Data Mining Special Issue: Mining relevant information on the web: a clique-based approach

Massimiliano Caramia, Giovanni Felici

Data Mining Special Issue: Mining relevant information on the web: a clique-based approach

Journal:	*International Journal of Production Research*
Manuscript ID:	TPRS-2006-IJPR-0194
Manuscript Type:	Original Manuscript
Date Submitted by the Author:	08-Mar-2006
Complete List of Authors:	Caramia, Massimiliano; CNR, IAC Felici, Giovanni; CNR, IASI
Keywords:	WEB-BASED MANUFACTURE, DATA MINING, COMBINATORIAL OPTIMIZATION, DECISION SUPPORT SYSTEMS, E-MANUFACTURING, E-BUSINESS
Keywords (user):	

Note: The following files were submitted by the author for peer review, but cannot be converted to PDF. You must view these files (e.g. movies) online.

tprs-2006-ijpr-0194final.tex

Mining Relevant Information on the Web: A Clique-Based Approach

Massimiliano Caramia * Giovanni Felici †

Abstract

The role of information management and retrieval in production processes has been gaining importance in recent years. In this context, the ability to search and quickly find the little information needed in the huge amount of information available has crucial importance. One category of tools devoted to such a task is represented by search engines. The satisfaction of the basic needs of the web user led to the research of new tools that aim at helping more sophisticated users (communities, companies, interest groups) with more elaborated methods. An example is the use of clustering and classification algorithms or other specific data mining techniques. In such context, a proper use of a thematic search engine is a crucial tool to support and orient many activities. Several practical and theoretical problems arise in developing such tools, and we try to face some of them in this paper, extending some previous work on web mining [6]. Here we consider two related problems: how to select an appropriate set of keywords for a thematic engine taking into account the semantic and linguistic extensions of the search context, and how to select and rank a subset of relevant pages given a set of search keywords. Both problems are solved using the same framework, based on a graph representation of the available information and on the search of particular node subsets of such graph. Such subsets are effectively identified by a maximum-weight clique algorithm customized ad-hoc for the specific problems. The methods have been developed in the framework of a funded research project for the development of new web search tools, they have been tested on real data, and are currently being implemented in a prototypal thematic search engine. The web mining method presented in the paper can be applied to web-based design and manufacturing.

*Institute of Applied Mathematics "M. Picone" IAC-CNR, Viale del Policlinico, 137 - 00161 Rome, Italy. E-mail: m.caramia@iac.cnr.it

†Institute of System Analysis and Computer Science "A. Ruberti" IASI-CNR, Viale Manzoni, 30 - 00185 Rome, Italy. E-mail: felici@iasi.cnr.it

Keywords Web search, page vectorization, thematic search, maximum clique, k-densest subgraph.

1 Introduction

The importance of information management in production processes has been steadily growing with the evolution of information and communication technology and with the recent globalization process. The consolidated presence of information management has been joined with emerging importance by information retrieval, that is, by the technologies that enable their users to gain information about the production process, market competitors, and customers.

Today the web represents a key driving force for a large spectrum of applications in which users interact with or within companies, organizations, governmental agencies, and educational or collaborative environments. In particular, the area of collaborative enterprise networks, where different actors share through the web - directly or indirectly - information about their production processes, seems particularly interesting for the application of thematic search and for the solution of the related problems that arise in this context.

User preferences and expectations, together with usage, content, and structural patterns obtained from the web, form the basis for intelligent, personalized, and business-optimal services. Key Web business metrics enabled by proper data collection and processing are essential to run an effective business or service. Enabling technologies include data mining, scalable data warehousing and preprocessing, sequence discovery, real time processing, document classification, user modeling and evaluation models. Recipient technologies that demand user profiling and usage patterns include recommendation systems, Web analytics applications, content management systems, and fraud or intrusion detection systems.

Another application is web mining for e-commerce. Typical concerns in e-commerce include improved cross-sells, up-sells, personalized ads, targeted assortments, improved conversion rates, and measurements of the affectivity of actions. Other applications are concerned with recommendation and personalization systems, intelligent web services, contextual information access and retrieval, alert and information filtering systems, adaptive

hypertext systems, web mining applications for business and competitive intelligence, log analysis for security applications.

For all the tasks described above it is also very important to obtain a general consensus, among the specific group of users, on what makes some piece of information relevant, and on the main definitions and meaning of the words used within that context; a specific way to tackle these problems is found in the recent development of onthologies (methods to build relations among the meanings of context-oriented terms).

In this framework, several have considered the use of data mining and optimization techniques, often referred to as *web mining* (for a recent bibliography on this topic see, e.g., [14, 23]). Such techniques are needed to support sophisticated applications, as the standard search methods used in classical search engines are designed for a too general category of users.

In previous work [6] we proposed a method for improving standard search results in a thematic search engine, where the documents and the pages made available are restricted to a finite number of topics, and the users are considered to belong to a finite number of user profiles. The method uses clustering techniques to identify, in the set of pages resulting from a simple query, subsets that are heterogeneous with respect to a vectorization based on context or profile; small and potentially good subsets of pages are constructed extracting from each cluster the pages with higher scores. Operating on these subsets with a genetic algorithm, the subsets with a good overall score and a high internal dissimilarity are identified. This provides the user with a few non-duplicated pages that represent, from a different viewpoint, the structure of the initial set of pages.

