A new approach for optimal knowledge extraction from heterogeneous web sources, using hypercubic clustering

Vassilis Nikolopoulos[1]

***Abstract*— This research paper presents a different mathematical and algorithmic approach for a special hypercubic clustering method in order to optimize multidimensional Information Retrieval methods based on Vector Space principle and to produce optimal ranked output data using simple search engine machines applied on heterogeneous ad-hoc data sources. The current algorithm will be also used for a distributed data mining method, using parallel hypercubic grids and special routing algorithms in order to produce relative ranked results from SQL queries and optimize OLAP procedure and structured data base estimation efficiency.**

***Index Terms*— Information Retrieval, Clustering, Hypercube, K-means algorithm, OLAP, search engines**

I. Introduction

IN order to understand better the notion and the functionality of a search engine, we have first to introduce some special definitions, concerning search engine efficiency, functionality and background theory.

A. Search Engine Theory

There are a variety of search engine techniques. As of June 2000, there were at least 3,500 different search engines on the Web.In fact, due to the proprietary nature of the field of information retrieval, it is difficult to say which techniques are most prevalent in industrial search engines such as Yahoo, Excite or Overture. Search engine creators loathe sharing their innovative ideas with other vendors as this could jeopardize millions of dollars in revenue. However, without a doubt, some search engines are more successful than others, an example being Google. This section outlines just a few of the most basic information retrieval techniques. Specifics of these techniques can easily become very complicated and are, in general, hard to come by since many vendors refuse to share in this competitive environment.

The Boolean model of information retrieval, one of the earliest and simplest retrieval methods, uses the notion of exact matching to match documents to a user query. A query is the information request of the user. A search engine answers a query by finding documents that are most relevant to the user's query. The Boolean model's more refined descendants are still used by most libraries. The adjective Boolean refers to the use of Boolean algebra, whereby words are logically combined with the Boolean operators AND, OR, and NOT. For example, the Boolean AND of two logical statements x and y means that both x AND y must be satisfied, while the Boolean OR of these two statements means that at least one of these statements must be satisfied. Any number of logical statements can be combined using the three Boolean operators. The Boolean information retrieval model operates by considering which keywords are present or absent in a document. Thus, a document is judged as relevant or irrelevant; there is no concept of a partial match between documents and queries. Other more advanced set theoretic techniques, such as the so-called fuzzy sets, try to remedy this black-white Boolean logic by introducing shades of gray. For example, a title search for car AND maintenance on a Boolean engine causes the virtual machine to return all documents that use both words in the title. As a result, a relevant document entitled "Automobile Maintenance" will not be returned.

B. Vector Space Model

Another information retrieval technique uses the *vector space model*, developed by Gerard Salton in the early 1960s, to sidestep some of the information retrieval problems mentioned above. Vector space models transform textual data into numeric vectors and matrices, then employ matrix analysis techniques to discern key features and connections in the document collection. Some advanced vector space models address the common text analysis problems of synonymy and polysemy. Advanced vector space models, such as LSI (Latent Semantic Indexing), can access the hidden semantic structure in a document collection. For example, an LSI engine processing the query car will return documents whose keywords are related semantically (in meaning), e.g. automobile. This ability to reveal hidden semantic meanings makes vector space models, such as LSI, very powerful information retrieval tools.

Two additional advantages of the vector space model are relevance scoring and relevance feedback. The vector space model allows documents to partially match a query by assigning each document a number between 0 and 1, which can be interpreted as the likelihood of relevance to the query. The group of retrieved documents can then be sorted by degree of relevancy, a luxury not possible with the Boolean model. Thus, vector space models return documents in an ordered list, sorted according to a relevance score. The first document returned is judged to be most relevant to the user's query. Some vector space search engines report the relevance score

[1]PhD Candidate at Multimedia Technology Laboratory, National Technical University of Athens, School of Electrical and Computer Engineering, Greece - email : vnikolop@medialab.ntua.gr

as a relevancy percentage. For example, a 97 per cent next to a document means that document is judged as 97 per cent relevant to the user's query. Relevance feedback is an information retrieval tuning technique that is a natural addition to the vector space model. Relevance feedback allows the user to select a subset of the retrieved documents that are useful. The query is then resubmitted with this additional relevance feedback information and a revised set of generally, more useful, retrieved documents is listed.

