[21] TRIANTAPHYLLOU, E., AND LU, J.: 'The knowledge acquisition problem in monotone Boolean systems', in A. KENT AND J.G. WILLIAMS (eds.): *Encycl. Computer Sci. and Techn.*, M. Dekker, 1998, pp. 89–106.

[22] TRIANTAPHYLLOU, E., AND SOYSTER, A.L.: 'An approach to guided learning of Boolean functions', *Math. Comput. Modelling* **23**, no. 3 (1995), 69–86.

[23] TRIANTAPHYLLOU, E., AND SOYSTER, A.L.: 'A relationship Between CNF and DNF systems derivable from examples', *ORSA J. Comput.* **7**, no. 3 (1995), 283–285.

[24] TRIANTAPHYLLOU, E., AND SOYSTER, A.L.: 'On the minimum number of logical clauses which can be inferred from examples', *Computers Oper. Res.* **23**, no. 8 (1996), 783–799.

[25] TRIANTAPHYLLOU, E., SOYSTER, A.L., AND KUMARA, S.: 'Generating logical expressions from positive and negative examples via a branch-and-bound approach', *Computers Oper. Res.* **21**, no. 2 (1994), 185–197.

[26] TRICK, M., AND JOHNSON, D.: *Second DIMACS challenge on cliques, coloring, and satisfiability*, DIMACS. Amer. Math. Soc., 1995.

[27] VAN DIJCK, J., VERBEEK, A., HENDRICKS, J., AND HOLLAND, R.: 'The current detectability of breast cancer in a mammographic screening program', *Cancer* **72**, no. 6 (1993), 1933–1938.

Evangelos Triantaphyllou
Dept. Industrial and Manufacturing Systems Engin.
3128 CEBA Building
Louisiana State Univ.
Baton Rouge, LA 70803-6409, USA
E-mail address: trianta@lsu.edu
Web address: www.imse.lsu.edu/vangelis

Jennifer Austin-Rodriguez
Dept. Industrial and Manufacturing Systems Engin.
3128 CEBA Building
Louisiana State Univ.
Baton Rouge, LA 70803-6409, USA

MSC2000: 90C09, 90C10

Key words and phrases: medical diagnosis, inductive inference problem, Boolean classification problem, learning algorithm, conjunctive normal form, CNF, disjunctive normal form, DNF, satisfiability problem, SAT, minimum number of clauses, one clause at a time approach, OCAT, GRASP approach, randomized heuristics, missing information, unclassifiable examples.

OPTIMIZATION IN CLASSIFYING TEXT DOCUMENTS by Triantaphyllou et al.

From the 1950s onwards, the search for computerized tools and mathematical models that can speed up the *classification of large collections of documents* has been the focus of many research efforts. These efforts have been centered in developing tools that can speed up the classification of documents according to some underlying context. A current example of this situation is the *Internet*. In this worldwide conglomerate of databases, one can easily see the speed at which documents on the topic, say, 'basketball' are retrieved from among the millions of documents produced daily on the Internet. Document classification is also of paramount importance in many information retrieval applications, such as news routing [7], classification/declassification of official documents [15] e-mail filtering [27], and context derivation of electronic meetings [3].

From the 1950s onwards, various fields of the human knowledge have produced several solutions for the document classification problem (see, for example, [23], [21], and [2]). Some examples of these fields are mathematical optimization, computational linguistics, expert systems, neural networks, and genetic algorithms. These methodologies have been severely limited to some degree by the huge amounts of information, both textual and graphical, generated by today's information driven society. On the other hand, this 'technological' limitation has been the boost for the development of more efficient and effective classification procedures [15].

The purpose of this article is to exhibit some contributions of discrete optimization during the process of *automatic document classification*. This paper illustrates these contributions by presenting three cases (application areas) in which optimization is used in the classification process. The first case deals with a generic procedure for the selection of a set of *indexing terms* (*keywords* or *content descriptors*). The second case deals with the selection of an optimal set of indexing terms to minimize the overlapping of keywords used in different documents. The last case deals with the classification of text documents from mutually exclusive classes. These three cases are only a tiny sample of a vast collection of related instances in the field of information retrieval systems; see [11], [1], [25] for additional literature.