In this paper we consider two data mining problems related to the classification and search procedures that take place inside a thematic search engine when the pages must be stored and retrieved to satisfy a user search request. The solution of these problems would enhance the capability of a search engine, for example to identify structural information in a set of pages, or to provide semantic extension to search keywords.

Although such problems may be found in many application environments, we believe that they can be applied successfully to the many web mining problems mentioned above that arise in production processes. The quality of the information retrieved on the web is indeed important when collaborative networks among enterprises that operate in the same logistic chain are set up, or when information on a specific market is retrieved from

the web.

The work presented in this paper refers to some experimental work that has been carried out in a research project (see also [6]) with the aim of developing a thematic engine with new search and information retrieval capabilities to provide more quality when focused on specific topic, category of users, or business target.

Web mining methods such as those presented in this paper find a novel and promising application in web-based design and manufacturing.

The paper is organized as follows. In Section 2 we describe the current architecture of the thematic search engine that is being developed and highlight its features, followed by the notation that will be used throughout the paper (Section 3). Such considerations bring to the surface some of the current limits that we try to overcome with the techniques presented. Here we identify two main problems: the identification of a proper *vectorization* strategy that can be used to represent the pages in a vector space where data mining techniques may be used for advanced search; and the identification of semantically connected subsets of pages. Solving the former problem is of crucial importance for the effectiveness of the search engine, as it is the first step of information compression and affects the quality of most of the features of the engine. A solution to the latter problem intends to overcome the typical drawbacks that a user encounters when he/she is submitted a set of pages selected according to some score function that is added on the single pages of the set rather than computed on the sets as a whole.

These two problems, and the models proposed to represent them, are the topic of Sections 4 and 5, respectively. In these two sections we show how the main computational step to solve the two problems is the identification of subgraphs of a given graph G with certain properties, namely some variants of two strongly related problems: the maximum-weight clique and the k-densest subgraph. Given the particular nature of these problems in our application we propose an original solution algorithm that is able to guarantee good performances in very little time and can find the optimal solution to the problem when the graph is of reasonable dimension (Section 6). Computational experience on real data derived from the prototype of the search engine is reported in Section 7, where we describe the behavior of the subset selection algorithm and indirectly measure the effects of the proposed method on the pages considered. Finally, some conclusions are drawn in Section 8.

2 Architecture of the Thematic Search Engine

The search engine considered in this paper is a prototype currently being developed by an Italian company in collaboration with IASI-CNR. Its scope is to meet the demand of specific communities of private or professional users (associations, interest groups, large companies) providing customized services. Such services are mainly based on data mining techniques.

The web pages are collected from the web using some ad-hoc spiders that start their search from a set of sensible keywords (context keywords) that are provided by the designer or extracted from a corpus of pages and/or documents.

A given customization of the engine will be composed by a set of *contexts*, that identify some particular topic or category of knowledge, and a set of *profiles*, that identify groups of users that deploy the search engine with homogeneous needs. Contexts and profiles are identified by a limited set of relevant keywords (in the order of few hundreds) that are dynamically evolved according to the behavior of the user and to the pages that are collected from the web. Whenever an advanced search function is executed within a context or a profile, the pages are vectorized according to a transformation of the frequencies of the keywords of that context/profile and data mining is applied; in particular,

- page clustering and genetic algorithms are used for the identification of heterogeneous subsets of pages matching a particular search,
- tree structures are used to enable the user to navigate a set of pages with additional perspectives,
- neural networks are trained on line to learn the preferences of the user.

The main aim of the advanced search functions is to identify meaningful subsets of pages that are relevant both for their matching with some user provided search keywords and for their intrinsic structure, i.e., the relations among the selected pages. One of the additional features that are being implemented in the engine is the construction of a similarity measure among words that is used to extend the search in a semantic sense.

As anticipated, we consider here two particular problems:

- how to select a compact subset of keywords from a larger set to perform the context/profile vectorization in an effective way that may exploit the similarities among

words;

- how to select a subset of pages that, beside being matched with some search keywords, form a strongly connected cluster with respect to a context or a profile.

In the following sections an approach to deal with these problems is described.

3 Notation

In this section we provide some notation. We are considering the data stored in the search engine after the page collection phase (spidering); such data may be synthesized by the pages, the words contained in these pages, and the frequency of each word in each page. We will thus have:

- P: set of pages, page $p \in P$;
- W: set of words, word $w \in W$;
- $M_{W,P}$: occurrence matrix of words in W in pages in P, where the cell M_{w_j,p_i} represents the number of times word $w_j \in W$ appears in page $p_i \in P$;
- $i_W(p_1, p_2)$: number of words appearing both in pages p_1 and p_2 (*word-intersection* of p_1, p_2).
- $i_P(w_1, w_2)$: number of pages containing both the words w_1 and w_2 (*page-intersection* of w_1, w_2).

A similarity function among words is also introduced. This function is intended to take into account one or more aspects of word similarity from three standpoints: grammatical, semantic, and linguistic. The first (grammatical) is dealt with using standard language-specific stemming algorithms; it measures whether two words derive from the same root. The second (semantic) takes into account similar meaning of words, and is based on the information contained in thesauri and lists of synonyms. Such lists are built ad-hoc for each specific context/profile of the search engine. The third similarity (linguistic) is verified when the meaning of the two words match in one of the supported languages. In this context we do not enter the details of how these measures are computed, but we assume that some overall similarity measure among words is defined according to the scope of the

specific search, where a user may request to measure similarity by only one of the three measures above or by all of them, disjunctively. Thus, we define:

- $s_G(w_1, w_2)$ grammatical similarity (stemming);
- $s_S(w_1, w_2)$ semantic similarity (synonym);
- $s_L(w_1, w_2)$ language similarity (translation);
- $s(w_1, w_2)$ overall similarity (a combination of the selected above).

