

# Finding Hotspots in Document Collection

Wei Peng

School of Computer Science  
Florida International University  
FL 33199, USA  
wpeng002@cs.fiu.edu

Tao Li

School of Computer Science  
Florida International University  
FL 33199, USA  
taoli@cs.fiu.edu

Chris Ding

Lawrence Berkeley National Lab  
University of California Berkeley  
CA 94720, USA  
chqding@lbl.gov

Tong Sun

Xerox Innovation Group  
Xerox Corporation  
NY 14580, USA  
Tong.Sun@xerox.com

## Abstract

*Given a document collection, it is often desirable to find the core subset of documents focusing on a specific topic. We propose a new algorithm for this task. Document clustering aims at partitioning the document-term datasets into different groups by optimizing certain objective functions. However, they are not suitable for finding hotspots that are described by a small set of documents with few tightly coupled terms. In this paper we propose a novel hotspot finding algorithm, DCC (Dense Concept Clustering) in document collections. DCC can extract distinct small topics with most representative documents and words simultaneously. The hotspots are dense bicliques in binary document-word matrices and they can be discovered sequentially one at a time using the generalized Motzkin-Straus formalism. The representative documents and words are tightly correlated for concept descriptions. Experiments on real document datasets show the effectiveness of the proposed algorithm.*

## 1 Introduction

Given a document collection, it is often desirable to find the *core* subsets of documents focusing on specific topics. This is like the usual knowledge discovery process in which

we learn something new by concrete and often exciting examples, rather than a generic summary of the topics. In this paper we propose a new algorithm for this task.

Our approach is close to biclustering [3, 17, 14]. Before discussing biclustering, we begin with data clustering. Clustering has been a popular method for discovering hidden structures. Intuitively, clustering aims at partitioning a finite set of points in a multi-dimensional space into groups (called clusters) so that (i) the points belonging to the same group are *similar* to each other while (ii) the points belonging to different groups are *dissimilar* [11]. Many algorithms (e.g., K-means) have been proposed to perform one-side clustering, i.e., group the data points into clusters. Co-clustering algorithms have been proposed to explore the correlations and simultaneously cluster data points and features [5, 18, 6, 13]. However, they cluster data into several clusters simultaneously using a single global objective.

Both K-means and co-clustering type global single-objective clustering methods do not work efficiently in high dimensional spaces due to the *curse of dimensionality*. In addition, as demonstrated in [1, 15], the correlations among the dimensions are often specific to data locality. For example, for document collections, some document groups are correlated with a given set of terms and others are correlated with respect to different terms [12]. The subspace clustering deal with these situations in a global (single-objective for all clusters) way.

Very recently, bi-clustering algorithms have been proposed to explore the correlations between points and features in *local* manner, i.e., computing one cluster at a time focusing entirely on local property [3, 17, 14]. This approach is widely used in bioinformatics.

In this paper we propose a new approach in exploring

the very *strong local* correlations between points and features. Consider the following scenario: when some important events happen, e.g., the earthquake or tsunami disaster, we want to quickly find the representative descriptions of the event from thousands of new articles. These emerging events are usually described by a subset of articles with certain specific terms. We call such transient trends/topics in document collections as *hotspots*. Generally, hotspots are described by a set documents with a few specific terms and all these terms appear very frequently in all those documents. Traditional single-global objective clustering algorithms partition the documents (and terms) into different groups at the same time by optimizing a certain objective function. They are not suitable for finding the emerging trend or topic from the collection.

More specifically we propose a novel algorithm *DCC* for finding hotspots in document collections. The hotspots are indicated as bicliques or biclusters in binary document-term matrices [2]. They represent distinct topics in document collections, and are explicitly supported/described by both representative subgroups of documents and representative subgroups of words. The hotspots can overlap with each other. They are discovered sequentially, one after another and thus we can find high quality trends and topics.

The Motzkin-Straus formalism is generalized in *DCC* to approximately compute the maximal-edge bicliques with the largest areas. The computational complexity is  $O(|E|)$  where  $|E|$  is the number of edges (i.e., the co-occurrences of documents and terms). *DCC* can be easily extended to favor biclusters with more documents less words, or vice versa, by adjusting the constraint parameters.