The drawbacks to vector space models are **their computational expense** and **poor scalability**. At query time, distance measures (also known as similarity measures) must be computed between each document and the query, and advanced models, such as LSI, require an expensive singular value decomposition of a large matrix that numerically represents the entire document collection. As the collection grows, the expense of this matrix decomposition becomes prohibitive.

C. New IR techniques

New information retrieval methods, in addition to those described above, are often used to search the Web. The Web's hyperlink structure, a structure not present in a standard information retrieval repository of documents, provides additional information that can be exploited in the retrieval process. Google was one of the first commercial search engines to recognize the importance of the Web's hyperlink structure. The underlying model for the successful Google engine is a sequential Markov chain.

In addition to the information retrieval challenges briefly mentioned above, such as polysemy, synonymy, query language or speed, other challenges exist in web information retrieval. Searching the Web presents its own unique challenges. The Web can be viewed as one huge database with the following unique properties: large amounts of volatile data (rapid updates, broken links, file disappearances), an exponentially growing amount of web pages, heterogeneous data (multiple formats, languages, alphabets), lack of structure, redundant data, and a lack of an editorial review publication process, which leads to numerous information errors, falsehoods and invalid statements. Furthermore, some advertisers try to increase traffic to their webpage by taking elaborate measures to fool automated search engines. For example, an advertiser might label sports, beer and swimsuits as keywords in its subject tag, thinking these topics are likely to arouse Web surfers' interests, while the true content of the page is classical clocks, a much less alluring topic. In fact, there are even companies whose sole purpose and means of profit is the manipulation of search engines. There are also search engines whose owners sell companies the right to be listed at the top of the retrieved documents list for particular queries.

An additional information retrieval challenge for any document collection, but especially pertinent to the Web, concerns precision. Although the amount of accessible information continues to grow, user ability to look at documents has not. Users rarely look beyond the first 10 or 20 documents retrieved. This user impatience means that search engine precision must increase just as rapidly as the number of documents

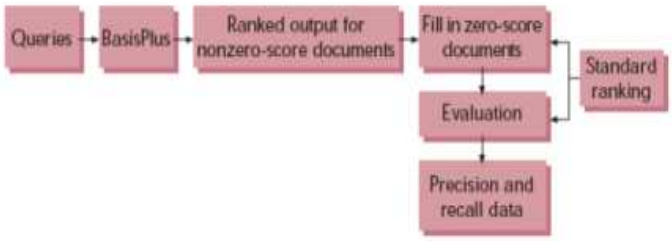

Fig. 1. General graph of search query

is increasing. Broken links also present an information retrieval problem. Often the most appealing retrieved document turns out to be a broken link, which quickly causes user frustration. Most search engines use a webcrawling technique to crawl the Web, gathering and storing information about individual webpages and documents, thus, in effect, removing broken links from their database of retrievable documents.

II. Optimization of Vector Space Method

As it was said in the previous section, Vector Space methods are extremely complicate, when we have additional data coming in and their scalability is rather poor. In this paper we will describe a new algorithmic method in order to optimize Vector space methodology by using multidimensional centroid clustering in a hypercubic network

A. Vector Space Method Analysis

The vector space model for document retrieval is based, upon building an n dimensional vector for the query and each document in the collection. The full non-optimized model causes *n* to be equal to the number of words in the language, whereas in reality *n* is usually equal to the number of different words in the document collection (words in the query are not important, because if they dont appear in the document then they are of no use for retrieval). The following is an example vector for a document in a language that only contains 10 words (i.e. there are only ten unique words in the collection):

$$D_1 = (1, 0, 0, 1, 0, 0, 0, 2, 2, 0)$$

The elements of the vectors are the frequency of occurrence of the word within the document; for example word 1 appears once in the document, whereas word 9 appears twice. The query vector is slightly different in that the presence, or not, of a word is indicated by a 1 or 0 respectively, counts of the words are not recorded. Finding relevant documents using this model, involves generating a vector for the query and each document in the collection and then ranking the documents by similarity to the query. This is usually expressed as the difference, θ, in direction of the query vector and a document vector, as shown in the figure below.