According to the specific application, different ways of measuring the similarity among words may be adopted. In the simplest setting, we assume similarity to vary between 1 (maximum similarity) and 0 (no similarity), and state that we use $s_G(w_1, w_2) = 1$ only when w_1 and w_2 come from the same root, $s_S(w_1, w_2) = 1$ only when w_1 and w_2 are synonyms, and $s_L(w_1, w_2) = 1$ only when w_1 and w_2 have the same meaning in different languages.

We can now introduce the models for the two problems described above.

4 The Identification of the Vectorization Set

Data mining algorithms are generally applied on a matrix representation of the data, where the rows are associated with the observed *records* and the columns are associated with the *variables* measured on the records. In the case of web search, the records are typically the pages contained in the engine data base, while the variables that describe them are some measure of their characteristics, derived from word frequencies and page structure (e.g., html tags, meta tags, and other structural information). It is of crucial importance for the effectiveness of the algorithms and for the compactness of the stored data that the *dimension* of the space where the pages are represented (that is, the number of columns of the data matrix) is limited. In previous work [6] we have shown a good experimental performance of clustering and genetic algorithms on a limited $(50 - 100)$ number of "special" words that are used to represent the pages within a context/profile.

The problem that we formulate here is how to choose this limited subset to optimize the retained information. Assume thus that a given set of pages P can be associated with a subset of words W (e.g., all significant words that appear in pages of P at least

k times, or the words that are considered relevant within a context by experts). Our intention is to select a subset W^* of given size with the objective of maximizing the information maintained by W^*, where $|W^*| << |W|$. Similar problems are faced by most *feature selection* techniques, where greedy selection techniques are typically used when many variables are present.

Consider a subset of words $\bar{W}$ and a set of pages P, and the related occurrence matrix $M_{\bar{W},P}$, whose cell $M_{i,j}$ represents the number of times word w_j appears in the page p_i. A direct measure of the information contained in this matrix is the density of $M_{\bar{W},P}$, $d(M_{\bar{W},P})$, that is equal to the ratio between the number of non-zero cells of the matrix and the total number of cells.

If we fix a value h for the dimension of set W^*, the problem is to determine a subset W^* such that $|W^*| = h$ and $d(M_{W*,P})$ is as large as possible. Moreover, we would also like the words in set W^* to be sufficiently different among each other, in order to enhance their ability to cover different pages. Such objective may be represented as described below. Assume that set W^* is used to generate an *extended occurrence matrix* $M^+_{W*,P}$ with the following procedure. For each pair of words $w_i \in W^*$ and $w_j \in W$, define $\delta_{ij} = 1$ if the similarity between words w_i and w_j is below a given threshold, and $\delta_{ij} = 0$ otherwise. Now, define the generic cell associated to keyword w_i and page p_k as $M^+_{w_i,p_k} = \sum_{w_j \in W} \delta_{i,j} \times M_{w_j,p_k}$. Matrix $M^+_{W*,P}$ is thus determined by an expansion of the occurrence counts taking into account not only the occurrences of the words in the "special" set W^*, but also those of words that are *similar* to the words in W^*. Obviously it is of great interest to obtain an extended occurrence matrix of maximum density, such that the page vectorization, based on similarity rather than on exact matching, results in as much information as possible.

The maximization of the retained information in this setting cannot be formulated and solved in a straightforward way, due to the highly combinatorial nature of the problem, to the dimensions of the instances that arise in real applications, and to the particular type of objective function. We thus propose a model based on a graph representation of the problem, where the solution is identified by a subset of the nodes that maximizes a linear combination of the weights on the nodes and on the arcs. The model has two, possibly competing, objectives:

1. To maximize the number of pages covered, we want to choose words that appear in

many pages;

2. To avoid the selection of similar words and thus maximize the extended occurrence matrix, we want the sum of the similarity among the words in W^* to be minimal.

Consider now the following complete graph $G = (V_W, E)$, where a node v is present for each word $w \in W$ with weight $a_v = \sum_{p \in P} M_{w,p}$, and an edge is present between each pair of nodes (u, v) with weight a_{uv} proportional to the inverse of the similarity between the words associated with u and v. A good way to represent the two objectives described above would be to identify a subset of h nodes that maximizes the sum of the weights of the selected nodes (page coverage) and the sum of the weights of the edges among the selected nodes (words dissimilarity). More formally, we can state the problem as follows:

$$\max \alpha \cdot \sum_{v \in S} a_v + \beta \cdot \sum_{v \in S} \sum_{w \in S} a_{v,w}$$
$$S \subset V_W$$
$$|S| = h$$

where α and β are the overall weights associated with the nodes and edges components of the objective function, respectively. The problem as stated above is a particular type of subgraph selection problem, for which we adopt a solution approach tailored ad-hoc, described in Section 6 and then applied for experimental tests (Section 7.2).