The remainder of the paper is organized as follows. In Section 2, we first introduce the definition of the biclique, and give an illustrating example to compare the differences between hotspots and clusters. We then present the algorithm procedure by using the Motzkin-Straus formalism with an iterative process in Section 3. In Section 4, we present our experimental results on six real document datasets. Finally Section 5 concludes.

## 2 Notations and Formulation

### 2.1 Problem Formulation

Given a binary document-word matrix  $\mathbf{S}$ , the document index set  $D$ , and the word index set  $W$ , an ideal biclique is  $(D_i, W_j)$  where  $D_i \subset D$  and  $W_j \subset W$  such that every pair of the document and the word from  $D_i$  and  $W_j$  co-occurs. The biclique is presented as a rectangle sub-matrix  $\mathbf{S}_{D_i, W_j}$  with all 1's in  $\mathbf{S}$ . The number of document-word co-occurrence is  $|D_i| \times |W_j|$ , the area of  $\mathbf{S}_{D_i, W_j}$ . The maximal biclique in our paper is the biclique with the largest area.

### 2.2 An Illustrating Example

In this section, we use a simple dataset to illustrate the differences between hotspots and clusters. The simple dataset contains the following six system log messages from two different situations: **Start** and **Create**.

#### Start Situation

**S1:** User temporary application start successfully

**S2:** The Database version for the application service started

**S3:** The shell service has just started an application version 2.1

#### Create situation

**C1:** Can not create temporary file version 1.5 for the application service

**C2:** Create temporary product configuration service version 2.0

**C3:** Can not create temporary service for starting the Oracle engine

Messages/Terms	T1	T2	T3	T4	T5	T6
S1	1	1	0	0	0	1
S2	1	1	1	1	0	0
S3	1	1	1	1	0	0
C1	0	1	1	1	1	1
C2	0	0	1	1	1	1
C3	1	0	0	0	1	1

**Figure 1. Log message example: The 6 terms are start, application, version, service, create, temporary respectively.**

After removing stop words and words only appear once, we get the binary document-term matrix as shown in Figure 1. For this example, we observe the following:

- The hotspots are the dense regions (i.e., bicliques) denoted by subblocks of all 1's. 3 hotspots, e.g., G1, G2 and G3, can be identified in our example using *DCC*. G1 is the situation of “start application”, G2 is the situation of “create temporary”, while G3 represents the situation of “service version”.
- Using one-side clustering algorithms, e.g., K-means, we generally get two clusters where the first three messages are in one cluster and the last three messages are in another cluster.
- The bi-clustering algorithms<sup>1</sup> aim at minimizing the sum squared residue [3, 4, 17] or maximizing the mutual information between the biclusters and the original data [6]. Using bi-clustering algorithms, e.g., information-theoretical bi-clustering, we could get both message clusters and term clusters. In this example, T1 and T2 belong to one term cluster while T5 and T6 belong to another term cluster. T3 and T4 are treated as feature noise by most bi-clustering algorithms as they have similar distributions across multiple clusters.
- The hotspots discovered in our algorithm are different from the bi-cluster in several different ways. First, hotspots are at a lower granularity than the bi-clusters and thus they are more compact as the messages and terms are tightly connected/associated to each other. Second, hotspots can span multiple clusters (e.g., G2 in the example). Third, hotspot is able to capture the information that is usually discarded or ignored by traditional bi-clustering algorithms (e.g. G2). Fourth, hotspots can overlap and they are discovered sequentially, one after another. Finally, the documents and terms in hotspots are tightly coupled. The hotspots can be easily explained using the corresponding messages and terms and the concept descriptions obtained from hotspots are generally more accurate and precise.

More detailed discussions and comparisons between bi-clustering and Bicliques are presented in Appendix A.

### 3 The Algorithm on Finding Hotspots

We now describe the algorithm for discovering hotspots. The algorithm is inspired by the maximal *clique* identification. The Motzkin-Straus theorem was used to find maximal *cliques* [8, 16].

<sup>1</sup>For simplicity, in the rest of the paper, we do not distinguish between bi-clustering and co-clustering and use bi-clustering to denote the family of algorithms that perform clustering both points and features simultaneously.