This article is organized as follows. The next section presents an overview of the document classification process. The subsequent section

es the three application areas in which optimi-
on has contributed in the solution of the clas-
cation problem. Finally, a summary section is
en.

verview of Automatic Classification of ocuments. The automatic classification of text ocuments consists of grouping documents of simi- r context into meaningful groups in order to facil- ate their storage and retrieval [22]. *Text classifi- ation* can be viewed as a four-step process. In the irst step, a representative sample of documents rom various classes is presented to a computer- zed system, and a list of the co-occurring words with their frequencies is secured (see, for example, [22] and [4]).

In the second step the frequency of the words is analyzed, and only the most 'meaningful' words are extracted as indexing terms (keywords or context descriptors) [14]. The 'meaningful' words or keywords are the words with moderate co-occurring frequencies. H.P. Luhn [13], G. Salton [21], D. Cleveland and A.D. Cleveland [4], and Ch. Fox [6] suggest the elimination of the 'common' and 'rare' words (i.e., frequent and infrequent words, respectively) as indexing terms because they convey little lexical meaning. Some examples of common words are 'a', 'an', 'and', and 'the' [6]; 'rare' words are dependent on the document's subject [13].

In the third step the context of unclassified documents is determined by affixing them with the keywords that occur in their text. According to [4], 'the assignment of these keywords to a document is correct because authors usually repeat words that conform with the document's subject'. Finally, the documents which were indexed with similar keywords are grouped together [22].

The set of keywords attached to each document during the third step is often referred to as a *document surrogate* or just a *document* [4]. A surrogate is a convenient way to represent and to computationally process the context of real documents. For instance, the surrogate of seven words {'document classification', 'document indexing', 'optimization', 'vector space model', 'logical analysis approach', 'OCAT algorithm', and 'machine learning'} is a condensed and convenient way to represent the context of this article which contains thousands of words, symbols, and numbers.

Often, a surrogate is further simplified by defining it as a binary vector. (For nonbinary surrogates, see [22].) In this case, when a surrogate's element $w_{ij} = 1$ (or 0), it indicates the presence (or absence) of keyword T_i $(i = 1, \ldots, t)$ in document D_j $(j = 1, \ldots, N)$. For example, the surrogate $D_k = [011110]$ of six binary elements indicates the presence of keywords T_2, T_3, T_4, and T_5 and the absence of keywords T_1 and T_6 in D_k. Fig. 1 shows a popular way to summarize a collection of N documents (surrogates) which are defined on t [22]:

$$\begin{array}{c} \\ D_1 \\ \cdot \\ \cdot \\ \cdot \\ D_N \end{array} \begin{array}{c} \begin{array}{ccc} T_1 & \cdots & T_t \end{array} \\ \begin{bmatrix} w_{11} & \cdots & w_{1t} \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ w_{N1} & \cdots & w_{Nt} \end{bmatrix} \end{array}$$

Fig. 1: A collection of N surrogates.

In the last step, documents sharing similar keywords are grouped together. This classification follows from the pairwise comparison of the surrogates in Fig. 1 [22]. More on this is described in the following section.

Examples of Optimization in Document Classification. Optimization techniques have been used in various areas of text classification with various levels of success. Their utilization has been limited mainly because the size of the document classification problem is so large that even with the current computerized technologies, it would take them very long time to produce an optimal solution. Despite these technological limitations, the contributions of optimization in this field can be seen, for example, in the selection of keywords, automatic classification of documents, automatic retrieval, etc. Some applications of these techniques are presented next.

The first example illustrates the principle of least effort [29]. This principle is used for the derivation of an indexing vocabulary based solely on the frequency of the co-occurring words in a

collection of documents. The second example illustrates the application of the vector space model [23] for the derivation of an optimal indexing vocabulary to minimize the overlapping of keywords used by different documents. The third example illustrates the utilization of a machine learning and operations research algorithm called the one clause at a time (OCAT) algorithm [28] for the classification of documents which belong to mutually exclusive classes.

Optimization in the Principle of Least Effort. The *principle of least effort* (PLE) can be viewed as one of the first optimization attempts in the area of document classification. It was introduced by H.P. Zipf [29]. Although the PLE does not have a strict mathematical formulation, the problem it solves can be stated as follows. Given a collection of documents, the question is how to derive the 'best' set of indexing terms that will be used to identify the subject of documents in the collection. The set of the best indexing terms (or keywords) is often referred to as *indexing vocabulary* (see, for example, [4]). Hence, the goal of the PLE is to derive an optimal indexing vocabulary with the most *meaningful words* occurring in these documents.

