2PROM: a Two-Phase image Retrieval Optimization on dataspace using predictive Modeling

Gile Narcisse FANZOU TCHUISSANG*‡, Ning WANG*, Nathalie Cindy KUICHEU*, François SIEWE†, and De XU*
* School of Computer and Information Technology, Beijing Jiaotong University, China.
fanzounar2002@yahoo.fr, nwang@bjtu.edu.cn (Correspondence Author), nathkuicheu@yahoo.fr, dxu@bjtu.edu.cn
†Software Technology Research Laboratory, Faculty of Technology, De Montfort University, United Kingdom
fsiewe@dmu.ac.uk
‡IBM Security Center, Montreal, Canada
narcisse@ca.ibm.com

***Abstract*—This paper introduces 2PROM, a new algorithm that can efficiently retrieve information from a set of multimedia and heterogeneous data sources. We published IHM, a model to predict whether a x-ray picture carries the trace of cancer (viruses). We further generalized our approach to The NoCancerSpace, a dataspace for medical diagnosis of lung cancer. This paper presents the optimization algorithm used by the NocancerSpace. 2PROM is designed to optimize the Dataspace retrieval process with two main phases. Its first phase consists of building a pipeline to find the best retrieval strategies. In fact, the pipeline explores the set of alternative execution strategies to determine the cheapest one. The retrieval strategies are initial nodes of the next phase. As for the second phase, retrieval strategies are combined with predictive model to determine the most efficient way to execute a query. In order words, the optimizer considers the possible retrieval strategies for a given input query, and attempts to determine which of those strategies will be the most efficient. The retrieval strategies are represented as XML tree of "strategy nodes". The output of the second phase is the best results found. Experiments show that 2PROM retrieves more relevant results in less time than existing systems.**

Keywords: Information Retrieval, Optimization, Algorithm, Dataspace, Predictive Modeling

I. INTRODUCTION

Predictive modeling is a process used to create statistical models for future behavior. A predictive model is made up of a number of predictors, which are variable factors that are likely to influence future results. Compared with conventional models, it has the advantage of generating compromised management strategies.

AdaBoost[1] integrates support vector machine and Bayes classifier for classification. Abiteboul [2] proposed OptimAX, an Optimizer for Active XML documents (AXML). AXML are XML documents, a transition between data, queries and query plans. Chen [3] presents TPOA, a two-phase optimization algorithm for Mastermind. TPOA solved the problem of Mastermind by using an optimal strategy for the worst case and the expected case. Jiratta [4] introduces a Two-Phase Optimization (2PO), which is a combination of iterative improvement with multiple views processing. Henriques [5] proposes a two-phase algorithm, to create clusters by analyzing a set of trajectories and improve the optimized solution.

Most of these existing optimization works focused their research on combining many predictive models and on building query plans. Unfortunately, query plans often cannot take advantage of pipelining because of limited buffer or CPU resources. Moreover, handling query plans during optimization needs large uphill moves. A dataspace is a set of multimedia and heterogeneous data sources, process queries in parallel seems to be the best way to retrieve quickly information. That's why we propose to use the pipeline to process queries on NoCancerSpace.

This paper proposes 2PROM, a pipeline based algorithm that processes information in parallel, and so retrieves information in the most rapid way. 2PROM uses IHM[6], to predict the data sources where the query result can be found, and collects feedback from satisfied users. If users are not satisfied or if the system did not find any result to the posed query, 2PROM stores the queries to optimize query processing. Users do not have to repeat their query if it is not satisfied. Indeed, saving queries could share common computation so to reduce the I/O cost significantly.

The 2PROM algorithm is distinguished to others in the following way. First, it uses IHM to predict the data sources where the query result can be finding. Second, it uses a pipeline to process information in parallel, so as to optimize the dataspace retrieval process.

The rest of this paper is organized as follows: section 2 presents an overview of the NoCancerSpace, the 2PROM algorithm is describes in section 3, section 4 presents the experimental results and section 5 concludes the paper.

II. BACKGROUND: THE NOCANCERSPACE

NoCancerSpace stand for "A World without Cancer", that is a dataspace for medical diagnosis of lung cancer, a predictive chest x-ray pictures space using user hidden information. The primary objective of this work is to provide users with diagnoses and possible treatment of lung cancer. XIRS[7][8] allows searching for x-ray pictures by similarity, using a simple keyword or an image file as request, see figure 1. In addition IHM is used to predict whether an x-ray picture carries the trace of cancer (viruses).