### 3.1 Hotspots in Document-Word Matrices

The document collection can be represented as a binary document-term matrix where each entry is 1 or 0 denoting whether the corresponding document and term co-occur or not. The document-term matrix can be expressed as a unweighted bipartite graph with a set of *document* nodes and a set of *term* nodes. If a document contains a term, an edge exists to connect them. Finding the maximal bicliques obtains the hotspots described by both highly correlated documents and words. *DCC* generalizes the Motzkin-Straus formalism on bipartite graph for computing the maximum-edge biclique. The maximum-edge biclique is computed via the solution to the following optimization problem:

$$\max_{d \in F_d^\alpha, w \in F_w^\beta} d^T S w, \quad (1)$$

where  $F_d^\alpha = \{d \in R^n \mid \sum_{i=1}^n d_i^\alpha = 1, d_i \geq 0\}$ ,  $n$  is the number of documents, and  $F_w^\beta = \{w \in R^m \mid \sum_{j=1}^m w_j^\beta = 1, w_j \geq 0\}$ ,  $m$  is the number of words.

We can find the maximal edge biclique by using Generalized Motzkin-Straus Theorem as follows:

**Theorem 1** *If  $(d^\#, w^\#)$  is an optimal solution of  $(d, w)$ , the nonzero elements of  $d^\#$  have the same values, and nonzero elements of  $w^\#$  have the same values, then  $S_{D_1, W_1}$  is the maximum edge biclique in  $S$ , where  $D_1 = \{i \mid d_i^\# > 0\}$ , and  $W_1 = \{j \mid w_j^\# > 0\}$ . The optimal value  $J^\#$  of Equation 1 is  $|D_1|^{1-1/\alpha} |W_1|^{1-1/\beta}$ .*

We can prove that the optimal value is

$$J^\# = |D_1|^{1-1/\alpha} |W_1|^{1-1/\beta}.$$

Because  $\alpha > 1$  and  $\beta > 1$ ,  $|D_1|$  and  $|W_1|$  are maximized simultaneously. If  $\alpha = \beta$ , then  $J^\# = (|D_1| |W_1|)^{1-1/\alpha}$ , and  $S_{D_1, W_1}$  is the maximal edge biclique where the biclique area  $|D_1| |W_1|$  is maximized [7].

From the optimal value  $O$ , we note that by setting different values to  $\alpha$  and  $\beta$  we are able to obtain different shapes of the computed maximal biclique. If  $\alpha > \beta$ , then  $1 - 1/\alpha > 1 - 1/\beta$ . The algorithm favors the bicliques with more rows than columns.

### 3.2 Algorithm Description

The following algorithm procedure is used to compute the optimal solution to Equation 1 *iteratively*.

**Initialization:** The initial  $d^{(0)}$  and  $w^{(0)}$  are given  $(1 \cdots 1)^T$ .  $\alpha = 1 + \epsilon_1, 0 < \epsilon_1 \ll 1, \beta = 1 + \epsilon_2, 0 < \epsilon_2 \ll 1$ . We set  $\alpha = \beta = 1.1$  for finding the maximal edge bicliques in our experiments. The maximal bicliques can be visualized as the blocks along the matrix diagonal. Every

time after a maximal edge biclique is found, there are two ways to initialize the next round. One is to mask the corresponding sub-matrix entries with zeros, e.g.  $\mathbf{S}_{D_1, W_1} = 0$ , and discover the hotspots with overlapping document and word sets. The other is to mask all corresponding rows and columns with zeros, e.g.,  $\mathbf{S}_{D_1, W} = 0$  and  $\mathbf{S}_{D, W_1} = 0$  and the discover the hotspots with non-overlapping document and word sets.

**Iterative Process:**  $d$  and  $w$  can be updated by using:

$$d_i^{(t+1)} = (d_i^{(t)} \frac{(\mathbf{S}w^{(t)})_i}{(d^{(t)})^T \mathbf{S}w^{(t)}})^{1/\alpha}, \quad (2)$$

$$w_j^{(t+1)} = (w_j^{(t)} \frac{(\mathbf{S}^T d^{(t)})_j}{(d^{(t)})^T \mathbf{S}w^{(t)}})^{1/\beta}. \quad (3)$$

$d$  and  $w$  can be proved correct and convergent. Due to page limitation, we will not specify them here.