Under the PLE an indexing vocabulary is derived as follows. At first all the co-occurring words and their frequencies are extracted from the collection of documents. Then, these words are ranked in descending order according to their frequencies. Finally, the words with frequencies in between some preestablished upper and lower frequency limits are selected as the indexing vocabulary. The frequency boundaries of the meaningful words are determined by a trial-and-error approach [13]. Other words with co-occurring frequencies above or below the preestablished limits are known as 'common' and 'rare' words (the frequent and infrequent words, respectively) and usually are discarded for indexing purposes because they convey little lexical meaning (see, for example, [13] and [6]).

It is interesting to notice here that although the PLE does minimize the number of keywords, its unwise utilization may jeopardize the quality of the indexing vocabulary. This can be illustrated by considering the word 'a'. The word 'a' is one of the most common words in the English language (other such words are 'an', 'and', and 'the'; see, for example, [6]). Thus, if the collection of documents is about nutrition, then 'a' may represent the name of the vitamin 'a' or 'A', and its elimination clearly would jeopardize the quality of the indexing vocabulary.

Optimization in the Vector Space Model. One of the most successful models in information retrieval systems is the *vector space model* (VSM). It was introduced in the mid 1970s in [23]. The VSM solves problems of the following nature: Given are samples of documents. Then, the question here is how to derive an optimal indexing vocabulary such that keywords used in one document are minimally used in other documents. In [23], the VSM was also extended to determine a vocabulary that minimizes the overlapping of keywords used in different classes. That is, keywords used in documents belonging to one class are minimally used in other classes.

The VSM derives this *optimal vocabulary* as follows. At first, a sample of t words is taken from all the words co-occurring in a collection of N documents. This sample of words is used as a candidate indexing vocabulary. Then, all documents in the collection are indexed (their subject is defined) by using words from this candidate vocabulary. Document surrogates are formed in this step. Next, the VSM computes the *similarity of all the surrogates* in the collection according to

$$F = \sum_{i=1}^{N} \text{sim}(D_i, D_j) \qquad (1)$$

$$\text{for} \quad j = 2, \ldots, N \quad \text{and} \quad i \neq j.$$

Where $\text{sim}(D_i, D_j)$ measures the similarity between documents D_i and D_j. Usually, $\text{sim}(D_i, D_j)$ is replaced by a function that relates any two vectors, such as the functions illustrated in Table. This procedure is repeated by using various candidate vocabularies. Finally, the candidate vocabulary that minimizes the expression in (1) is selected as the optimal indexing vocabulary [23]. The following example illustrates an application of the VSM with *binary surrogates*. The cosine coefficient is used to solve (1).

similarity measure $\text{sim}(X,Y)$	Evaluation for binary term vectors	evaluation for weighted term vectors
inner product (IP)	$\lvert X \cap Y \rvert$	$\sum_{i=1}^{t} x_i \cdot y_i$
dice coefficient (DC)	$2\frac{\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert}$	$2\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2}$
cosine coefficient (CC)	$\frac{\lvert X \cap Y \rvert}{\lvert X \rvert^{1/2} \cdot \lvert Y \rvert^{1/2}}$	$\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{t} x_i^2 \cdot \sum_{i=1}^{t} y_i^2}}$
Jaccard coefficient (JC)	$\frac{\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert - \lvert X \cap Y \rvert}$	$\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sum_{i=1}^{t} x_i^2 + \sum_{i=1}^{t} y_i^2 - \sum_{i=1}^{t} x_i \cdot y_i}$

$X = (x_1, \ldots, x_t)$.
$|X|$ = number of terms in $|X|$.
$|X \cap Y|$ = number of terms appearing jointly in X and Y.

Table 1: Measures of vector similarity (taken from [22, Chap. 10].

EXAMPLE 1 Let the words $T_1, \ldots, T_7$ be the set of all the words which were found in documents D_1 and D_2. (In real practice, this set may contain hundreds or even thousands of words.) Next, let D_1 and D_2 be indexed with only four of these words. Hence, the question here is: what is the 'optimal' indexing vocabulary of four words that make document D_1 to be indexed with keywords that are minimally used by D_2?