We now turn to the second web mining problem discussed in this paper, the search of subsets of "semantically connected" pages.

5 The Search of Semantically Connected Pages

When a user submits a query to a search engine he/she provides one or more keywords and is returned a set of pages that contain these keywords; the pages are ranked according to some criteria, usually based on the number and type of occurrences of the keywords in the pages. The ranking function differs from search engine to search engine, and strongly characterizes the quality of the engine itself. One problem in ranking is that it is a function that evaluates each page independently, and does not take into account a set of pages as a whole. Such aspect has some relevance in the quality of the search, as the

user normally visualizes only the top pages of the ranking assuming that they are the most interesting from his/her perspective. In [6], this problem has been considered and a method to determine a small subset of pages compliant with the scope of the search, but as different as possible from each other, has been proposed.

Here we consider the same problem from a different perspective: we wish to select a subset of pages that are compliant with the search keywords, but that also are characterized by the fact that they share a large subset of words different from the search keywords. We believe this characteristic represents a sort of *semantic connection* of these pages that may be of use to spot some particular aspect of the information present in the pages. In addition, we propose to represent such characteristic of a set of pages by some particular subgraph property that can be sought for by a proper algorithm, similarly to what is done in Section 4.

Suppose a search on a set of pages P is required based on the search keywords $SW = (sw_1, sw_2, \ldots, sw_k)$, and that the pages are vectorized according to a set of relevant words W. Given a set of words $W^o \in W$, let P_{W^o} be the set of pages *induced* by W^o, that is, the pages that contain at least one of the words in W^o; more formally,

$$P_{W^o} = \{p \in P | \exists w \in W^o : M_{w,p} > 0\}$$

Consider now two words w_1 and w_2 in W. We want to express a measure of how these two words are connected with respect to the pages where they appear; naturally, such measure may be given by their *page-intersection* $i_P(w_1, w_2)$; if we restrict the pages used to compute the page-intersection to the pages that contain at least one search keyword, we get a measure of the connection strongly related with our objective: words highly connected are words that co-appear in pages that are relevant for the search.

The idea is thus to select pages that, beside being relevant for the search, are induced by a set of strongly connected words. Moreover, we are inclined to select in this set those words that have a high degree of similarity with the search keywords, to enhance the significance of the pages induced.

We can formulate this problem as a *maximum-weight clique* problem on the following graph $G(\lambda) = (V_W, E)$, where:

- a node v is present for each word $w \in W \setminus SW$, with weight $a_v = \sum_{sw \in SW} s(w, sw)$,

that measures the similarity of the node/word with the search keywords;

- an edge is present between nodes u and v representing words w_1 and w_2 if the page-intersection $i_{P_{SW}}(w_1, w_2)$ computed on the set of pages P_{SW} is positive and if the overall page-intersection $i_P(w_1, w_2)$ is greater than λ.

The node weight of a clique of $G(\lambda) = (V_W, E)$ measures how much the words in the clique are similar to the search keywords in SW; moreover, all pairs of words induce at least λ pages, and at least one page containing the search keywords. Higher values of the parameter λ make the graph more sparse and the clique of maximum weight more connected from the semantic standpoint. We are again faced with a hard optimization problem, as we will discuss in Section 6; moreover, in this case we have stronger time constraints on the computation of the solution as this problem may need to be solved while the user is doing the search.

We adopt the same algorithmic approach used for the identification of the vectorization set (presented in Section 4) - properly adapted - that provides good heuristic solutions when truncated before termination. Experiments are presented and commented in Section 7 for different values of $G(\lambda)$.

6 Maximum-Weight Clique and Subgraph Selection

Maximum-weight clique and subgraph selection problems have been studied under different angles in the graph theoretical and mathematical programming community, and have been used in many contexts to model real life problems. We briefly consider below this family of combinatorial problems providing the basic definitions and some insights in their complexity and solution algorithms.

Maximum-weight clique problem. A clique C in a graph G is a node induced subgraph where each couple of nodes is connected by an edge. The maximum-weight clique problem consists in finding a clique of largest weight in a graph. Such a problem has been shown to be NP-hard [12]. Moreover, for every $\delta > 0$, no $(1 + \delta)$-approximation to the maximum clique problem exists (unless P=NP) and there exists a $\gamma > 0$ such that no n^γ-approximation exists, where n is the number of vertices in the graph (unless P=NP). In particular, the current-best approximation ratio for maximum clique that is achievable

in polynomial time is $O(n/log^2 n)$ [5]. Recent years have seen much progress in understanding the inapproximability of the problem, culminating in the result of [19] that it cannot be approximated in ZPP to within $n^{1-\epsilon}$ for any fixed $\epsilon > 0$, unless NP$\subseteq$ ZPP. The reader is referred to [18] for a survey of the (in)approximability of maximum clique.