The query engine starts by reading the index from a file and uses this to re-create the data structure outlined above. Each query is then read in and analyzed to produce a data structure such as that shown below :

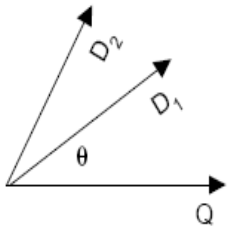

Fig. 2. Difference in direction of two document vectors with a query vector

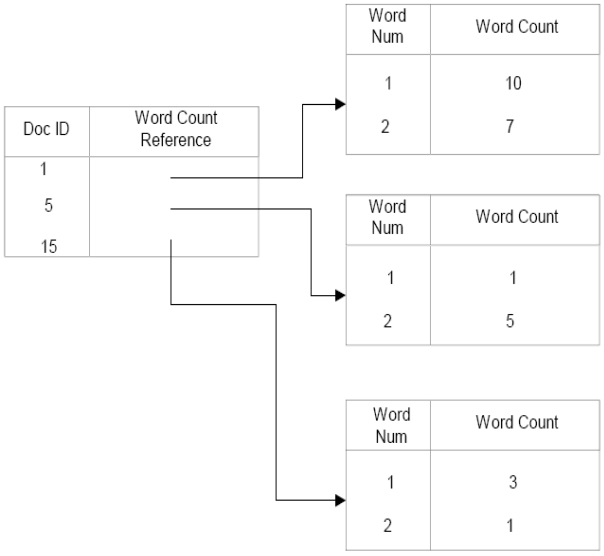

Fig. 3. Data structure for each query

This structure is document based, rather than word based, but is easy to construct from the query words and the index. It can be seen from the data structure used for the query and the algorithm for building this structure that the index is stored in a way which easily allows this algorithm to build up the data structure. Also the combination of indexing and querying structures contain all the data required for ranking the documents, using the TF or TF, IDF ranking algorithms.

B. Vector Space Method Optimization

The above algorithm ranks output data based on word count. Word count though poses many problems about the relevance degree of a specific document. Our proposed method **uses an optimized version of k-means clustering algorithm and a hypercubic grid in order to cluster pages in a distributed way**, not only by a word count analysis but using a relevance distance calculation method from an optimal value, which is called **centroid**. The probable pages that will be used to measure distances, form a surrounding grid in a multidimensional space. For each possible web page that is going to be assessed, we choose and construct an attribute vector describing some attributes that we have to take into account in order to decide if the page is relevant with a specific query. These attributes might *be keyword count, number of images (image tags) inside the page, time that a user stayed on that page after a relevant hit* etc. After the formation of the attribute vector, we assign weights (w_i) to the attributes to distinguish the most important and we choose possible optimal values (optimal pages) that are describes by various optimal attribute vectors. These optimal values are the centroids. After the formation of the centroids, we start to measure metric distances (*calculation of p-norm*) from each centroid and all the possible pages that we have gathered (*crawlers*) through internet (*identical to indexer*). This can be achieved by a hypercubic parallel network with moving agents, where the centroids are the vertices of a multidimensional hypercube. With this method, we create relevance tables for each centroid and the total system is called **Hypercubic Knowledge Grid (HNG)**.

The above algorithm can be seen on the figure below :