It is not difficult to realize that the number of candidate vocabularies of four words that can be formed out of seven words is equal to $\binom{7}{4} = 35$. Table 2 shows the similarities between D_1 and D_2 for only three vocabularies. The first column of this table shows these vocabularies. For example, words T_1, T_2, T_4, and T_7 correspond to the first vocabulary. The second and third column show the binary surrogates of documents D_1 and D_2. For instance, the surrogate D_1 = [1 0 1 1] indicates the absence of word T_2 and the presence of words T_1, T_4, and T_7 in D_1. Similarly, the surrogate for D_2 = [0 0 1 1] indicates the absence of T_1 and T_2 and the presence of words T_4 and T_7 in D_2. Finally, the fourth column shows the CC similarity values, or $\text{sim}(D_1, D_2)$), between D_1 and D_2 for the three vocabularies.

	Surrogates		
Vocabularies	D_1	D_2	$\text{sim}(D_1, D_2)$
T_1, T_2, T_4, T_7	[1 0 1 1]	[0 0 1 1]	0.50*
T_2, T_4, T_5, T_6	[0 1 0 0]	[0 1 1 0]	0.25
T_3, T_4, T_5, T_6	[0 1 0 0]	[0 0 1 1]	0.00

*: $\text{sim}(D_1, D_2) = 2/(4^{1/2} \times 4^{1/2}) = 0.50$.

Table 2: Similarity $\text{sim}(D_1, D_2)$ for three candidate vocabularies of four words.

The CC similarity values in Table 2 indicate that when words T_3, T_4, T_5, T_6 are used as indexing terms, the similarity of documents D_1 and D_2 is minimal. Furthermore, the similarity value $\text{sim}(D_1, D_2) = 0.00$ indicates that both documents are completely different because their surrogates do not contain common words. Therefore, according to the VSM the optimal indexing vocabulary corresponds to terms T_3, T_4, T_5, and T_6. Thus, the other words T_1, T_2, and T_7 can be discarded as indexing terms.

The solution presented in this table seems to be a trivial one. However, it can be easily shown that for realistic indexing problems such a solution can be quite an elaborate one. Suppose, for example, that the number of words extracted from D_1 and D_2 is not seven, but fifty (which still is a small number for realistic situations). Furthermore, if this time one is interested in finding candidate vocabularies of ten words, rather than combinations of four words, then the number of vocabularies that can be constructed is $\binom{50}{10} = 10.27$ billions. That is, in addition to finding the minimization of (2), the VSM must also use an efficient search strategy to quickly eliminate many non optimal vocabularies. By the same token, it can be easily shown that if vocabularies of all sizes are considered, then the number of such vocabularies is determined by $2^t - 1$ (where t is the number of extracted words), which even for moderate values of t this expression is a too large number.

As a result of these humongous searching spaces, researchers have been compelled to design fast heuristic search strategies at the expense of optimality. Furthermore, because the size of the indexing problem can be very large, suboptimal heuristic solutions have been preferred over optimal but slow ones [2]. □

Optimization in the Classification of Text Docu-

ments. The process of document classification consists of grouping documents according to their underlying subject. Some examples of familiar document subjects are history, geography, music, engineering, etc. Usually, this underlying subject is determined by the set of indexing terms which was attached to each document. That is, documents about History share indexing terms whose content describes past events. Similarly, the indexing terms of documents about geography share content which describes different geographic places on earth. Hence, the problem that this process of document classification attempts to solve is described as follows: Given samples of preclassified documents (i.e., their surrogates), the question is how to use the information contained in their surrogates such that new unclassified documents can be grouped into the appropriate classes.

Input:	E^+ and E^-.
Output:	Logical rules in CNF (or DNF) form.
	$i = 1$; $C = \emptyset$;
DO	WHILE ($E^- \neq \emptyset$)
1:	$i \leftarrow i + 1$; /* i indicates the ith iteration/
2:	find a clause c_i which accepts all members of E^+ while it rejects as many members of E^- as possible;
3:	let $E^-(c_i)$ be the set of members of E^- which are rejected by c_i;
4:	let $C \leftarrow C + c_i$;
5:	let $E^- \leftarrow E^- - E^-(c_i)$;
	END;

Fig. 2: The one clause at a time (OCAT) algorithm.