### 3.3 Illustration of The Algorithm

To illustrate the typical run of the iterative process, Table 1 and Table 2 show the algorithm on an example document-term matrix  $\mathbf{S}$  below.

$$\mathbf{S} = \begin{bmatrix} \left| \begin{array}{cc} 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{array} \right| & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \left| \begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array} \right| \end{bmatrix}$$

These two tables show the iteration steps of the algorithm. By setting  $\alpha = 1.1$  and  $\beta = 1.1$ , we find the maximal edge biclique as shown in Table 1. The maximum biclique in the case includes the last three documents and the last three terms. If  $\alpha$  is set to be bigger than  $\beta$ , then the biclique with more rows is preferred as shown in Table 2. The maximum biclique in the case includes the first four documents and the first two terms.

### 3.4 Other Implementation Issues

**Relaxation:** The bicliques are sub-matrices with all entries 1. Their sizes could be very small in real datasets. If more documents and terms are needed to describe the hotspot, the relaxation on the maximal biclique can be employed to allow a small number of zeros inside the bicliques. The idea is straightforward: in each round of finding the maximal biclique, we select the documents and terms which are corresponding to the elements with the largest values in vectors  $c$  and  $w$ . Suppose we obtain a maximum edge biclique which have the document set  $D_1$  corresponding to the highest values  $\{d_1, d_2, \dots, d_p\}$  in  $d$ , and the term set  $W_1$  corresponding to the highest values  $\{w_1, w_2, \dots, w_q\}$  in  $w$ .

In order to add one more document in this biclique, we can select the document having the  $(p + 1)$ -th highest value in  $d$  because compared to the other documents it should have more correlation with the term set  $W_1$ . Similarly, we could choose the terms with the  $(q + 1)$ -th highest value in  $w$  if we want one more term to explain the hotspot.

**Stop Criterion:** If we want bicliques to have all entries with 1's, the stop criterion is to impose the size restriction *SizeRes* on the bicliques. When the size of the biclique falls down under a threshold *SizeRes*, the algorithm stops.

Once zeros are allowed in bicliques, we use the following stop criterion. Given a binary  $n \times m$  matrix  $\mathbf{B}$ , and a function  $F$  for calculating density of 1 in the matrix

$$F(\mathbf{B}) = \frac{|\{(i, j) \mid \mathbf{B}_{ij} == 1, \forall i, j\}|}{nm}. \quad (4)$$

The threshold density percentage  $P$  can be set to  $0 < P \ll 1$  such that when  $F(\mathbf{S}_{D_1, W_1}) < P \cdot F(\mathbf{S})$ , the algorithm stops.

## 4 Experimental Results

We compare *DCC* with the tradition one-way K-Means clustering, and several bi-clustering algorithms including information theoretic bi-clustering algorithm [6], Euclidean bi-clustering algorithm, and minimum squared residue bi-clustering algorithm [4] on six real world datasets to show that *DCC* extracts more meaningful hotspots. Each hotspot is composed of representative documents and representative terms. The experiments show that documents in each hotspot have higher purity, and terms are more descriptive and concise. We also present a case study on mining customer requirements.

### 4.1 Dataset Descriptions

The characteristics of the six experimental datasets are listed in Table 3.

**Table 3. Dataset Description.**

Datasets	# Documents	# words	# Classes
CSTR	476	1000	4
WebKB4	4199	1000	4
WebKB7	8280	1000	7
Reuter	2900	1000	10
WebAce	2340	1000	20
Log	1367	200	8

**CSTR** This is the dataset of the abstracts of technical reports (TRs) published in the Department of Computer Science at a research university. The TRs are available at <http://www.cs.rochester.edu/trs>. The dataset contained 476

**Table 1. The results of the 2nd,10th,20th,30th,and 50th iterations by Algorithm DCC running on the example matrix S.  $\alpha$  and  $\beta$  are set to 1.1.**

