agent to pin point the element having the biggest chance to appear in next demand. **The CI of demand average takes in consideration the popularity of the element and also the stability of it. By stability we mean, the homogeneity of the element demand, it is represented by the standard deviation. It is evident so that the element having the biggest lower bound of the CI is the most demanded stable element. This one needs to be moved in first range. Finally the agent makes an descending sort of all element of the array by the lower bound of the CI.**

4. ALGORITHMS DESIGN

4.1 Reorganization Algorithm

In this section is the design of algorithms implementing models discussed in section 3. The algorithm 1 reorganizes data in the array according to the left bound of the confidence interval $\zeta_{95}(X^{e_i})$: the algorithm makes a descending sort of elements according to the worst frequency of demand[2]. That makes most demanded elements reorganized in the beginning of the array in order to be found quickly. See the algorithm :

Algorithm 1: Data Reorganizer

Input: $\mathbb{E}$: Data Set

for $i = 1$ **to** $m - 1$ **do**
  for $j = i + 1$ **to** m **do**
    if $LB(\zeta_{95}(X^{e_i})) < LB(\zeta_{95}(X^{e_j}))$ **then**
      $vTmp \leftarrow e_j$
      $e_j \leftarrow e_i$
      $e_i \leftarrow vTmp$

Computational complexity of the algorithm 1 is quadratic $\mathcal{O}(m^2)$.

4.2 Simulation Algorithm

In order to evaluate the performance and the reliability of the approach, this work proposes three key performance indicator to compute during the simulation. Those keys are the cost of research after each research operation, the average of total costs after all the process of simulation and finally the cumulative cost. See the next section for more discussions.

The algorithm 2, whose computational complexity is $\mathcal{O}(mn)$, generates a fixed number of data and store them, one by one, in a traditional array in one hand and proposed intelligent array in the other hand. The testing data are stored initially with the same way in the two arrays. The next step is making a search operation of the same randomized value in the two arrays and collecting costs in each case in order to compare them. This last operation is repeated quite a few times to have enough data to plot curves.

[1] Confidence interval.
[2] Lower Bound of the Confidence Interval.

Algorithm 2: Performance Simulator

Output: $iCost$: Costs on Intelligent Array
Output: $sCost$: Costs on Simple Array

// Testing Data
for $i = 0$ **to** n **do**
  $sArray[i] \leftarrow i$
  $iArray[i] \leftarrow i$

// Researched Data
for $i = 0$ **to** m **do**
  // Random Data with Normal Law
  $e \leftarrow GenerateRandomValue()$
  for $j = 0$ **to** n **do**
    if $sArray[j] \neq e$ **then**
      // Cost on simple array
      $sCost[i] \leftarrow sCost[i] + 1$
    else
      $break$
  for $j = 0$ **to** n **do**
    if $iArray[j] \neq e$ **then**
      // Cost on intelligent array
      $iCost[i] \leftarrow iCost[i] + 1$
    else
      $break$
  // Reorganize data for each 100 opertaion
  if $m = 0[100]$ **then**
    $DataReorganizer()$

4.3 Normal Law Generation Algorithm

The function *GenerateRandomValue()*, called in the algorithm 2, generates values with normal law with a specific mean and variance. Most programming languages allows the generation of uniform random variable that are transformable to an other probability law using different methods. Let consider U_1 and U_2 two uniform random variables on $[0, 1]$ and X a given random variable. K. Ranga and al (2011) presents, in (12), the method introduced by Box-Muller (1958) allowing the generation of normal law $X \sim N(\mu, \sigma^2)$ with :

$$X = \mu + \sigma \sin(2\pi.U_2)\sqrt{-2ln(U_1)} \quad (5)$$

The algorithm 3 is able to generate random values with gaussian law, it is introduced in order to make simulation with a normal distribution. This algorithm relies on Box-Muller method explained in equation 5. See the algorithm :

Algorithm 3: Normal Distribution Generator

Output: μ : Desired mean
Output: σ : Desired standard deviation
Output: $X : X \sim N(\mu, \sigma^2)$

// All programming language provide this function
$U_1 \leftarrow UniformeRandomValue()$
$U_2 \leftarrow UniformeRandomValue()$