There are many methodologies for solving this document classification problem. Some examples of such methodologies are: the vector space model for document classification [22]; fuzzy set theories [12] and [17]; semantic analysis methodologies [10] and [19]; and some others which use artificial intelligence approaches, [3], and [2]. To some extent all these methodologies use optimization in order to maximize (or minimize) some performance measure, which usually is the similarity between indexing terms. In what follows, we present only one methodology which is based on artificial intelligence and operations research approaches. This methodology is the one clause at a time (OCAT) algorithm [28] for the classification of examples (e.g., documents) in mutually exclusive classes. The OCAT algorithm uses optimization methodologies for constructing classification clauses (e.g., *word patterns*) of minimal (or near minimal) size. Fig. 2 shows this algorithm.

The OCAT algorithm is also a machine learning algorithm. It uses logical analysis and branch and bound approaches to extract knowledge (sets of rules) from sets of preclassified examples. It takes as input data samples of examples from (usually two) mutually exclusive classes and extracts knowledge that is represented in a compact form of key data patterns which can be used to classify new unclassified examples into these two classes.

The two mutually exclusive classes are referred to as the sets of positive and negative examples (denoted by E^+ and E^-, respectively). Furthermore, the collections of examples in both classes are defined over the same set of parameters (also called atoms, characteristics, or factors) which are assumed binary valued. Fig. 3 illustrates a set of four positive examples: e_1, e_2, e_3, e_4 and a set of six negative examples: e_5, e_6, e_7, e_8, e_9, e_{10}. All ten examples are defined on the four atoms A_1, A_2, A_3, and A_4. For instance, example $e_1 = [0\,1\,0\,0]$ indicates the presence of atom A_2 and the absence of atoms A_1, A_3, and A_4 in e_1. On the other hand, example $e_5 = [1\,0\,1\,0]$ indicates that atoms A_1 and A_3 are present and that atoms A_2 and A_4 are absent.

$$E^+ = \begin{matrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{matrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

and

$$E^- = \begin{matrix} e_5 \\ e_6 \\ e_7 \\ e_8 \\ e_9 \\ e_{10} \end{matrix} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

Fig. 3: Two illustrative sets of positive and negative examples.

When the OCAT algorithm is used to solve the document classification problem, E^+ and E^- (i.e., the sets with the positive and negative examples respectively) correspond to the sets of documents

which belong to two mutually exclusive classes. That is, documents in the positive class are the ones that belong in only *one* of the two classes, while the documents in the other class are the negative examples. Hence, Fig. 3 may represent a set of ten documents (surrogates) which were indexed by using four keywords.

The OCAT algorithm is a greedy algorithm which determines a set of compact clauses either in the *conjunctive normal form* or *disjunctive normal form* (CNF or DNF, respectively, as defined below). For example, CNF clauses are determined as follows. In the first iteration it determines a clause that accepts all the examples in E^+ while it rejects as many examples in E^- as possible. In the second iteration it performs the same operation using the original E^+ set but this time the current E^- set contains only the negative examples that have not been rejected by any of the previous clauses. The iterations continue until the set of constructed clauses reject all the negative examples. Hence, when these CNF or DNF clauses are taken together, they accept all the positive examples while they reject all the negative examples.

The conjunctive normal form and disjunctive normal form (see, for example, [24]) are defined as in expressions (2) and (3), respectively.

$$\bigwedge_{j=1}^{k} \left(\bigvee_{i \in \rho_j} a_i \right), \tag{2}$$

$$\bigvee_{j=1}^{k} \left(\bigwedge_{i \in \rho_j} a_i \right). \tag{3}$$

Where a_i can be either A_i or $\overline{A}_i$. Thus, a CNF expression (also called a *logical clause*) over a vector $v \in \{0,1\}^t$ is a conjunction of disjunctions defined on the terms A_i $(i = 1, \ldots, t)$. Similarly, a DNF expression is a disjunction of conjunctions on the same terms A_i.