$d^{(2)}$	$w^{(2)}$	$d^{(10)}$	$w^{(10)}$	$d^{(20)}$	$w^{(20)}$	$d^{(30)}$	$w^{(30)}$	$d^{(50)}$	$w^{(50)}$
0.0983	0.2385	0.0004	0.0977	0	0.0383	0	0.007	0	0
0.0983	0.2385	0.0004	0.0977	0	0.0383	0	0.007	0	0
0.0983	0.0295	0.0004	0	0	0	0	0	0	0
0.4213	0.0295	0.8371	0	0.7465	0	0.5764	0	0.4014	0
0.2208	0.2122	0.1104	0.3161	0.1647	0.3498	0.26	0.3655	0.3517	0.3683
0.2208	0.2122	0.1104	0.3161	0.1647	0.3498	0.26	0.3655	0.3517	0.3683
	0.2122		0.3161		0.3498		0.3655		0.3683
$(d^T \mathbf{S} w)^{(2)}$		$(d^T \mathbf{S} w)^{(10)}$		$(d^T \mathbf{S} w)^{(20)}$		$(d^T \mathbf{S} w)^{(30)}$		$(d^T \mathbf{S} w)^{(50)}$	
0.8037		1.1643		1.1837		1.2084		1.2208	

**Table 2. The results of the 2nd,10th,15th, 20th,and 30th iterations by Algorithm DCC running on the example matrix  $S_2$  given in Section 2.  $\alpha$  is set to 1.7, and  $\beta$  to 1.1.**

$d^{(2)}$	$w^{(2)}$	$d^{(10)}$	$w^{(10)}$	$d^{(15)}$	$w^{(15)}$	$d^{(20)}$	$w^{(20)}$	$d^{(30)}$	$w^{(30)}$
0.2495	0.2551	0.2416	0.2972	0.3103	0.3964	0.4111	0.5182	0.4422	0.5325
0.2495	0.2551	0.2416	0.2972	0.3103	0.3964	0.4111	0.5182	0.4422	0.5325
0.2495	0.0287	0.2416	0	0.3103	0	0.4111	0	0.4422	0
0.5381	0.0287	0.6803	0	0.6529	0	0.5243	0	0.4432	0
0.3695	0.2014	0.2674	0.1867	0.1769	0.1147	0.0277	0.015	0	0
0.3695	0.2014	0.2674	0.1867	0.1769	0.1147	0.0277	0.015	0	0
	0.2014		0.1867		0.1147		0.015		0
$(d^T \mathbf{S} w)^{(2)}$		$(d^T \mathbf{S} w)^{(10)}$		$(d^T \mathbf{S} w)^{(15)}$		$(d^T \mathbf{S} w)^{(20)}$		$(d^T \mathbf{S} w)^{(30)}$	
1.3669		1.5120		1.5686		1.8124		1.8848	

abstracts, which were divided into four research areas: Natural Language Processing(NLP), Robotics/Vision, Systems, and Theory.

**WebKB7** The WebKB7 dataset contains webpages gathered from university computer science departments. There are about 8280 documents and they are divided into 7 categories: student, faculty, staff, course, project, department and other. The raw text is about 27MB. Among these 7 categories, student, faculty, course and project are four most populous entity-representing categories. The associated subset is typically called **WebKB4**.

**Reuters** The Reuters-21578 Text Categorization Test collection contains documents collected from the Reuters newswire in 1987. In our experiments, we use a subset of the data collection which includes the 10 most frequent categories among the 135 topics and we call it **Reuters-top 10**.

**WebACE** The dataset was from WebACE project and has been used for document clustering [9]. It contains 2340 documents consisting news articles from Reuters new service via the Web in October 1997. These documents are divided into 20 classes.

**Log** The dataset contains 1367 log text messages collected from desktop machines. The log messages are grouped into 8 categories, i.e., *configuration, connection, create, dependency, report, request, start, and stop*.

In all our experiments, we select the top 200 words for log dataset and 1000 words for the rest datasets by occurring frequency.

## 4.2 Evaluation Measures

Our DCC algorithm does not cluster all documents and all words. It extracts the hotspots, each of which contains a small set of most representative documents and words. We assume that documents of a hotspot come from a particular topic (class). Purity can then be used to measure the extent to which each hotspot contained documents from primarily one class [19].

$$Purity = \sum_{i=1}^K \frac{n_i}{n} P(S_i), P(S_i) = \frac{1}{n_i} \max_j (n_i^j), \quad (5)$$

where  $S_i$  is a particular hotspot or cluster with  $n_i$  documents,  $n_i^j$  is the number of documents of the  $j$ -th class that

were assigned to the  $i$ -th hotspot or cluster,  $K$  is the number of hotspots or clusters and  $n$  is the total number of extracted documents. In general, the larger the values of purity, the better the documents can describe the hotspot. To measure whether the hotspots really have representative terms, we compare the extracted terms with the top ranked frequent keywords in the corresponding classes.