// Box-Muller method
return $\mu + \sigma \sin(2\pi.U_2)\sqrt{-2ln(U_1)}$

5. SIMULATION RESULTS

5.1 Research cost KPI

The algorithm 2 making the simulation computes the arrays $iCost$ and $sCost$ containing costs to find the same data in the intelligent array and the simple array respectively. In order to assure the visibility of curves in the figure, The algorithm makes sampling each 100 operation research and that gives the figure 1 :

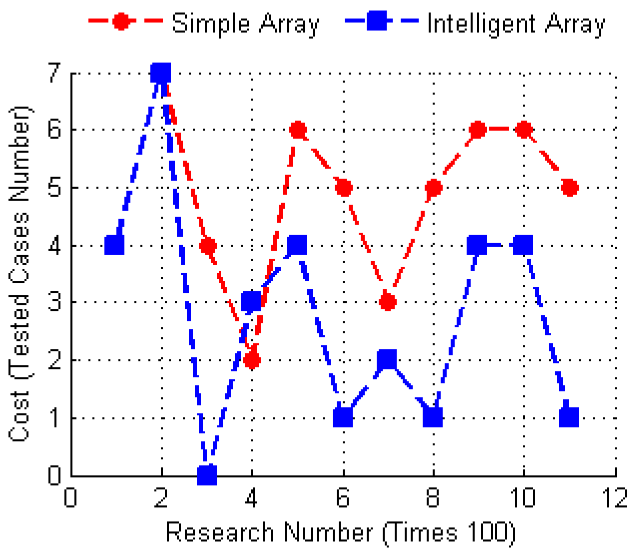

Fig. 1: Research Cost Variation

The figure 1 shows that the adoption of the proposed intelligent array optimizes considerably after a small training period.

5.2 Cost average KPI

Based on the two arrays $iCost$ and $sCost$, Let define the average of cumulative cost on the intelligent array Λ^i in all the simulation process on m operation research as :

$$\Lambda^i = \frac{\sum_{k=1}^{m} iCost[k]}{\sum_{k=1}^{m}(iCost[k] + sCost[k])} \quad (6)$$

With the same way, the average of cumulative cost on the simple array Λ^s during all the process is the following :

$$\Lambda^s = \frac{\sum_{k=1}^{m} sCost[k]}{\sum_{k=1}^{m}(iCost[k] + sCost[k])} \quad (7)$$

The simulation program affirms in all cases that $\Lambda^i < \Lambda^s$. The table 5.2 shows the result of cumulative cost average comparative study between the two approaches after 1100 iteration. See the table :

Cumulative Cost Average	$n = 1100$
Λ^i for Intelligent Array	42%
Λ^s for simple Array	58%

5.3 Cumulative Cost KPI

Let $\theta^i(n)$ be the cumulative cost for the intelligent array until the n^{th} operation research defined in the equation 8. Formally :

$$\theta^i(n) = \sum_{k=1}^{n} iCost[k] \tag{8}$$

With he same way, let define $\theta^s(n)$ for the simple array given by the equation 9. Formally :

$$\theta^s(n) = \sum_{k=1}^{n} sCost[k] \tag{9}$$

The figure 2 represents the variation of $\theta^s(n)$ and $\theta^i(n)$ during the process of simulation. See the figure :

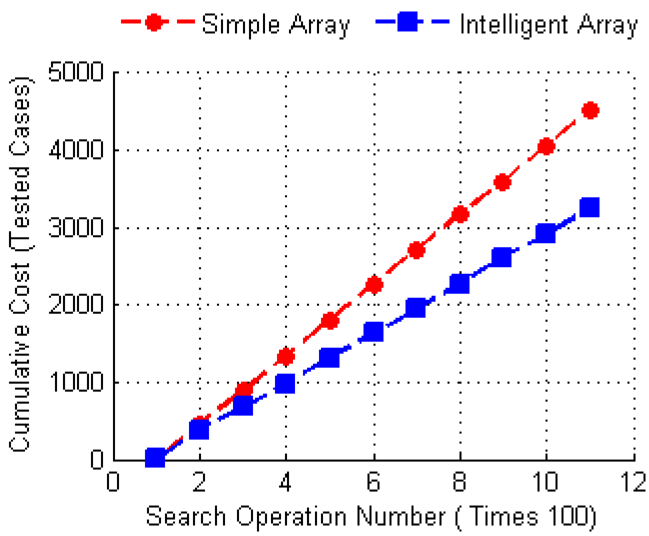

Fig. 2: Cumulative Cost Variation

Based on previous KPI, The figure 2 shows that the gain on response time is increasing using the proposed intelligent array.