Let n be the number of atoms and M_1 the number of positive examples. It can be easily shown that the maximum number of clauses that can be formed using n atoms and M_1 examples is equal to M_1 [28]. To form these clauses consider the first example $e_1 = [0\ 1\ 0\ 0]$. It can be observed that in order to accept this positive example at least one of the four atoms A_1, A_2, A_3, A_4 must be specified as follows: (A_1 = FALSE; i.e., $\overline{A}_1$ = TRUE), (A_2 = TRUE), (A_3 = FALSE; i.e., $\overline{A}_3$ = TRUE), and (A_4 = FALSE; i.e., $\overline{A}_4$ = FALSE). Hence, any valid CNF clause must include $\overline{A}_1$, or A_2, or $\overline{A}_3$, or $\overline{A}_4$. Similarly, the second positive example $e_1 = [1\ 1\ 0\ 0]$ indicates that any valid CNF clause must include $A\overline{A}_1$, or $A\overline{A}_2$, or $\overline{A}_3$, or $\overline{A}_4$. In this manner, all valid CNF clauses must include at least one atom as specified from each of the following sets: $\{\overline{A}_1, A\overline{A}_2, \overline{A}_3, \overline{A}_4\}$, $\{A\overline{A}_1, A\overline{A}_2, \overline{A}_3, \overline{A}_4\}$, $\{\overline{A}_1, \overline{A}_2, A_3, A\overline{A}_4\}$, and $\{A_1, \overline{A}_2, \overline{A}_3, A_4\}$. Relation (4) shows a CNF system which was derived by using the OCAT algorithm on the examples in Fig. 3:

$$(A_2 \vee A_4) \wedge (\overline{A}_2 \vee \overline{A}_3) \wedge (A_1 \vee A_3 \vee \overline{A}_4). \tag{4}$$

EXAMPLE 2 An application of the OCAT algorithm can be illustrated by using a new example, say, $e_{11} = [0\ 0\ 1\ 0]$. When e_{11} is 'tested' by the above CNF expression, then it can be seen that e_{11} is classified as a negative example. This is as follows. The clause $A_2 \vee A_4$ evaluates to 0 because e_{11} does not contain neither the second nor the fourth atoms. On the other hand, both clauses $\overline{A}_1 \vee \overline{A}_3$ and $A_1 \vee A_3 \wedge \overline{A}_4$ evaluate to 1. However, when the three clauses are taken together, expression (4) evaluates to 0, thus indicating that e_{11} is a negative example. □

Conclusions and Future Research. This article illustrated some contributions of optimization for solving the document classification problem. These contributions were illustrated by presenting three cases (application areas) in which optimization has been used. The first case dealt with the principle of least effort (PLE) which is used for the selection of an indexing vocabulary based solely in the frequency of the co-occurring words. The second case dealt with the vector space model (VSM) for the selection of an indexing vocabulary that minimizes the overlapping of words used in various documents (or in various document classes). The third case illustrated the one clause at a time (OCAT) algorithm for the classification of documents into mutually exclusive classes.

A common characteristic of these three cases is the huge amounts of information that need to be processed before optimal solutions can be found. Therefore, the optimization techniques pre-

sented in these examples have been used extensively only on document classification problems of small size. The main reason for this limitation is that even with the current computerized technologies, these techniques would take unacceptable processing times to find optimal solutions for larger classification problems. As a consequence, scientific research efforts have focused their attention in developing effective and efficient heuristics for solving problems of more realistic size.

See also: **Boolean and fuzzy relations; Checklist paradigm semantics for fuzzy logics; Alternative set theory; Finite complete systems of many-valued logic algebras; Optimization in Boolean classification problems; Inference of monotone Boolean functions; Linear programming models for classification; Statistical classification: Optimization approaches; Mixed integer classification problems.**

References

[1] BROPHY, P., BUCKLAND, M.K., AND HINDLE, A.: *Reader in operations research for libraries*, Inform. Handling Services, Englewood, CO, USA, 1976.

[2] CHEN, H.: 'A machine learning approach to document retrieval: An overview and experiment', *Working Paper Univ. Arizona, College of Business Administration* (1996).

[3] CHEN, H., HSU, R., ORWING, R., HOOPES, L., AND NUMAMAKER, J.F.: 'Automatic concept classification of text from electronic meetings', *Comm. ACM* **30**, no. 10 (1994), 55–73.

[4] CLEVELAND, D., AND CLEVELAND, A.D.: *Introduction to indexing and abstracting*, Libraries Unlimited, Littleton, CO, USA, 1983.

[5] CROFT, W.B.: 'Knowledge-based and statistical approaches to text retrieval', *IEEE Expert* **8**, no. 2 (1993), 8–12.