978-1-4673-2197-6/12/$31.00 ©2012 IEEE

Since in most lung cancer cases, 4 of 5 patients are not aware that they have the disease, symptoms are hidden and that's why IHM uses three learning strategies to propose a diagnosis: - UHB holds the user medical history, - UHS is the user hidden semantics, - UHK is the user hidden meta-data. The NoCancerSpace helps users to avoid lung cancer or, to have possible treatment of their cancer. This could help physician to confirm his diagnosis and his treatment, with geographical location of ingredients. This application is also open to traditional practitioners who often cannot read or write, so cannot create archives of medication (treatment).



Fig. 1. *Searching the NoCancerSpace*

We recently published IHM, a predictive information retrieval model for dataspace. IHM is the process by which a model is created to best predict the probability of having cancer viruses on a set of chest x-ray pictures. Predictions are made and the model is validated (or revised) as additional data becomes available. Readers are referred to [6] for a full presentation of the model. The user query is decomposed into Elementary Sub Queries (ESQ) and the Retrieval Status Value (RSV) is used to find which ESQ can be finding in which Data Source (DS). A DS is a set of domains (D). An IHM is a rewriting model denoted as (DSP, Q, O):
-The Dataspace: DSP=$\{DS_1, DS_2, \ldots, DS_n\}$.
-The ESQ: Q=$\{^{q}1, ^{q}2, \ldots, ^{q}m\}$, $m \leq n$.
-Output results (Observations): O=$\{O_1, O_2, \ldots, O_m\}$.
$IHM : (DSP, Q) \rightarrow O; (DS_i\ ,\ q_i\ ) \rightarrow$ **IHM**$_i$.

This paper presents the optimization algorithm used by the NocancerSpace.

III. A PIPELINE BASED TWO-PHASE ALGORITHM

2PROM is a new algorithm, designed to act like an oracle with two main phases.

In phase 1, starting from a lowest precision of a retrieval strategies $^{q}1D_{11}$ = RSV(q_1, D_{11}), a pipeline of "m" stages and "$m \cdot n$" instructions is constructed as on figure 2. At t = n, all stages of the pipeline are solicited, and m operations take place simultaneously. Then, at each clock of time, we do apply one retrieval strategy and then explore the set of alternative execution strategies to determine the cheapest one.

In phase 2, retrieval strategies are combined with Predictive Model to determine the most efficient way to execute a query. The optimizer considers the possible retrieval strategies for a given input query, and attempts to determine which of those strategies will be the most efficient. Retrieval strategies are represented as XML tree of "strategy nodes", see figure 5. The output of this phase is the best result found. S_j=y_{ij} is the retrieval strategies at time t=i, y_{ij} is defined by :

$$\begin{cases} y_{00} = y_{0j} = y_{i0} = \{\} \\ y_{ij} = q_i D_{ij} \quad y_{i-1\ j}, \quad i = 1\ to\ m\ and\ j = 1\ to\ n \end{cases}$$

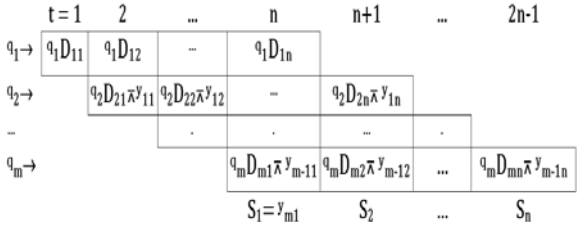


Fig. 2. *2PROM pipeline principle*

In figure 2, Q=$\{^{q}1, ^{q}2, \ldots, ^{q}m\}$ is the query, DS_j=$\{D_{j1}, D_{j2}, \ldots, D_{jn}\}$ is a data source and $^{q}2D_{21}y_{11}$ =IHM($^{q}2D_{21}$,$^{y}11$) is the retrieval strategy at time t=2, it's the processing of $^{q}2D_{21}$ knowing the retrieval strategy $^{q}1D_{11}$. In this pipeline, the output of one element is the input of the next one. Elements are executed in parallel. IHM (D_{ij}, q_i) is retrieved only when the probability of similarity between the query q_i and the domain D_{ij} is greater than a certain threshold θ.