6. CONCLUSION

This work proposes an intelligent array having the capacity to self-organize its content in order to optimize some performance aspects. The reorganization is based on the unilateral confidence interval of a random variable representing the average of demand of each element of the array. The approach is validated by simulation and results are more competitive comparing than the traditional approach.

This work presents also the algorithms designed implementing adopted statistical models and the algorithm of simulation designed and implemented in order to prove the reliability of the approach. Algorithms in this work are implemented with Java programming language following the Java reference of B. Eckel (2006) described in (13). The approach is validated by simulation and that gives satisfying numerical results.

Next works will continue on this way by using more advanced automatic learning technics to optimize complex systems efficiency and then increase performance.

APPENDICES

6.1 Appendix 1 : Proof on confidence interval.

In order to find the upper and lower bound of X^{e_i} with $100(1-\alpha)\%$. Let suppose :

$$\pi(t^{n-1}_{\alpha/2} \leq \frac{X^{e_i} - m_n}{s_n/\sqrt{n}} \geq t^{n-1}_{1-\alpha/2}) = 1 - \alpha \tag{10}$$

With π is a probability function. Let fixe $\alpha = 0.05$, according to Student law table, we have $T^{n-1}_{1-\alpha/2} = -T^{n-1}_{\alpha/2} = 1.96$ for an infinite freedom degree. The choice of infinite freedom degree is due to the big quantity of data that will be stored in the array. Some simplifications of equation 5 gives :

$$\pi(\underbrace{m_n - 1.96\frac{s_n}{\sqrt{n}}}_{LB(\zeta_{95}(X^{e_i}))} \leq X^{e_i} \leq \underbrace{m_n + 1.96\frac{s_n}{\sqrt{n}}}_{RB(\zeta_{95}(X^{e_i}))}) = 1 - \alpha \tag{11}$$

Based on equation 11, a percentage of 95%, gives $X^{e_i} \geq LB(\zeta_{95}(X^{e_i}))$ and $X^{e_i} \leq LB(\zeta_{95}(X^{e_i}))$.

6.2 Appendix 2 : Data container with Java.

In order to implement the approach with a programming language, let develop a Java class to contain necessary parameters. See proposed class denoted $CData$:

```
import java.io.FileWriter;
import java.io.IOException;
import static java.lang.Math.*;
import java.util.ArrayList;
import java.util.Random;


/*
    @author : Smail TIGANI
    @version : 1.0
*/
class CData {

    // Data
    private int element;
    // Avarage of demand of the data
    private double average;
    // Standard deviation of demand
    private double SDeviation;
    // Left bound of the confidence interval of demand
    private double LeftBoundUIC;
    public CData() {
    }
    public CData(int element) {
        this.element = element;
        this.average = 0;
        this.SDeviation = 0;
        this.LeftBoundUIC = 0;
    }
    public int getElement() {
        return element;
    }
    public void setElement(int element) {
```

```
        this.element = element;
    }
    public double getAverage() {
        return average;
    }
    public void setAverage(double average) {
        this.average = average;
    }
    public double getSDeviation() {
        return SDeviation;
    }
    public void setSDeviation(double SDeviation) {
        this.SDeviation = SDeviation;
    }
    public double getLeftBoundUIC() {
        return LeftBoundUIC;
    }
    public void setLeftBoundUIC(double LeftBoundUIC) {
        this.LeftBoundUIC = LeftBoundUIC;
    }
}
```

Acknowledgment

I would like to express all my gratitude to my supervisors Dr Mohamed OUZZIF, Dr Abderrahim HASBI and Dr Rachid SAADANE for excellent human behavior and technical support. Special thank to the Director of RITM Lab and the Director of High School of Technology - Casablanca, Dr Mounir RIFI.

Authors would like also to thank all reviewers for helpful comments and recommendations.