In the literature, many heuristic approaches for the maximum clique problem are based on sequential greedy heuristics [2, 4, 10, 17, 21]. The idea is to build maximal cliques, starting from an empty clique, and iterating through the repeated addition of vertices. To decide which vertex is added, one uses a greedy heuristic such as choosing the vertex that has the highest degree among candidate vertices. To avoid usual greedy traps, greedy heuristics can be improved by injecting a mild amount of randomization combined with multiple restarts. Also, weights used by the greedy heuristic may be adapted from restart to restart as proposed, e.g., in [17, 21]. To improve the quality of a constructed clique, local search can be used to explore its neighborhood, i.e., the set of cliques that can be obtained by removing and/or adding a given number of vertices: local search iteratively moves in the search space composed of all cliques, from a clique to one of its (best) neighbors. To avoid being trapped in local optima, where all neighbors are cliques of smaller sizes, local search may be combined with some advanced meta-heuristics. For example in [1, 20], Simulated Annealing is used to jump out of local optima by allowing moves towards smaller cliques with a probability proportional to a decreasing temperature parameter. In [11, 13, 15], Tabu Search is used to prevent local search from cycling through a small set of good but suboptimal cliques by keeping track, in a tabu list, of forbidden moves among cliques. In [4], Reactive Search enhances Tabu Search by reactively adapting the size of the tabu list with respect to the need of diversification. In [22], local search is combined with a Genetic Algorithm that allows escape from local maxima by applying crossover and mutation operators to a population of maximal cliques.

Subgraph selection problem. The subgraph selection problem is as follows: given a graph $G = (V, E)$ and weights $w_{ij} = w_{ji}$ on the edges $(i, j) \in E$, the lightest (resp. heaviest) k-subgraph problem is solved by determining a subset $S \subseteq V$ with dimension k such that the sum of the weights associated with the edges of the subgraph induced by S on G is minimum (resp. maximum).

When unitary weights are associated with the edges, the lightest k-subgraph problem

amounts to the selection of the subgraph of dimension k with the minimum number of edges, namely, the sparsest k-subgraph problem. By switching the sense of the objective functions of the two problems above, it is then easy to obtain the heaviest k-subgraph and densest k-subgraph problems. Equivalently, the heaviest k-subgraph and the densest k-subgraph can be obtained from the lightest k-subgraph and the sparsest k-subgraph, respectively, with a trivial transformation of the objective function coefficients.

Feige and Seltser [8, 9] propose a reduction of the densest k-subgraph to the searching of a clique of a certain size in a graph. Moreover, the same authors show with additional arguments that the densest k-subgraph problem is in the NP-complete class even for bipartite graphs with the node degree bounded by 3. Efficient approximation schemes for the densest k-subgraph problem are analyzed; the authors show that, under the condition that G contains a clique on k nodes, an approximation algorithm exists that, for each $0 < \varepsilon < 1$, determines a subgraph on k vertices with at least $(1-\varepsilon)\binom{k}{2}$ edges with computational complexity equal to $n^{O((1+\log \frac{n}{k})/\varepsilon)}$.

In [7], a combinatorial feature selection problem is formulated as a *max distinct points* problem that reduces to the problem of the densest k-subgraph. Evidence is provided that an α-approximated algorithm for the max distinct points problem results in a 2α-approximated algorithm for the densest k-subgraph problem, and that an α-approximated algorithm for the *min-l distinct dimension* problem gives an $\alpha(\alpha+1)$-approximated algorithm for the densest subgraph problem. On the other hand, the greedy algorithm with backward selection has been shown to provide an approximation ratio of $O(k/n)$ for any k in [3].

The same case has been considered in [16] with a randomized technique that produces an approximation ratio of 0.25. In general, all approximation ratios obtained for this class of problems are constant in k and linear in n.

Analogies between the maximum-weight clique and the subgraph selection problems. When the lightest (heaviest) k-subgraph problem is run on a complete graph, trivially the output subgraph is a clique itself, and thus the problem reduces to finding a clique of given cardinality k with the minimum (maximum) weight of its edges (nodes). Moreover, also if the graph is not a complete one, we can search for a complete subgraph with k nodes with the lightest or heaviest weight. This strong condition on the structure

of the searched subgraph can be useful to enhance the connectivity characteristics of the subset found.

6.1 The Solution Algorithm

What we propose in the following is an algorithm to find a k-densest subgraph or a clique in a graph. This algorithm is fast and concentrates on the weight of the cliques of given input size. Thus, differently from the above mentioned algorithms that are concentrated on finding the maximum clique independently of its size, our algorithm focuses on finding cliques of the same requested size selecting the one with maximum weight, an approach to the problem that is particularly suitable for the two subgraph selection problems described in Sections 4 and 5. This characteristic is, in general, not possessed by the maximum clique algorithms mentioned above. Moreover, even if the problem of finding a clique of size k is polynomial with complexity $O(n^k)$, we note that here we do not want a clique of given size but rather a clique of maximum weight among those of a given size.

The algorithm is constructive and works by choosing a node with highest degree (which initializes the set) from which the constructive process starts. The nodes in the neighborhood of the starting node are ordered according to non increasing degree, and are stored in a list L. Nodes in L are are visited one by one checking if the current clique can be expanded. In case of a positive answer we expand the current clique.

Once a node is stored in the list, we associate with the node a parameter *distance* that corresponds to the distance of such a node from the starting node; i.e., the distance is 1 if it belongs to the neighborhood of the starting node, the distance is 2 if it belongs to the neighborhood of a node in the neighborhood of the starting nodes, and so on. Moreover, we associate with each node another parameter called *position* that takes into account the position of the node in the ordering of the neighborhood to which it belongs.

Once we run out of nodes in the neighborhood, we consider the neighborhoods of these nodes just visited and these neighborhoods are stored at the end of list L ordered according to their degree as well, each one with its *distance* and *position* values.