A. Pipeline Algorithm

The pipeline input are i) the data souces domains D_{ij} and ii) the decomposed sub queries q_i. The output is the efficients retrieval strategies S_j=y_{ij},

Algorithm Pipeline()
Input
1: *Data* $DS_i \in DSP$ *//data sources*
2: Data $D_{ij} \in DS_i$*//domains*
3: Data $q_i \in Q$*// Sub queries*
4: *integer* var i, j, k, n, m, t;
5: $y_{0j} \leftarrow \{\}$; $y_{i0} \leftarrow \{\}$; $y_{00} \leftarrow \{\}$;
Output
6: *Data* $y_{ij} \leftarrow S_j\ \leftarrow\{\}$ *//retrieval strategies*
7: BEGIN
8: **Case** t **from** 1 **to** n-1: Compute InitialPhase();
9: **Case** t **from** n **to** 2n-1: Compute GetsStrategies();
10: END

Algorithm InitialPhase(): - Figure 3a -

1: Begin
2: t←1
3: **repeat**
4: i←1 ; j←t ;
5: **while** (i≤t) **and** (i≤m) **do**
6: $y_{ij} \leftarrow q_i D_{ij}\ y_{i-1j}$ *//recursive sorting of* y_{ij}
7: i←i+1
8: j←j-1
9: **end while**
10: t←t+1
11: **until** t=n-1
12: End

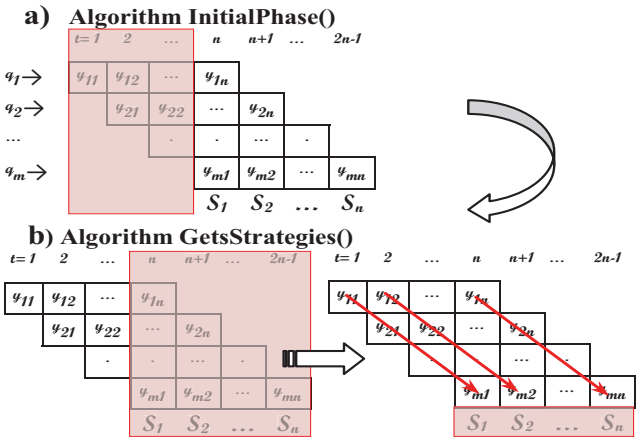


Fig. 3. *2PROM pipeline Algorithm*

Algorithm GetsStrategies(): - Figure 3b -

1: Begin
2: t←n; k←1;
3: **repeat**
4: i←k; j←n;
5: **while** (i≤m) **and** (j≥k) **do**
6: $y_{ij} \leftarrow q_i D_{ij}\ y_{i-1j}$ *//recursive sorting of* y_{ij}
7: i←i+1
8: j←j-1
9: **end while**
10: **if** (RSV(q_{i-1}, D_{i-1j+1})>0.4) **then** $S_i \leftarrow y_{i-1j+1}$
11: t←t+1
12: k←k+1
13: **until** t=2n-1
14: End

At t=1, algorithm InitialPhase() processes $y_{11}=q_1D_{11}$, q1 is processed in the first domain D_{11} of DS_1.
At t=2, algorithm InitialPhase() processes $y_{12}=q_1D_{12}$ and $y_{21} = q_2D_{21}\ y_{11}$ at the same time. q_1 is processed in the second domain D_{12} of DS_1, q_2 is processed in the first domain D_{21} of DS_2 knowing the result of q_1D_{11}.
And so on.
At t=n-1, algorithm InitialPhase() processes $y_{1\ n-1}$, $y_{2\ n-2}$, …, $y_{mn-1\ 1}$at the same time. q_1 is processed in the domain $D_{1\ n-1}$of DS_1, q_2 is processed in the n-1 domain $D_{2\ n-1}$of DS_2 knowing the result of $q_1D_{1\ n}$,..., q_m is processed in the first domain D_{m1} of DS_m knowing the result of $y_{m-1\ 1}$,we then get the first retrieval strategy y_{m1}=(q_mD_{m1} knowing $q_{m-1}D_{m-1\ 1}$ knowing $q_{m-2}D_{m-2\ 1}$ knowing ... knowing q_1D_{11}).
At t=n, GetsStrategies() processes $y_{1\ n}$, $y_{2\ n-1}$,…,$y_{m\ 1}$ at the same time; so n operations are processed at the same time. From t=n to 2n-1, GetsStrategies() gets one retrieval strategy at each clock of time.

B. Example

Let n=m=4, $\theta = 0.4$, according to figure 3 :