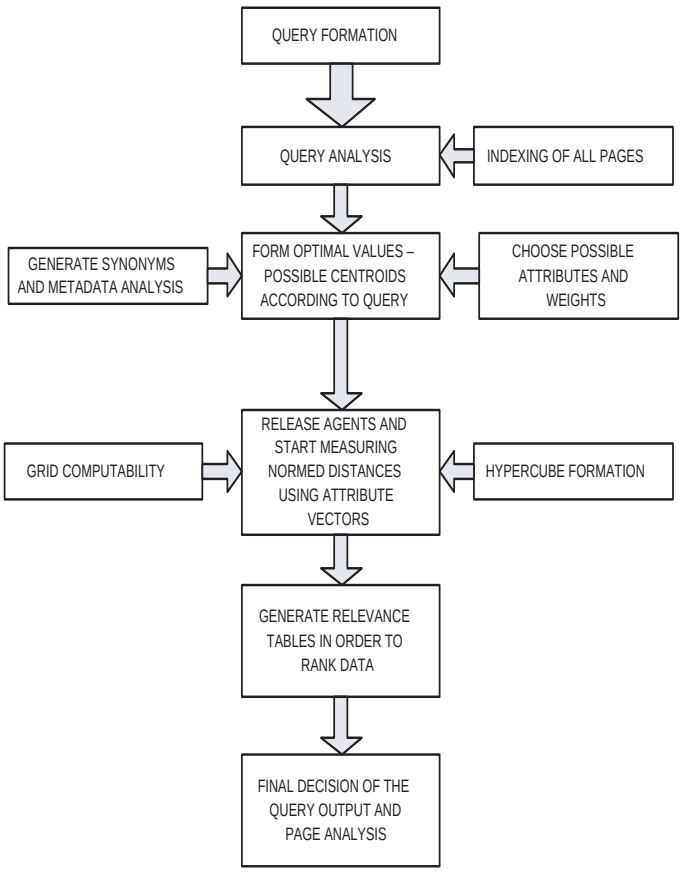

Fig. 4. Hypercubic Grid Algorithm

The above algorithm generates the knowledge grid and by using search algorithms we can access the relevance tables and rank output data according to a specific query. The above algorithm will be also used for Data bases analysis and SQL query optimization.

III. Introduction to the Hypercubic Topology

The Hypercubic topology will be used in this paper, in order to optimize and better explain the computational complexity of the clustering algorithm, suggested here.

A. Hypercubic Graph

We define a directed Graph *G* existing inside a finite metric Vertex subspace *V* consisting of the elements $\{x_1,x_2,...,x_n\}$ called *nodes* or *vertices* of the hypercubic network and one set *E* which is a subset of a Cartesian product $V \times V$ which is the set of *edges*.

The total number of the vertices defines the **degree** of the hypercubic graph, and we note that with *n*.

One directed hypercubic edge is defined by the arc *(x,y)*, which is formed by the two vertices *x* and *y* we note this topological relation by $\widehat{\{xy\}}$

Definition : Inside a hypercubic network, the total number of vertices (could be processors, threads, tasks) ***n*** represents the computational **degree n** of the parallel computability, which is also defined by the dimension ***n*** of the hypercube

Definition : The number of directed edges that leave a vertex is defined by the degree **N** of the hypercube. As we will see, the binary topology of the hypercube permits to use mathematical representation of power series (degree of 2) in order to explain and express the hypercubic dimension in connection with the number of vertices existing in a hypercube.

Definition : We define the dimension ***N*** of a hypercube equal to :

$$N = \log_2 n \Longleftrightarrow n = 2^N \tag{1}$$

The above mathematical representation will be used from now on, to describe the dimensional topology of a hypercube. (ie. a hypercube of a dimension five (*N=5*), has $n = 2^5 = 32$ vertices).

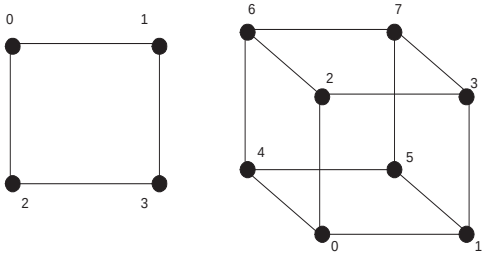

Fig. 5. Hypercubes of dimension N=2 and N=3

Definition : We call $\widehat{\{x_1x_2\}}$ neighbor vertices. We will see that for the hypercubic routing and data mining procedure, the vertices exist in a binary space $\mathbb{B}^N$ defining a binary dimension **N** of the hypercube and for binary vertices $\forall\{x_1,...,x_n\} \in \mathbb{B}^N$.

Definition : The distance $\mathbf{D}_{x_1 \to x_2}$ is defined as the number of minimum edges that the algorithm has to travel, in order to go from one vertex into a neighbor. This distance is defined as the **Hamming Distance**, between two nodes inside a hypercubic network.

One of the biggest advantages of the hypercube and the basic reason that was chosen for parallel data mining and Information retrieval, is the isomorphic nature of its topology, such that there is a bijective mapping $\varphi : V \to V'$ which verifies :

$$xy \in E \Leftrightarrow \varphi(x)\varphi(y) \in E', \forall\{x,y\} \in V \tag{2}$$

Isomorphic topology means that every vertex and every edge are symmetric and thus we can find for each vertex pair an automorphism operator φ such that : φ ($G_H(N)$) so that $(\varphi(x_i), \varphi(x_j)) = (x_r, x_s)$.