Once the storage is ended, the process iterates, i.e., the nodes inserted in the list and already not visited are scanned starting from the head of L, checking whether the clique can be extended, and keeping on examining neighborhoods till all the graph is visited.

Parameters *distance* and *position* help in applying a sort of backtracking procedure

when we run out of nodes during the exploration of the node vector, or when the target cardinality clique is reached in the case of an instance where k is assigned (as it is for the problem described in Section 4). In fact, when this happens, we progressively delete all the nodes in the vector associated with the same distance, i.e., having the same parameter *distance*, as the last nodes put in the clique, and restart the search process by the node with the successive *position* within this distance. If there is no node left within that distance, then we delete all the nodes at $distance - 1$, i.e., with the immediate shorter distance, and start the process with a node with the successive value of *position*.

The described process is repeated for a certain number of iterations; when this limit is reached, the process restarts from scratch with a starting node of the successive highest degree. In the following we give a compact description of the algorithm for the run with the first node with the highest degree.

Scheme of the Algorithm

1. Choose a node with highest degree, and initialize the current clique with this node; initialize a list $L = \emptyset$;
2. Put neighborhood nodes in L according to non increasing order of degree; set $position = 1$ and $distance = 1$;
3. Select the first node in L already not visited and check if the current clique can be extended with this node; assign to this node parameters *position* and *distance*. In case of a positive answer, extend the current clique. If a target clique size is reached then goto 6;
4. If L is empty then goto 5, otherwise $position = position + 1$ and goto 3;
5. If all the nodes in the graph have been visited, goto 6; otherwise, select nodes with $distance = distance + 1$, and put them in L according to non increasing order of degree; set $position = 1$; goto 3;
6. Update the best clique if the one found is of better weight and apply backtracking: delete from L all the nodes with *distance* equal to that of the last node inserted in the current clique; delete all the nodes in L with such *distance*, and *position* less than or equal to *position* of the last node inserted in the current clique; goto 4.

Dataset Name	Number of Pages	Number of keywords	Number of nonzero elements in occurrence matrix
A	2,544	3,012	4,399
B	3,405	6,443	177,797
C	8,086	10,453	44,619

Table 1: Dimensions of the dataset used for the experiments

7 Experimental Results

The tests of the proposed methods should aim at two objectives. The first is to verify the effectiveness of our algorithms on the formulated problems, with particular attention to the fact that the dimension of the problems involved may be large and we are bound to use heuristics or truncated search; the second is to check the relevance of the proposed methods in improving the quality of a search engine. Unfortunately, the second objective is more difficult to achieve, as it involves complex user evaluations that must follow the development of a proper prototype. Here we consider mainly the first objective, and only partially the second one, that is presently still under test within the project that sponsored this research activity.

The data used for the experiments were extracted from a database of approximatively $100,000$ pages associated with different contexts collected and vectorized by the search engine since the beginning of 2004. We considered three sub-contexts of different sizes, from which we derive a set of words and the related occurrence matrices. Table 1 reports the sizes of the three datasets, referred to as A, B, and C.

We have tested both methods on these datasets, analyzing the behavior of the algorithms according to their main parameters (Sections 7.1 and 7.2). For Vectorization Set, we considered a dimension ranging from 20 to 120 for datasets A, B, and C. For Page Selection, we also need to consider the set of search keywords that are used to determine the weight of the nodes; here we focus on the sets A and B and analyze the behavior of the method with 3 and 5 search keywords. In these experiments the graph connectivity was controlled by the threshold on the minimum weights of the arcs to be included in the graph, that ranged from 1 to 7.

7.1 The Vectorization Set

Some of the experiments conducted on the selection of an optimal vectorization set are synthesized in Table 2. As mentioned above, we have considered different dimensions for the target set h (column 1 of Table 2, starting from 20 up to 120 with step 10). Such value is thus the dimension of the clique identified by the algorithm. The objective function to be maximized adopts weights α for its node component and β for its edge component. Obviously, the tuning of these parameters has an impact on the value of the solution, and we have adopted this reasonable strategy that appears to provide sufficiently stable results:

- $\alpha = \frac{average\ edge\ weight}{average\ node\ weight}$
- $\beta = 2 * \frac{\alpha}{h^2}$

The software required for the experiment was coded in standard C language and run on an Intel 4 2Ghz processor with 1GB RAM under Linux Debian 2.0, compiled with the standard *gcc* options. The maximum running time for each problem was set to 600 seconds, as in this case there is no need for on-line computation. The best subset determined within the time bound was considered as a satisfactory solution.

As anticipated in Section 4, we measure the quality of the target set with the increase in the density of the projected standard and extended occurrence matrices ($d(M_{W,P}), d(M^{+}_{W,P})$). Table 2 reports the variation in the density value of the matrices for the target set and the increase with respect to the whole set for the normal (columns 2 and 3 of Table 2) and the extended case (columns 4 and 5 of Table 2).

