Domain Thesaurus Construction from Wikipedia*

WenKe Yin [1], Ming Zhu [2], TianHao Chen [2]

[1] Department of Electronic Engineering and Information Science, University of Science and Technology of China, HeFei 230027, China
[2] Department of Automation, University of Science and Technology of China, HeFei 230027, China
{ywk & cth}@mail.ustc.edu.cn, mzhu@ustc.edu.cn

Abstract - The domain thesaurus plays an important role in information retrieval, natural language processing, question answering system etc. Due to the complexity of the natural language, the NLP based thesaurus constructing methods are difficult to achieve a desired result. In recent years, Wiki has been widely used as a knowledge base. Based on the characteristics anchor description and topic locality of hyperlinks, this paper proposes a hyperlink structure graph clustering based domain thesaurus construction method. The method first constructs a domain-specific hyperlink structure graph using Wiki, and then uses LSI algorithm to calculate the weight of each hyperlink. Then our method uses CPMw algorithm to cluster the weighted undirected hyperlink structure graph. After this step, domain thesaurus can be achieved. Experiments show that our method can get better results.

Index Terms - Domain Thesaurus; Wiki; CPMw; LSI;

1. Introduction

Domain thesaurus is comprised of by vocabularies and the relationship between vocabularies. Domain thesaurus has important applications in information retrieval, natural language processing, question answering system, and so on. Currently, the lack of semantic knowledge has become an important obstacle to the natural language applications. Generic domain thesaurus, such as WordNet and HowNet only covers limited knowledge. Besides, these two thesauruses are manually built, so updating and maintaining will be a laborious job.

To solve this problem, researchers have carried out numerous researches about NLP based domain thesaurus automatically building. Such methods usually deal with massive documents using NLP techniques first, and then extract the concepts and relationship between the concepts using statistical methods. However, due to the complexity of the natural language, the accuracy of NLP is usually limited, which resulting a poor quality of the constructed domain thesaurus.

With the vigorous development of the Internet, some researchers began to try building domain thesaurus automatically via the Internet. Compared with the documents, the biggest difference is that web pages have hyperlinks. Paper [1] pointed out that hyperlinks have two important characteristics: anchor description and topic locality. Anchor description means that the anchor text is usually a good description of the theme of the targeting page. Topic locality means that two web pages linked together by a hyperlink are more likely to have the same topic. Therefore, if the page is replaced with the corresponding anchor text, the linked texts usually are related to a same subject, which is the basic idea of constructing domain thesaurus by analyzing the hyperlink structure.

In recent years, web knowledge base has been widely used, and one of the most famous is Wikipedia. Wiki is the largest interactive online encyclopaedia. Wiki has the following four advantages: (i) massive pages, so far Wiki China has already contains 537,265 web pages; (ii) frequent updates, Wiki is updated and maintained by the Internet users; (iii) dense hyperlinks, paper [2] indicates that each Wiki page contains an average of 29.96 hyperlinks pointing to other Wiki pages; (iv) good anchor description, the anchor text in Wiki is often a word. Therefore, this article will use Wiki as the knowledge source to build domain thesaurus.

The paper is organized as follows. Section 2 provides some related work on domain thesaurus construction. Section 3 introduces some algorithms that will be used in our method. Section 4 provides details about the method that we used. In section 5, the experimental method and results are given. Section 6 draws conclusions and presents future perspectives.

2. Related Work

Artificially constructing domain thesaurus is the most common method. WordNet [3] organizes words with the same meaning as the synsets. WordNet summaries each synset with a brief definition, and records the semantic relationship between synsets. HowNet [4] uses Chinese word as the description object, and aims to reveal the concept and the relationship between the properties of concepts. The disadvantage of artificially constructing domain thesaurus is that it is very time consuming, and updating and maintaining cost is high. To solve this problem, researchers carried out numerous researches about automatically building domain thesaurus. Next, we will introduce some classic methods. These methods fall into three categories: NLP-based methods, Web mining based method, Wiki mining based methods.