[6] FOX, CH.: 'A stop list for general text', *Special Interest Group on Information Retrieval* **24**, no. 1–2 (1990), 19–35.

[7] JACOBS, P.S.: 'Using statistical methods to improve knowledge-based news categorization', *IEEE Expert* **8**, no. 2 (1993), 13–23.

[8] JACOBS, P.S., AND RAU, L.: 'SCISOR: Extracting information from on-line news', *Comm. ACM* **33**, no. 11 (1990), 88–97.

[9] KIM, J-T., AND MOLDOVAN, D.I.: 'Acquisition of linguistic patterns for knowledge-based information extraction', *IEEE Trans. Knowledge and Data Engin.* **7**, no. 5 (1995), 713–724.

[10] KORFHAGE, R.R., AND OLSEN, K.A.: 'The role of visualization in document analysis': *Third Annual Symposium on Document Analysis and Information Retrieval*, UNLA/ISRE, Las Vegas, NV, USA, 1994, pp. 199–207.

[11] KRAFT, D.H., AND BOYCE, R.B.: *Operations research for the libraries and information agencies. Techniques for the evaluation of management decision alternatives*, Acad. Press, 1991.

[12] LEE, J.H., KIM, M.H., AND LEE, Y.J.: 'Enhancing the fuzzy set model for high quality document ranking', *Microprocessing and Microprogramming* **35** (1992), 337–334.

[13] LUHN, H.P.: 'The Automatic creation of literature abstracts', *IBM J. Res. Developm.* **2** (1958), 159–165.

[14] MEADOW, CH.T.: *Text information retrieval systems*, Acad. Press, 1992.

[15] MERIDIAN, D.: 'Declassification productivity initiative study report', *Techn. Report DynCorp Comp.* (1996), prepared for the U.S. Dep. of Energy, Germantown, MD, USA.

[16] MERRIAN-WEBSTER: *Collegiate dictionary*, tenth ed., Mirrian-Webster, 1993.

[17] MOLINARY, A., AND PASSI, G.: 'A fuzzy representation of HTML documents for information retrieval systems': *Proc. Fifth IEEE Internat. Conf. Fuzzy Systems*, 1996, pp. 197–112.

[18] NASH, S.G., AND SOFER, A.: *Linear and nonlinear programming*, McGraw–Hill, 1996.

[19] RAU, L.F., AND JACOB, P.S.: 'Creating segmented databases from free text for text retrieval': *Proc. Fourteenth Annual Internat. ACM/SIGIR Conf. Research and Development in Information Retrieval*, ACM, 1991, pp. 337–346.

[20] RIGGS, F.W.: 'Delphic language: A problem for authors and indexers', *Library Sci.* **28**, no. 1 (1991), 18–30.

[21] SALTON, G.: *Automatic information organization and retrieval*, McGraw–Hill, 1968.

[22] SALTON, G.: *Automatic text processing. The transformation, analysis, and retrieval of information by computer*, Addison-Wesley, 1989.

[23] SALTON, G., WONG, A., AND YANG, C.S.: 'A vector space model for automatic indexing', *Comm. ACM* **18**, no. 11 (1975), 613–620.

[24] SCHNEEWEISS, W.: *Boolean functions with engineering applications and computer programs*, Springer, 1989.

[25] SWANSON, D.R., AND BOOKSTEIN, A.: *Operations research: Implications for libraries*, Univ. Chicago Press, 1971.

[26] TAHA, H.A.: *Operations research: An introduction*, sixth ed., MacMillan, 1997.

[27] TAKKINEN, J.: 'An adaptive approach to text categorization and understanding - A preliminary study', *Working Paper Dept. Computer and Information Sc. Linkoping Univ.* (1996).

[28] TRIANTAPHYLLOU, E.: 'Inference of a minimum'

Boolean function from examples by using a new efficient branch-and-bound approach', *J. Global Optim.* **5** (1994), 64–94.

ZIPF, H.P.: *Human behavior and the principle of least effort*, Addison-Wesley, 1949.