### 4.3 Results Analysis

We run *DCC* with equal constraint parameters  $\alpha = \beta = 1.1$  on Log dataset. We stop finding bicliques when the number of biclique edges drops below 40. We obtain 18 bicliques after removing the bicliques with the number of words or documents less than 5. We compare the purity of *DCC* with several other clustering algorithms. We observe that the extracted hotspots cover all classes in Log dataset and achieve the purity 0.972 much higher than other algorithms. Since the log dataset has been labeled manually and each message is attached with a class label, we can obtain the frequent keywords in each class. These keywords are compared with the associated terms of the corresponding hotspot. We found these terms cover the most top ranked keywords. We list the terms of 10 bicliques in Table 4. Although many hotspots are from the same class, they provide compact interpretations with different context focuses and aspects. For example in Table 4 terms under “Dep1” and “Dep2” are both from the same class *Dependency*. “Dep1” describes the situation that the directory cannot be contacted while “Dep2” describes that the file cannot be opened.

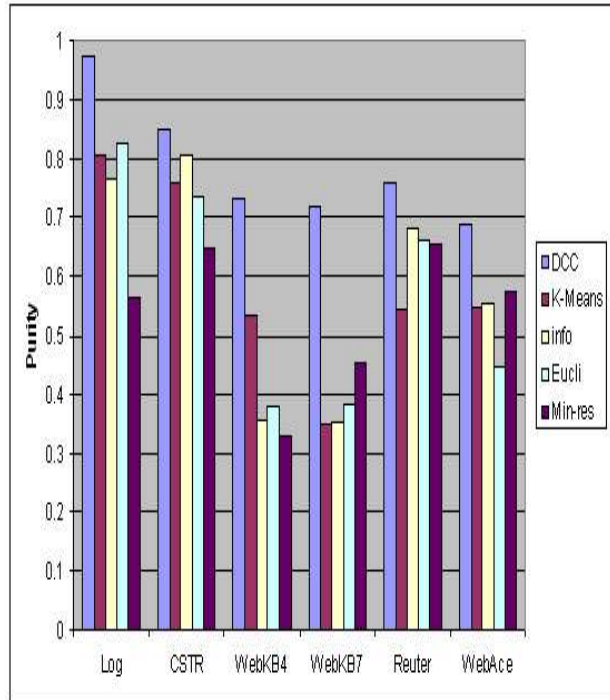
We run *DCC* on dataset CSTR, WebKB4, WebKB7, Reuter, and WebAce by setting  $\alpha = \beta = 1.1$  as well. Each biclique is composed of 20 documents and 20 words which are corresponding to the highest scored elements of vector  $c$  and  $w$  in Equation 1. The algorithm stops when the density of the maximum biclique drops under 0.3 which is almost 10 times of the original document-word matrix density. After each round, all corresponding rows and columns of the maximum biclique are masked with zeros. Therefore we obtain non-overlap maximum bicliques. 8 maximum bicliques are discovered in CSTR dataset. The discovered hotspots on WebKB4, WebKB7, Reuter, and WebAce are 22, 20, 14, and 29, respectively.

The purity results are presented in Table 5 and Figure 2. We observe that the purity of *DCC* algorithm is higher than all other algorithms. Especially in two WebKB datasets, *DCC* achieves significant purity increases.

The terms in a particular concept are ranked to indicate how typical of the term can describe the hotspot. We lists the 10 top ranked terms of 6 different hotspots of CSTR dataset in Table 6. The top ranked terms are turned out to be the most frequent words in some corresponding classes, and they are more descriptive than other words for the cor-

	DCC	K-Means	Info	Eucli	Min-res
Log	0.972	0.806	0.768	0.827	0.566
CSTR	0.850	0.758	0.807	0.735	0.651
WebKB4	0.732	0.534	0.358	0.381	0.330
WebKB7	0.718	0.351	0.353	0.384	0.453
Reuter	0.757	0.545	0.684	0.661	0.656
WebAce	0.688	0.546	0.553	0.447	0.575