A. NLP-Based Domain Thesaurus Construction

H.Chen [5] et al proposed a method used to automatically build a common thesaurus in electronic community system. The method uses word filtering, automatic indexing, and clustering analysis technology. Pedersen [6] et al proposed co-occurrence algorithm which represented a word as a vector in multidimensional space to capture the co-occurrence information between words. Words are defined as similar, if the two words have similar co-occurrence patterns. Y.H.Tseng

* This work is supported by National Science and Technology Support Program #2011BAH11B01.

[7] proposed a method automatically building Chinese thesaurus from documents. The article first proposed a keyword extraction algorithm. The extracted keywords will be further used for the analysis of the relationship between words. TextRank [8] is a graph-based text sorting model, and the basic idea of TextRank comes from PageRank. The above methods all used NLP techniques such as segmentation, POS tagging. However, due to the complexity of natural language, such as synonyms, the accuracy of NLP is limited, which will directly affect the quality of the thesaurus. With the development of Internet, some researchers try to automatically build domain thesaurus via Internet.

B. *Web Ming Based Domain Thesaurus Construction*

Web link structure mining is widely used in web ranking. PageRank [9] and HITS [10] algorithm are the representatives. In recent years, the use of web link structure mining to automatically build the domain thesaurus gradually attracted the attention of researchers. Z.Chen [11] proposed a novel approach to automatically constructing a domain-specific thesaurus using link structure information. First, a set of high quality and representative websites of a specific domain is selected. After filtering out navigational links, link analysis is applied to each website to obtain its content structure. Finally, the thesaurus is constructed by merging the content structures of the selected websites. The method is faced with two problems: NLP and scalability. NLP will be used in merging content structures for word segmentation and concept matching. Besides, when merging content structures, the method will build a lot of sub-trees. If the algorithm is used for a large website, it will produce a large number of sub-trees, which bring huge overhead in memory and time.

C. *Wiki Ming Based Domain Thesaurus Construction*

Wiki mining has recently become a new research focus. Paper [12] first described the structure of Wikipedia, then analyzed and compared the principles and methods of acquiring concepts and instances by utilizing category structure graph, information box and definition sentence. Masahiro Ito [13] et al. proposed a method for constructing thesaurus from Wikipedia based on link co-occurrences. They also propose integration method of tfidf and link co-occurrence analysis. Nakayama [14] et al. proposed lfibf algorithm for calculation of the correlation between two nodes in the Wiki link structure graph. The idea of lfibf comes from tfidf. Paper [15] presented experiments on using Wikipedia for computing semantic relatedness and compare it to WordNet on various benchmarking datasets.

3. Background Knowledge

This section will briefly introduce the graph knowledge that will be used, as well as the LSI and CPMw algorithm.

A. *Web Graph*

Web can be abstracted as a graph, the node represents the web page, and the edge represents the hyperlink.

Definition 1 An undirected graph is a two-tuple <*V*, *E*>: (i) *V* is the set of vertices; (ii) *E* is the collection of edges. If each edge <v_i, v_j> in *E* has a weight w_{ij}, then *G* is entitled weighted graph.

B. *Latent Semantic Indexing*

Latent semantic indexing (LSI) [16] is an intelligent information retrieval model, and is also a method for automatic knowledge extraction and representation. LSI adopts vector space model, and applies singular value decomposition (SVD) to reduce the dimension of the high dimensional sparse vocabulary-document matrix.

Definition 2 Let *A* is an *m*×*n* real matrix, then the square root of the non-zero eigenvalues of matrix A^TA are called singular value.

Theorem 1 Let $A\square R^{m\times n}$, and its rank is *r*, then there will be an orthogonal matrix *U* of order *m* and an order *n* orthogonal matrix *V*, so that the following equation holds:

$$U^T AV = \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \quad (1)$$

Expression $A = U\begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix}V^T$ is the SVD of matrix *A*.