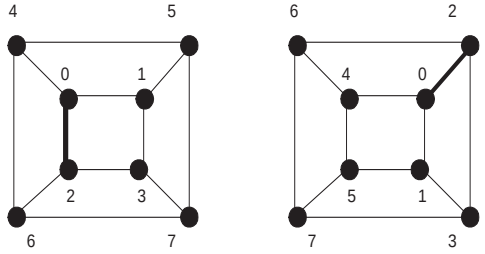

Fig. 6. Isomorphic symmetry between two vertex neighbors

The above mentioned symmetry, helps us a lot in order to **assign different optimal values to the various vertices of the hypercube and take these values as optimal centroid solutions for a multidimensional data mining application** that has as a result, ranked sets of information retrieval queries.

B. Hypercubic Agent routing for distributed Data Mining

In order to perform a distributed data mining and clustering algorithm inside a hypercubic network, we need to ensure that parallel communications between vertices follow some structured rules and that ranked output data will be available.

Definition : A hypercube with dimension **N**, (G_H(N)) is a network grid having 2^N binary vertices which are mapped on a binary space $\{1,0\}^N \in \mathbb{B}^N$ and we define two adjacent vertices (*neighbors*) if and only if they are different by one single binary bit.

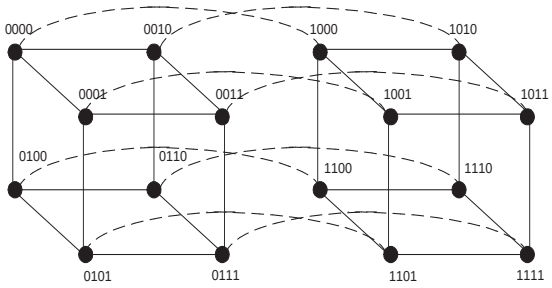

Fig. 7. Interconnected binary Hypercube of Dimension N=4

Definition : We define $H(x_1, x_2)$ as the **Hamming Distance** between two hypercubic vertices x_1, x_2, representing the total number of different binary bits for the vertices $\forall\{x_1,...,x_n\} \in \mathbb{B}^N$, where **N** is the hypercubic dimension.

$$H(x_1, x_2) = \sum_{i=0}^{N-1} h_i \in \mathbb{R}^1 \tag{3}$$

where $h_i = 1$ when the *i-th* binary bit of the edge x_1, x_2 is different, and 0 if not.

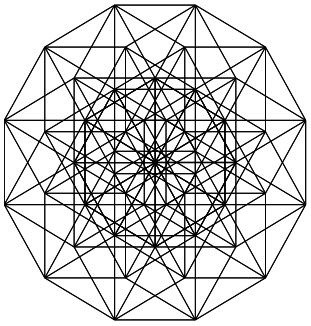

Fig. 8. A six-dimensional hypercubic grid with 64 vertices

It's clear from the above definition, that two topological neighbors $x_1 \rightarrow x_2$ have a Hamming distance equal to one $H(x_1, x_2) = 1$

The distributed agent hypercubic scheme that will be used in order compute the normed distance between an optimal centroid solution and a multidimensional vertex that represents a possible query to the search engine, will follow the above structure and the probabilistic routing algorithm of **Les Valiant**. The agent addressing scheme will be like :

We will define a probabilistic random vertex permutation Π, using a uniform distribution. In each vertex there exists an agent that will measure the n-th dimensional topological norm between the vertex and the optimal centroid, so the agent M_i will have a destination $\Pi(i) \equiv x_{iF}$ which is a random permutation.

Example : A 5-Dimensional hypercube $G_H(5)$, and the agent-routing algorithm will compute the distance norm at the beginning of the vertex $x_i = \{b_4^{x_i}, b_3^{x_i}, ..., b_1^{x_i}, b_0^{x_i}\}^5 = \{0,0,0,0,0\}^5$ until the destination vertex $x_{iF} = \{1,0,1,1,1\}^5$. The operation *XOR* will give us $x_i \bigoplus x_{iF} = \{1,0,1,1,1\}^5$. So the agent-routing algorithm will change the first, the third, the fourth and the fifth binary bit in *4 cycles* at least, because the Hamming distance $H(x_i, x_{iF}) = \sum_{i=0}^{4} h_i = \mathbf{4}$ and the relative agent path will be :

$$\{0,0,0,0,0\} \rightarrow \{1,0,0,0,0\} \rightarrow \{1,0,1,0,0\} \rightarrow \{1,0,1,1,0\} \rightarrow \{1,0,1,1,1\}$$