**Table 5. The purity results of *DCC*, K-Means, Information Theoretic bi-clustering (Info), Euclidean bi-clustering (Eucli), and minimum squared residue bi-clustering (Min-res) on six datasets.**



**Figure 2. Purity Comparisons.**

**Table 4. The associated terms of 10 hotspots in Log dataset.**

Dep1	Dep2	Conn1	Conn2	Start	Create	Config	Report	Request	Stop
fail	file	failed	ldiscn	service	create	configuration	version	product	shell
exist	unable	auto	inventory	user	temporary	product	fault	completed	stop
certificate	open	update	server	profile	file	complete	module	install	unexpected
enrollment	output	party	landeskcore	start	ldiscn	webfldrs	address	update	explorer
system		root	respond	version			profile		restart
contact		sequence		cleanup			application		
active		number							
directory		error							
domain		retrieval							

responding hotspots. Note that there are three hotspots in *System* category. *System1* concerns about memory and processor management for system performance, *System2* is about process synchronization, and *System3* is mainly about caching. These hotspots provide compact interpretations and better topic descriptions.

## 5 Conclusion

In this paper, we propose a novel algorithm *DCC* to find hotspots in document collections. We use generalized Motzkin-Straus formalism to find the maximum edge bicliques which can be thought as the dense concepts with representative document sets and representative term sets. The algorithm has several advantages. First, it is simple, easy to implement. It also discovers hotspots sequentially, one at a time, thus be able to obtain quality results. Second, it can discover the hotspots with both highly representative documents and terms simultaneously as shown in our experiments. Third, it is efficient to find the maximum edge biclique with the computational complexity  $O(|E|)$ , where the  $|E|$  is the number of edges in this biclique. Fourth, it is also very flexible to favor different shapes of bicliques. We compare our algorithm with several existing bi-clustering algorithms and encouraging experimental results have been obtained to show the effectiveness of our algorithm.

## Appendix A: Bi-Clustering vs Bicliques

In order to give more insight on why other bi-clustering algorithms are not suitable to find the dense regions, we present some detailed example comparisons here. Suppose we have two document-word matrices  $\mathbf{S}_1$  and  $\mathbf{S}_2$ :

$$\mathbf{S}_1 = \left[ \begin{array}{cc|cc} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{array} \right]$$

$$\mathbf{S}_2 = \left[ \begin{array}{cc|ccc} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array} \right]$$

The dense regions, bicliques which are denoted as the subblocks of all 1's between vertical lines can be found by the algorithm *DCC*. Consider the biclustering introduced by Cheng and Church [3]. They defined the **mean square residue** which is the sum of squared differences between each entry in the bicluster and the corresponding row mean and the column mean plus the mean of the bicluster. They find the maximal bicluster with the minimal mean square residue. Suppose we have bicluster  $(D_i, W_j)$  on the matrix  $\mathbf{S}$ . The mean square residue of this bicluster is

$$\|\mathbf{H}\| = \sum_{m \in D_i, n \in W_j} h_{mn}^2, \quad (6)$$

where

$$h_{mn} = \mathbf{S}_{mn} - \mathbf{S}_{mW} - \mathbf{S}_{Dn} + \mathbf{S}_{DW}. \quad (7)$$

$\mathbf{S}_{mW} = \sum_{n \in D} \mathbf{S}_{mn} / |D|$ ,  $\mathbf{S}_{Dn} = \sum_{m \in W} \mathbf{S}_{mn} / |W|$ , and  $\mathbf{S}_{DW} = \sum_{m \in W, n \in D} \mathbf{S}_{mn} / |D| |W|$ , where  $|D|$  and  $|W|$  are the cardinality of  $D$  and  $W$ . By minimizing the overall mean square residue of the matrix  $\mathbf{S}_1$ , we can get the "perfect" score zero of row clustering  $\langle 112222 \rangle$  and column clustering  $\langle 1122 \rangle$ . Obviously the result has no desirable bicliques. Actually it tends to find many undesirable bicliques in binary data. The direct clustering by Hartigan [10] has the different mean square residue which is the sum of squared differences between each entry in the bicluster and the mean of the bicluster. In direct clustering,  $h_{mn}$  can be defined as:

$$h_{mn} = \mathbf{S}_{mn} - \mathbf{S}_{DW}. \quad (8)$$

**Table 6. The associated words of 6 concepts.**

System1	System2	System3	Theory	Robotics	NLP
performance	program	cache	polynomial	visual	train
share	lock	load	np	image	language
memory	synchronization	coherent	complexity	environment	network
parallel	process	exploit	prove	vision	learn
multiprocessor	significant	share	set	human	temporal
application	compare	reduce	turing	world	knowledge
hardware	study	scalable	task	movement	plan
data	busy	increase	log	perception	implementation
software	technique	size	base	robot	action
distribute	sparse	cycle	scene	task	finite
machine	implement	logic	access	separation	run
collapse	alternative	wait	plan	model	segmentation