Despite of the limitation on the running time, the heuristic solutions provided seem to convey very interesting results. The density of the regular matrix is increased by a significant factor for the three datasets when projecting on a set of 50 words; high increments are present also for larger subsets, where the proportion of "good" words may be reduced, and also the quality of the heuristic solution may decrease due to the time bound. Similar behavior is registered when the increase with respect to the extended matrix is considered. We registered also some variability in the results of the subsets, that is likely to be attributed to the different sparsity degree of the occurrence matrices in the three datasets. It is important to note that most data analysis techniques would have

$\|W^*\|$	$d(M_{W^*,P})$	$Increase$	$d(M^+_{W*,P})$	$Increase^+$
Dataset A				
20	0.01677	29.42	41.28	13,578.73
40	0.00876	15.37	21.90	7,204.54
60	0.00594	10.42	15.92	5,237.47
80	0.00479	8.40	12.94	4,255.54
100	0.00395	6.93	10.75	3,536.99
120	0.00340	5.96	9.41	3,094.03
Dataset B				
20	0.01475	18.21	13.30	2890.71
40	0.00768	9.48	26.73	5811.24
60	0.00781	9.64	20.68	4496.37
80	0.00526	6.49	17.72	3852.17
100	0.00464	5.73	16.33	3550.12
120	0.00517	6.38	14.55	3162.51
Dataset C				
20	0.0031	4.44	9.23	3036.56
40	0.0035	5.10	11.57	3806.04
60	0.00224	2.93	9.70	3192.90
80	0.00303	4.27	10.66	3499.73
100	0.00249	3.33	8.7	2878.74
120	0.00234	3.08	8.05	2647.52

Table 2: Increase in density for normal and extended occurrence matrices for datasets A, B, and C

problems in managing such large data matrices using some sort of optimality criterion. Moreover, the selected columns also have the feature of being somehow "different" from each other and thus provide a better covering of the pages in the set. It is also interesting to note how the extended density $d(M^{+}_{W*,P})$ (4^{th} column of Table 2) are all significantly larger that one, meaning that the information maintained by the selected columns is very high when we consider also the frequency of the words that are similar to the selected ones.

7.2 Page Selection

To investigate the appropriateness of the solution method for the second problem considered (see Section 5) we have run some experiments, described in Tables 3 and 4. The experiments were run on the same platform of the previous case and a maximum running time of 10 seconds was given. In the majority of the cases the best solution was obtained by the proposed heuristic within 3 seconds. Such running times appear reasonable for real time application once the code has been properly optimized and a more up-to-date hardware is used - not to mention the strong parallelization that may be adopted for the clique and subset selection algorithms. The tables give the results of the method when applied to datasets A and B described above, using two different sets of search keywords with dimension 3 and 5, respectively. For each such set the tables report the value of λ (column 1), that is the minimum value of required page-connectivity of an edge in the graph (see Section 5); the dimension of the best clique determined by the algorithm (column 2); the number of different pages induced by the nodes/words in the clique (column 3); the average number of search keywords in the pages, and the average number of words that are similar to the search keywords (columns 4 and 5, respectively), where we declare a word to be similar to a keyword if the similarity measure between the two strings is above a given threshold.

From the results reported in Tables 3 and 4 we see how the increase in the connectivity threshold λ has in general the effect of reducing the number of selected nodes (words) of the clique, but this increase does not always translates to the number of pages that are induced by the selected words; in fact, as λ increases, the graph becomes sparse and the arcs connect nodes that share more pages. A clique will then be formed by nodes that induce a larger set of pages. The non monotonicity of the relations among the columns of

Dataset A			3 Keywords	
Value of λ	Number of nodes	Induced pages	Average number of search keywords	Average number of similar words
1	18	127	0.91	35.96
2	13	24	1.05	38.90
3	14	91	1.19	41.85
4	14	82	1.32	44.90
5	12	104	1.04	41.57
6	11	16	1.64	16.56
7	11	21	2.24	17.17
Dataset A			5 Keywords	
Value of λ	Number of nodes	Induced pages	Average number of search keywords	Average number of similar words
1	12	7	1.7143	10.14
2	6	45	0.6	6.51
3	5	45	0.6	6.51
4	5	45	0.6	6.51
5	4	45	0.6	6.51
6	4	45	0.6	6.51
7	4	13	0.9231	2.69

Table 3: Selection of pages with semantic connection for dataset A

Dataset B			3 Keywords	
Value of λ	Number of nodes	Induced pages	Average number of search keywords	Average number of similar words
1	18	27	1.75	47.71
2	13	35	0.24	29.36
3	13	32	0.63	29.36
4	7	29	0.07	29.36
5	4	42	1.35	29.36
6	4	38	1.35	29.36
7	4	17	2.68	40.23
Dataset B			5 Keywords	
Value of λ	Number of nodes	Induced pages	Average number of search keywords	Average number of similar words
1	31	97	2.3486	14.36
2	14	54	4.0556	24.85
3	14	91	2.5495	13.12
4	14	82	2.8293	14.56
5	12	58	2.6923	16.12
6	11	66	2.8125	16.56
7	11	70	1.8143	14.71

Table 4: Selection of pages with semantic connection for dataset B

the table may thus be attributed to the particular characteristics of the data considered. We also note that the last two columns of the tables, that measure the interestingness of the pages with respect to the search keywords, report large numbers and that such numbers increase with the dimension of the search keyword set, as is to be expected.

8 Conclusions

The work reported in this paper deals with two open problems: the first is related with the representation of the web pages in a search engine, and, more specifically, with a technique to determine small subsets of words that are convenient to use to project pages in a vector space and apply data mining techniques; the second deals with the identification of small sets of pages that have a strong semantic structure, that is, pages that contain many similar words and match one or more user requested keywords. Both methods exploit the potential of the use of a similarity function among words that takes into account, according to a very simple yet effective interpretation of semantics, stemming, synonyms, and translation.