In each cycle the algorithm will compute the normed distance between each local vertex (optimal centroid) that the agent resides in each cycle and the surrounding grid of possible solutions from the query. A multidimensional relativity matrix will be created with all the normed distances between the centroid vertex and the surrounding grid. After the determination of the local matrix the agent moves to the next vertex, defined by the routing algorithm.

The computational complexity of the above hypercubic-based algorithm was computed by using the Chernov born and the mathematical analysis tells us that an agent, by using *O(nN)* random bits and that given θ, with a high probability of $1 - \frac{2}{n^\theta}$ will finish the above routing task in within $(2\theta + 4)N$ cycles, maximum. We will have to add the computational load for calculating the *p-th dimensional norm* $\|\cdot\|_p$ for each vertex, which of course depends on the dimension of the surrounding grid.

C. Centroid Relativity Measurement of surrounding grid

As it was mentioned above, relativity measurement between an optimal centroid and a probable value from the surrounding grid, can be achieved by measuring the metric distance between these two vertices. The distance $r(\cdot)^p$ can be mathematically represented by the *p-th topological* norm $\|\cdot\|_p$, inside the hypercubic space.

Lets consider the graph below, showing a centroid c_i and 6 possible values $x_1 ... x_6$ of dimension 4 (4 attributes describing each vertex), from the surrounding grid. In order to calculate the distance $r(c_i, x1), r(c_i, x2 ... r(c_i, x6))$ between the centroid and each value, we compute the mathematical expression shown below, in a recursive loop :

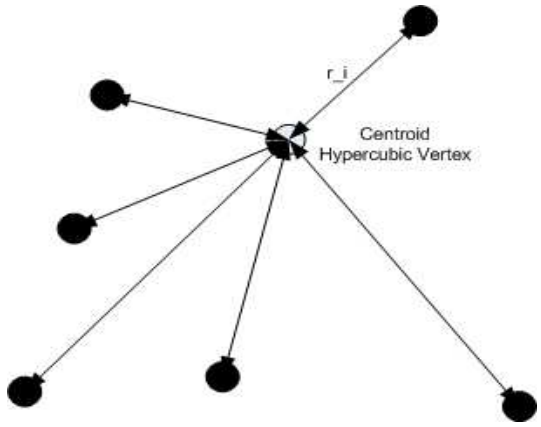

Fig. 9. Centroid vertex with surrounding grid

$$r_i(c_i, x_1) = \|c_i, x_1\|_p \quad (4)$$

$$r_i(c_i, x_2) = \|c_i, x_2\|_p \quad (5)$$

$$r_i(c_i, x_3) = \|c_i, x_3\|_p \quad (6)$$

$$r_i(c_i, x_4) = \|c_i, x_4\|_p \quad (7)$$

$$r_i(c_i, x_5) = \|c_i, x_5\|_p \quad (8)$$

$$r_i(c_i, x_6) = \|c_i, x_6\|_p \quad (9)$$

On the other hand, if we have one single value x_1 from the surrounding grid and we want to measure various distances from various centroids $c_{1..n}$, then we will have the expression *(3 centroids)* :

$$r_1(c_1, x_1) = \|c_1, x_1\|_p \quad (10)$$

$$r_2(c_2, x_1) = \|c_2, x_1\|_p \tag{11}$$

$$r_3(c_3, x_1) = \|c_3, x_1\|_p \tag{12}$$

So in order to take the p-th dimensional norm of the local vertex, we have to sum the 4-th dimensional difference attribute vector of each vertex, comparing with the centroid, such as :

$$r_i(c_i, x_1) = \|c_i, x_1\|_p \Rightarrow [\sum_{(j=1)}^{(n=4)} (c_i, (x_1)_{j1})^p]^{\frac{1}{p}} \tag{13}$$

with $p \equiv N$ where *N: Hypercubic dimension*

The above mathematical procedure is being used in order to calculate all the topological distances between centroids and relevant vertices, by using hypercubic routing and distributed agents. The results of the calculations form a uniform multidimensional table which is called **Relevance Table**. This table represents the degree of relevance of each vertex from the surrounding grid comparing with a unique optimal centroid (hypercubic node), **which can express an optimal solution, an optimal value or a suggested value to a problem or a query**.