By minimizing the residue measure, the matrix  $S_1$  can be correctly partitioned into two bicliques. However, the direct clustering on the matrix  $S_2$  will obtain  $\langle 111222 \rangle$  for row clustering and  $\langle 112222 \rangle$  for column clustering. The first maximal bicluster with the minimum residue can be the large up right-corner sub-matrix with all 0's inside. However, it is definitely not what we want to find. Other biclustering algorithms such as [17] minimizing the one of the two above mean square residue have the same drawbacks in finding dense concepts.

The bi-clique approach in this paper can clearly discover the dense regions.

## Acknowledgments

W. Peng and T. Li is partially supported by NSF CAREER Award IIS-0546280 and a Xerox UAC (University Affairs Committee) award. C. Ding is supported by the US Dept of Energy, Office of Science.

## References

- [1] C. Aggarwal, C. Procopiu, J. Wolf, P. Yu, and J. Park. Fast algorithms for projected clustering. *SIGMOD*, pages 61–72, 1999.
- [2] I. Bomze, M. Budinich, P. Pardalos, and M. Pelillo. The maximum clique problem. In *Handbook of Combinatorial Optimization*, volume 4. Kluwer Academic Publishers, Boston, MA, 1999.
- [3] Y. Cheng and G. M. Church. Biclustering of expression data. In *ISMB*, pages 93–103. AAAI Press, 2000.
- [4] H. Cho, I. Dhillon, Y. Guan, and S. Sra. Minimum sum squared residue co-clustering of gene expression data. In *Proceedings of The 4th SIAM Data Mining Conference*, pages 22–24, Lake Buena Vista, Florida, April 2004.
- [5] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *KDD'01*, pages 269–274, 2001.
- [6] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretical co-clustering. In *KDD'03*, pages 89–98, 2003.
- [7] C. Ding, Y. Zhang, T. Li and S. R. Holbrook. Biclustering Protein Complex Interactions with a Biclique Finding Algorithm. In *ICDM*, pages 178–187, 2006.
- [8] L. E. Gibbons, D. W. Hearn, P. M. Pardalos, and M. V. Ramana. Continuous characterizations of the maximum clique problem. *Math. Oper. Res.*, 22(3):754–768, 1997.
- [9] E. Han, D. Boley, M. Gini, R. Gross, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore. WebACE: A web agent for document categorization and exploration. In *Agents'98*. ACM Press, 1998.
- [10] J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129, March 2001.
- [11] T. Li. A general model for clustering binary data. In *KDD'05*, pages 188–197, 2005.
- [12] T. Li, S. Ma, and M. Ogihara. Document clustering via adaptive subspace iteration. In *SIGIR*, pages 218–225, 2004.
- [13] B. Long, Z. Zhang, and P. Yu. Co-clustering by block value decomposition. In *KDD'05*, pages 635–640, New York, NY, USA, 2005. ACM Press.
- [14] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 1(1):24–45, 2004.
- [15] L. Parsons, E. Haque, and H. Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explor. Newsl.*, 6(1):90–105, 2004.
- [16] M. Pelillo. Relaxation labeling networks for the maximum clique problem. *J. Artif. Neural Netw.*, 2(4):313–328, 1995.
- [17] J. Yang, H. Wang, W. Wang, and P. Yu. Enhanced biclustering on expression data. In *BIBE'03*, pages 321–327, Washington, DC, USA, 2003. IEEE Computer Society.
- [18] H. Zha, X. He, C. Ding, M. Gu, and H. Simon. Bipartite graph partitioning and data clustering. *CIKM 2001*, 2001.
- [19] Y. Zhao and G. Karypis. Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning*, 55(3):311–331, 2004.