Salvador Nieto Sanchez
Dept. Industrial and Manufacturing Systems Engin.
3128 CEBA Building
Louisiana State Univ.
Baton Rouge, LA 70803–6409, USA
Evangelos Triantaphyllou
Dept. Industrial and Manufacturing Systems Engin.
3128 CEBA Building
Louisiana State Univ.
Baton Rouge, LA 70803–6409, USA
E-mail address: trianta@lsu.edu
Web address: www.imse.lsu.edu/vangelis

MSC2000: 90C09, 90C10
Key words and phrases: document classification, computational linguistics, indexing terms, context descriptors, text classification, keywords, document surrogate, principle of least effort, PLE, vector space model, VSM, one clause at a time algorithm, OCAT, indexing vocabulary, optimal indexing vocabulary, semantic analysis methodologies, word patterns, conjunctive normal form, CNF, disjunctive normal form, DNF.

OPTIMIZATION IN LEVELED GRAPHS

A *k-leveled graph* or a *k-level hierarchy* is defined as a graph $G = (V, E) = (V_1, \ldots, V_k, E)$ with vertex sets $V_1, \ldots, V_k$, $V = V_1 \cup \cdots \cup V_k$, $V_i \cap V_j = \emptyset$ for $i \neq j$, and an edge set E connecting vertices in levels V_i and V_j with $i \neq j$ $(1 \leq i, j \leq k)$. V_i is called the *ith level*. In a geometric representation of a k-leveled graph, the vertices in each level V_i are drawn on a horizontal line L_i with y-coordinate $k - i$, and the edges are drawn strictly monotone, i.e., an edge $(v_i, v_j) \in E$, $v_i \in V_i$, $v_j \in V_j$, $i < j$, is drawn with decreasing y-coordinates. Essentially, a k-leveled graph is a k-partite graph that is drawn in a special way.

A *proper k-leveled graph* is a k-leveled graph $G = (V_1, \ldots, V_k, E)$ in which any edge in E connects vertices in two consecutive levels V_i and V_{i+1} for $i \in \{1, \ldots, k-1\}$. Fig. 1 shows a proper leveled graph on $k = 4$ levels. This graph represents the face lattice of the *cuboctahedron* [4].

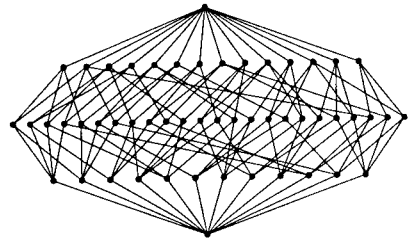

Fig. 1.

Optimization problems in leveled graphs arise in applications in *computational biology* and in *automatic graph drawing*.

Multiple Sequence Alignment. In computational biology the vertices in each level V_i represent letters of a sequence S_i over a finite alphabet Σ. The optimization problem which arises is the *multiple sequence alignment* problem. Here, the k sequences $S_1, \ldots, S_k$ should be aligned so that the cost of the alignment is maximized. An alignment can be interpreted as an array with k rows, one row for each S_i. Two letters of distinct sequences are said to be aligned if they are placed in the same column. There are many ways to measure the quality of an alignment, leading to different problem formulations. One of them is the *maximum weight trace* formulation introduced in [14]. Here, the letters of the sequences $S_i = (s_{i1}, \ldots, s_{in_i})$ are viewed as vertices in level i in a k-leveled graph $G = (V_1, \ldots, V_k, E)$. Every edge $e \in E$ has a nonnegative weight representing the gain of aligning the endpoints of the edge. We say that an alignment $\widehat{S}$ *realizes* an edge if it places the endpoints into the same column of the alignment array.

The set of edges realized by an alignment $\widehat{S}$ is called the *trace* of $\widehat{S}$, and the weight of an alignment $\widehat{S}$ is the sum of the weights of all edges in the trace of $\widehat{S}$. The goal is to compute an alignment $\widehat{S}$ of maximum weight.

The maximum weight trace problem is *NP*-hard, and can be solved in polynomial time for fixed k. A dynamic programming approach gives an algorithm with time complexity $O(k^2 2^k N)$ and space complexity $O(N)$, where $N = \prod_i n_i$, which is feasible only for very small problem instances. J. Kececioglu [15] presented a branch and bound

Nieto, S.N. and E. Triantaphyllou, (2001), **"Optimization in Document Classification,"** *Encyclopedia of Optimization,* (P.M. Pardalos and C. Floudas, Eds.), Kluwer Academic Publishers, Boston, MA, U.S.A., Vol. 4, pp. 182-189.