IV. Hypercubic Clustering Simulations with routed Agents

In this final section, we have produced some very important simulations from a 6-th dimensional hypercube. Vertex agents perform a distributed data mining process from a surrounding grid, following simple collision-avoidance algorithms. The overall performance of the agent knowledge grid can be seen at the graphs below :

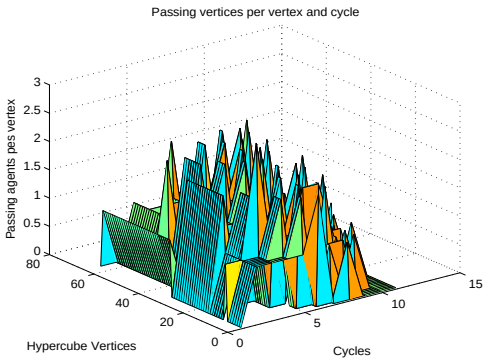

Fig. 10. 6th Dimensional hypercubic knowledge Grid Simulation

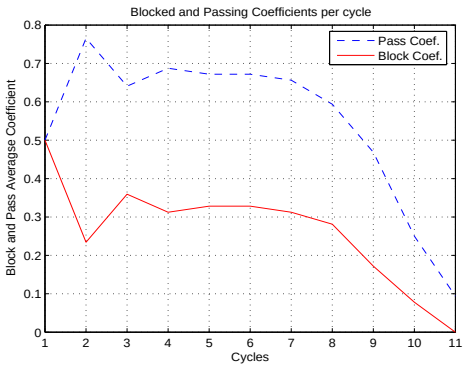

Fig. 11. 6th Dimensional hypercubic knowledge Grid Simulation

V. Conclusions

This paper has presented a different approach for a dynamic optimization of the Vector Space model in order to obtain ranked outputs from a specific query analysis. Hypercubic agents were used in order to perform a distributed data mining procedure and to generate an optimal knowledge grid from a huge number of web pages. The present paper is an introduction to hypercubic knowledge grid generation. Various simulations were produced, indicating the efficiency of the method. The algorithm can be executed in parallel, using hypercubic network.

References

[1] Vassilis Nikolopoulos, *Analyse et Simulation des methodes de routage dans la topologie d'hypercube*, Final Dissertation, Ecole Polytechnique, France 2002.
[2] Cori Robert, Hanrot Guillaume, Steyaert Jean-Marc, *Conception et Analyse des Algorithmes*, Majeure 2 d'Informatique, Ecole POLYTECHNIQUE X99, Paris.
[3] Ferreira Alfonso, *Issues in parallel computing with Hypercube multiprocessors*, CNRS - LIP, ENS Lyon, DIMACS Technical Report 94-49
[4] R. Baeza-Yates, B. Ribeiro-Neto., *Modern Information Retrieval*, Addison-Wesley, 1999.

Vassilis Nikolopoulos (1975) received a First Class Scottish Honours BEng Degree in Electrical and Electronic Engineering from the University of Dundee, Scoltand (2000) with a specialization in Control Theory and Robotics, an MSc and Diploma in Control Systems from Imperial College of Science, Technology and Medicine (2001), Certificates in Management and Marketing from London School of Economics (Summer 2001) and after preparing the classes préparatoires he obtained the Engineering Majors (Majeures d'Ingénieur) from the Ecole Polytechnique of Paris, in Applied Mathematics and Informatics (Promotion X99, 2002). He is currently a PhD Engineer candidate at the Multimedia Technology Laboratory of the National Technical University of Athens, Greece. He has gained the national IEE Prize 2000 for best academic performance (valedictorian), prizes in Mathematics and Physics and full scholarship from the Foundation of Ecole Polytechnique. His research areas cover adaptive knowledge management, multidimensional data mining and correlation techniques of distributed data bases, on-line web based OLAP analysis and advanced mathematical clustering algorithms. He is a member of Technical Chamber of Greece, IEE, BCS, InstMC, SEE, French Mathematical Society, IFAC and IEEE.