The two problems have been modeled as particular subgraph selection problems, and a fast heuristic algorithm for their solution has been proposed and tested on real data extracted from a prototypal search engine.

The results provide some interesting insight in the potential of the new problems that can be formulated and efficiently solved to give more value to web search, a very common process in many production processes. Web mining is now frequently applied in web-based design and manufacturing and the use of a thematic approach to search appears particularly relevant in this context.

The methods described are now being engineered and integrated in the new version of the prototypal search engine to undergo real user impact tests. Future research in this direction will include the tuning of the parameters in the algorithms used to identify the subset of words, and a deeper analysis of the interactions between computational time and solution quality to possibly improve the quality of the method with *ad-hoc* heuristic strategies.

References

[1] Aarts, E.H.L., and Korst, J.H.M., *Simulated annealing and Boltzmann machines: a stochastic approach to combinatorial optimization and neural computing*, 1989 (John Wiley & Sons, Chichester, U.K.).

[2] Abello, J., Pardalos, P.M., and Resende, M.G.C., On maximum clique problems in very large graphs. In *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, volume 50, pp. 119-130, 2000 (American Mathematical Society).

[3] Asahiro, Y., Iwama, K., Tamaki, H., and Tokuyama, T., Greedily finding a dense subgraph, in *Proceedings of the* 5^{th} *Scandinavian Workshop on Algorithm Theory (SWAT), Lecture Notes in Computer Science 1097, Springer-Verlag*, 1996, pp. 136-148.

[4] Battiti, R., and Protasi, M., Reactive local search for the maximum clique problem. *Algorithmica*, 2001, **29(4)**, 610-637.

[5] Boppana, R.B., and Halld'orsson, M.M., Approximating maximum independent sets by excluding subgraphs. *BIT*, 1992, **32**, 180-196.

[6] Caramia, M., Felici, G., and Pezzoli, A., Improving Search Results with Data Mining in a Thematic Search Engine. *Computers and Operations Research*, 2004, **31**, 2387-2404.

[7] Charikar, M., Guruswami, V., Kumar, R., Rajagopalan, S., and Sahai, A., Combinatorial Feature Selection Problems, in *Proceedings of FOCS*, 2000, pp. 631-640.

[8] Feige, U., and Seltser, M., On the densest k-subgraph problem, Report No. CS97-16, Weizmann Institute of Science, Rehovot, Israel, 1997.

[9] Feige, U., Kortsarz, G., and Peleg, D., The Dense k-Subgraph Problem. *Algoritmica*, 2001, **29**, 410-421.

[10] Feo, T.A., Resende, M.G.C., and Smith, S.H., A greedy randomized adaptive search procedure for maximum independent set. *Operations Research*, 1994, **42**, 860-878.

[11] Friden, C., Hertz, A., and de Werra, D., Stabulus: a technique for finding stable sets in large graphs with tabu search. *Computing*, 1989, **42**, 35-44.

[12] Garey, M.R., and Johnson, D.S., *Computers and Intractability*, 1979 (Freeman).

[13] Gendreau, M., Soriano, P., and Salvail, L., Solving the maximum clique problem using a tabu search approach. *Annals of Operations Research*, 1993, **41(4)**, 385-403.

[14] Getoor, L., Senator, T.E., Domingos, P., and Faloutsos, C., *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003 (Washington DC, USA).

[15] Glover, F., and Laguna, M., Tabu search. In *Modern Heuristics Technics for Combinatorial Problems*, pp. 70-141, 1993 (Blackwell Scientific Publishing, Oxford, UK).

[16] Goemans, M.X., *Mathematical programming and approximation algorithms*, Lecture on Approximate Solution of Hard Combinatorial Problems, 1996 (Summer School, Udine).

[17] Grosso, A., Locatelli, M., and Della Croce, F., Combining swaps and node weights in an adaptive greedy approach for the maximum clique problem. *Journal of Heuristics*, 2004, **10(2)**, 135-152.

[18] Halld'orsson, M.M., Approximations of independent sets in graphs, in *Proceedings APPROX '98 Conference, Lecture Notes in Computer Science 1444, Springer-Verlag*, 1998, pp. 1-13.

[19] Hastad, J., Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 1999, **182**, 105-142.

[20] Homer, S. and Peinado, M., On the performance of polynomial-time clique approximation algorithms on very large graphs. In *Cliques, Coloring, 27 and Satisfiability: second DIMACS Implementation Challenge*, volume 26, pp. 103-124, 1996 (American Mathematical Society).

[21] Jagota, A., and Sanchis, L.A., Adaptive, restart, randomized greedy heuristics for maximum clique. *Journal of Heuristics*, 2001, **7(6)** pp. 565-585.

[22] Marchiori, E., Genetic, iterated and multistart local search for the maximum clique problem, in *Applications of Evolutionary Computing, Proceedings of EvoWorkshops:*

EvoCOP, EvoIASP, EvoSTim, Lecture Notes in Computer Science 2279, Springer-Verlag, 2002, pp. 112-121.

[23] Zaiane, O.R., Srivastava, J., Spiliopoulou, M., and Masand, B.M., *Proceedings of the International Workshop in Mining Web Data for Discovering Usage Patterns and Profiles*, 2002 (Edmonton, Canada).