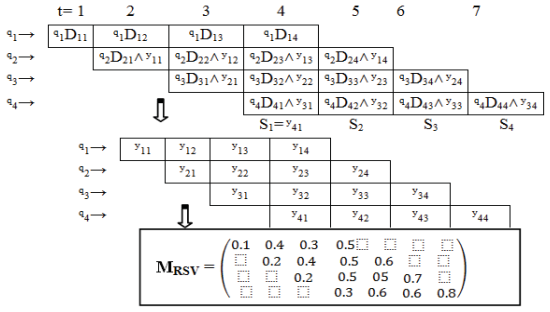


Fig. 4. *Matrix* M_{RSV}

$S_1=y_{41}$ then RSV(q, S_1)=RSV(q, y_{41}), GetsStrategies() processes y_{41} as the retrieval of q_4D_{41} knowing the strategy y_{31}, and y_{31} is in his turn the retrieval of q_3D_{31} knowing the strategy y_{21}, y_{21} is the retrieval of q_2D_{21} knowing the strategy y_{11}, y_{11} is the retrieval of q_1D_{11}, hence, RSV(q, S_1)= RSV(q, y_{41})= $q_4D_{41}q_3D_{31}q_2D_{21}q_1D_{11}$.

```
<?xml version="1.0" encoding="UTF-8"?>
<RetrievalStrat xmlns="http://www.fanzounar.ifrance.com/schemas/Strat"
xsi:schema="http://www.fanzounar.ifrance.com/schemas/Strat.xsd">
    <StrategyCode>S</StrategyCode>
    <Query id="Q">
        <SubStrat id="S2">
            <Source>DS4 $ DS3</Source>
            <QueryType code="Keyword $ SQL">
                <RSV>0.475</RSV>
                <IHM>q4D42^q3D32</IHM>
            </QueryType>
        </SubStrat>
        <SubStrat id="S3">
            <Source>DS4 $ DS3 $ DS2</Source>
            <QueryType code="Keyword $ SQL $ XQUERY">
                <RSV>0.475</RSV>
                <IHM>q4D43^q3D33^q2D23</IHM>
            </QueryType>
        </SubStrat>
        <SubStrat id="S4">
            <Source>DS4 $ DS3 $ DS2 $ DS1</Source>
            <QueryType code="Keyword $ SQL $ XQUERY $ SQL">
                <RSV>0.62</RSV>
                <IHM>q4D44^q3D34^q2D24^q1D14</IHM>
            </QueryType>
        </SubStrat>
    </Query>
</RetrievalStrat>
```

Fig. 5. *XML document representing a Retrieval Strategy*

According to the matrix M_{RSV} of figure 4, RSV(q, S_1) = (0.3+0.2+0.2+0.1)/4 = 0.2 <0.4. Using GetsStrategies(), 2PROM also gets RSV(q, S_2)=(0.6+0.5+0.4+0.4)/4 = 0.475>0.4, RSV(q, S_3) = (0.6+0.5+0.5+0.3)/4 = 0.475>0.4

and RSV(q, S_4)= (0.8+0.7+0.6+0.5)/4 = 0.62>0.4.
Then S_2 = IHM(y_{42}, q) = {y_{32}, y_{42}}, S_3 = IHM(y_{43}, q) = {y_{23}, y_{33}, y_{43}} and S_4 = IHM(y_{44}, q) = {y_{14}, y_{24}, y_{34}, y_{44}}.
Finally the set of strategies is :
S = {S_2, S_3, S_4}, see figure 5.

IV. Experimental results

We implement the proposed 2PROM algorithm using C++Builder 6.0 and PHP 5.0 and compared on a mixed network with 3 computers, each using a 3GHz cpu with 1Gb memory. We consider the time results to check the optimization technique over selected systems and, the precision of retrieval P and the recall R:

$$P = \left(\frac{Number\ of\ relevantimagesretrieved}{Totalnumberofimagesretrieved}\right)$$

$$R = \left(\frac{Numberofrelevantimagesretrieved}{Totalnumberofrelevantimages}\right)$$

- Precision of Retrieval(see figure 6 a): For 200 2PROM retrievals, about 197 are relevant to the user. For more than 350 retrievals, AXML returned many non relevant results, i.e. 73 are not selected by the user; about 31.61% of retrievals are not similar for OrientSpace, 2PO and TPOA.
- Response Time(See figure 6 b): Our five search systems are running almost at the same response time. For more than 8000 data, AXML is a little bit fast (0.101s) than 2PROM. For 10000 data 2PROM is faster(0.112s). 2PROM retrieves more relevant results in less time than others.