Thank to Dr Hafid GRIGUER, Mr Anis BOULAL, Miss Assya BENHIMA and Dr Hicham LAALAJ for all efforts done to encourage scientific research in EMSI Rabat.

Biography

Smail TIGANI : Is a network and telecommunication systems engineer and Phd Student in artificial intelligence and systems modelling. He worked as software engineer and now an information systems engineer and professor at Moroccan School of Engineering Science in Rabat. Recently, his researches focuses on the application of artificial intelligence on big data and performance analysis and optimization.

Mohammed OUZZIF : Is a Professor of Computer Science at High School of Technology of the Hassan II University. He has prepared his PHD at Mohammed V University in collaborative work field. His research interesting concerns distributed system and Formal description.

Abderrahim HASBI : Is a Professor of Computer Science at the Mohammadia School of Engineering of the University Mohamed 5 Agdal, Morocco. He is member of the Network and Intelligent systems Group and he has a lot of contributions researches.

Rachid SAADANE : He is currently an Associate Professor in the Electrical Engineering Department at Hassania School of Labor Works of Casablanca, Morocco. His research interests include array of UWB channel measurements modeling and characterization, mobile and wireless communications (GSM, WCDMA, TD/CDMA, LTE and LTE-A) and finally digital signal processing for wireless communications systems. Recently, he is intensively interested to the IR-UWB physical layer for WSN and WBAN. Rachid is an active reviewer of various international conferences and journals.

References

[1] Z. Yang, J. Tian and Y. Dai, *Towards a more accurate availability evaluation in peer-to-peer storage systems*, International Journal of High Performance Computing and Networking, Vol. 6, Nos. 3/4, 2010.

[2] I. Ioannidis, A. Grama and M. Atallah, *Adaptive Data Structures for IP Lookups*, IEEE INFOCOM, Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies, Vol. 1, pp 75 - 84, 2003.

[3] R. Gupta, *Journey from Data Mining to Web Mining to Big Data*, International Journal of Computer Trends and Technology, Vol. 10, No. 1, October 2014.

[4] A. Habibizad Navin, M. Naghian Fesharaki, M.K. Mirnia and M. Teshnelab, *A Novel Method for Improving the Uniformity of Random Number Generator Based on Data Oriented Modeling*, International Journal of Computer Science and Network Security, Vol.7 No. 7, July 2007, pp 269-273.

[5] K. Greer, M. Baumgarten, M. Mulvenna and C. Nugent, *An infrastructure for developing self-organising services*, International Journal of Adaptive and Innovative Systems, Vol. 1, No. 1, 2009, pp 88-103.

[6] M. Beck Rutzig and A. Carlos Schneider Beck, *An infrastructure for developing self-organising services*, International Journal of High Performance Systems Architecture, Vol. 4, No. 1, 2012, pp 13-24.

[7] D. Lefebvre, *Introduction : Some challenges for adaptive and innovative systems in the next future*, International Journal of Adaptive and Innovative Systems, Vol. 1, No. 1, 2009, pp 1-12.

[8] M. Tanusit, *Two-Side Confidence Intervals for the Poisson Means*, International Journal of Modeling and Optimization, Vol. 2, No. 5, October 2012.

[9] S. Niwitpong and S. Niwitpong, *On Simple Confidence Intervals for the Normal Mean with a Known Coefficient of Variation*, International Journal of Mathematical, Computational, Physical and Quantum Engineering Vol:7 No:9, 2013, pp 1-12.

[10] M. Tanusit, *A Novel Method for Improving the Uniformity of Random Number Generator Based on Data Oriented Modeling*, International Journal of Computer Science and Network Security, Vol.7 No. 7, July 2007, pp 269-273.

[11] L. Lebart, A. Morineay and M. Piron, *Statistique Exploratoire Multidimensionnelle*, ISBN 2 10 0028863, Dunod, Paris, 1995.

[12] K. Ranga, N. Kumar and M. Reddy, *Generation of standard normal random variables*, Indian Journal of Scientific Research, No. 4, Vol. 2, pp. 83-85, 2011.

[13] B. Eckel, *Thinking in Java 4th Edition*,ISBN 978-0131872486, Prentice Hall, 2006.