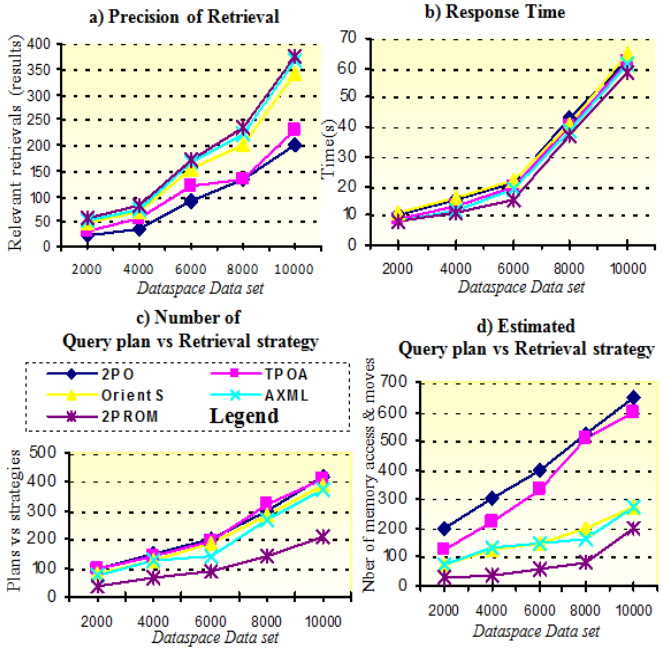


Fig. 6. *Plan vs Strategy and Response Time*

- Plans vs Strategies(See figure 6 c): For 6000 data, 2PO and TPOA are using 197 plans (i.e. 3,282% of 6000), AXML is using 145 plans(i.e 2,412% of 6000) and Strategies used by 2PROM is 89 (i.e. 1,481% of 6000). For 10000 data, plans used by 2PO, TPOA and AXML is almost 4,21% of dataset, while 2PROM uses only 2,782%. Hence, as the dataset data increase, the number of plans become very important.
- Plans vs Strategies cost(See figure 6 d): TPOA and 2PO need to access the memory almost twice the number of plans. Indeed, for 8000 dataset data, they used 270 plans, and access the memory 540 times. AXML and OrientSpace need to access the memory 1.2 times the number of plans. While, 2PROM needs to access only 1 time the number of strategies. The Response Time (chart b) is 1/10 time the number of plans or strategies (chart d).

V. Conclusion and Further Work

This paper proposed 2PROM, a two-Phase image Retrieval using IHM. IHM predicts which data sources hold the query result and collects feedback from satisfied users. If users are not satisfied or if the system did not find any result to the posed query, 2PROM stores the queries to optimize query processing. Users do not have to repeat their query if it is not satisfied. Indeed, saving queries could share common computation so that reduce the I/O cost significantly. Experiments show that 2PROM retrieves more relevant results in less time than existing systems, due to the pipeline which processes information in parallel. Our projects include working on massive scalability and large-scale storage solutions.

Acknowledgment

This work is supported by Jiangsu Provincial Natural Science Foundation of China(No.BK2011454), the Beijing Municipal Commission of Education (Grant No. KM201111417002) and by the Development Grant for Computer Application Technology Subject jointly Sponsored by Beijing Municipal Commission of Education and Beijing Jiaotong University(XK100040519).

References

[1] Essam A., Integration of Support Vector Machine and Bayesian Neural Network for Data Mining and Classification, *WAS'10*, pp202-207
[2] Abiteboul S., Ioana M. and Spyros Z., OptimAX: Optimizing Distributed ActiveXML Applications, *ICWE'08 on Web Engineering IEEE* Washington, DC, USA
[3] Chen S., Lin S., and Huang L. , A Two-Phase Optimization Algorithm For Mastermind, *the computer journal*, vol. 50 no. 4, 2007 pp. 435-443
[4] Jiratta P., and Raweewan A., Two-Phase Optimization for Selecting Materialized Views in a Data Warehouse , *World Academy of Science, Engineering and Technology 25* 2007
[5] Henriques R., Manuela C., Paulo S. and Antonio C., A Two-Phase Ant Colony Optimization Algorithm for solving the Vehicle Routing Problem with TimeWindows, *16th Conf. on Pattern Recognition (RecPad 2010)*.
[6] G. N. Fanzou Tchuissang, N. Wang, C. Kuicheu, F. Siewe, X. De; Predicting Dataspace Retrieval Using Probabilistic Hidden Information, *IEICE Trans on Information and Systems*, Vol. E93.D, Issue 7, pp. 1991-1994 (June 2010).
[7] G. N. Fanzou Tchuissang, N. Wang, N. C. Kuicheu, F. Siewe, X. Lin and De Xu. A Case Study of Image Retrieval on Lung Cancer Chest X-ray Pictures. *In the proceedings of 9th IEEE International Conference on Signal Processing (ICSP'08)*, Beijing, China, Oct. 2008, pp924-927.
[8] G. N. Fanzou Tchuissang, De Xu, Ning Wang, Francois Siewe; XIRS: an XML-based Image Retrieval System. *7th WSEAS International Conference on MULTIMEDIA, INTERNET VIDEO TECHNOLOGIES (ACM - MIV'07)*, Beijing, pp233-238, 2007.