For a document collection containing *m* feature words and *n* documents, an *m*×*n* vocabulary-document matrix $A=\{a_{ij}\}_{m\times n}$ can be constructed. Each row represents a feature vector of the word, and each column represents a feature vector of the document. Matrix *A* is sparse, where *m*>>*n*. Let rank(*A*) = *r*, then the SVD of *A* can be obtained from theorem 1:

$$A = TSD^T \quad (2)$$

Each column of *T* is orthogonal, and the length is 1, i.e. $T^TT = I$. The row vectors of *T*, which correspond to the feature vectors the words, are called left singular vector. *S* is a single-valued diagonal matrix, which is called singular value standard form of matrix *A*:

$$S = diag(\lambda_1, \lambda_2, ..., \lambda_n) \quad (3)$$

λ_i in S meets $\lambda_1 \ge \lambda_2 \ge ... \ge \lambda_r \ge \lambda_{r+1} = ... = 0$, and λ_i is also the single-value of *A*. Each column of *D* is orthogonal, and their length are all 1, i.e. $D^TD = I$. The row vectors of *D*, which correspond to the feature vectors the documents, are called the right singular vector. Taking the top-*k* columns from *T* and *D* to construct *k*-rank approximation matrix:

$$A=T_kS_kD_k^T \quad (4)$$

After this step the dimension of original matrix is reduced, and noise information can be efficiently removed.

C. *Clique Percolation Method with Weights*

Clique percolation method with weights (CPMw) [17] is the improvement of clique percolation method (CPM) for the weighted network. CPM is an undirected graph clustering algorithm, and CPM can effectively identify the overlapping structure of the complex network. Here we first give a few definitions.

Definition 3 For a given undirected graph *G*=<*V*, *E*>, *Clique* is a complete subgraph of *G*. If clique *A* is not contained by any other clique, then clique *A* is a *maximal clique*. *K-clique* is a clique of size k.

Definition 4 If two cliques share at least k-1 nodes, then these two cliques are *k-clique adjacent*.

Definition 5 *k-clique-community* is a union of all k-cliques that can be reached from each other through a series of adjacent k-cliques.

The purpose of CPM is to identify all the k-clique-communities from the graph, and each k-clique-community corresponds to a cluster, so finding all k-clique-communities is equivalent to the clustering of the graph. The algorithm of CPM is as follows: (i) extracting all the maximal cliques from the graph; (ii) preparing clique-clique overlap matrix; (iii) erasing every off-diagonal entry smaller than k-1 and every diagonal element smaller than k in the matrix; (iv) replacing the remaining elements by one; (v) merging all the cliques, which correspond one in the matrix, in the same row and column. Finally, the isolated communities are the clustering results. Figure 1 is a simple illustration of the extraction of the *k*-clique-communities at *k* = 4.

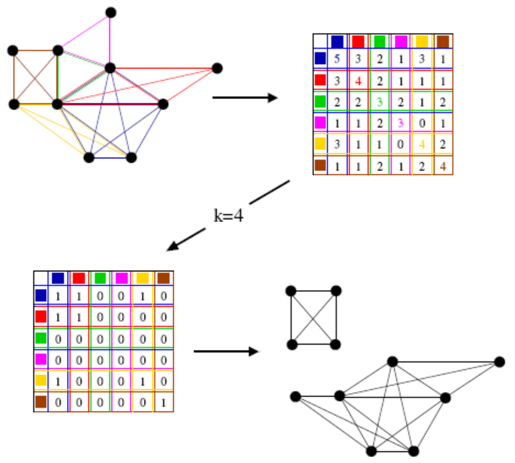


Fig. 1 A simple illustration of the extraction of the *k*-clique-communities

In order to find all maximal cliques, CPM repeatedly chooses a node, extracts every clique of this size containing that node, then deletes the node and its edges. When no nodes are left, the clique size is decreased by one and the clique finding procedure is restarted on the original graph. For the problem extracting all cliques of a size containing a special node, CPM adopts backtracking method to solve it.

The choice of the parameter *k* in CPM algorithm is very important. If *k* is too small, the extracting clusters will be few and the cluster size will be large, so that the rich structure information will be lost. If *k* is too large, the extracting clusters will also be very few. The article gives a heuristic rule for the choice of *k*. If the largest community is about twice as big as the second largest one, the *k* can be considered preferably. For the weighted graph, CPMw algorithm introduces the concept of clique intensity.

Definition 6 For a *k*-clique, its clique intensity is defined as:

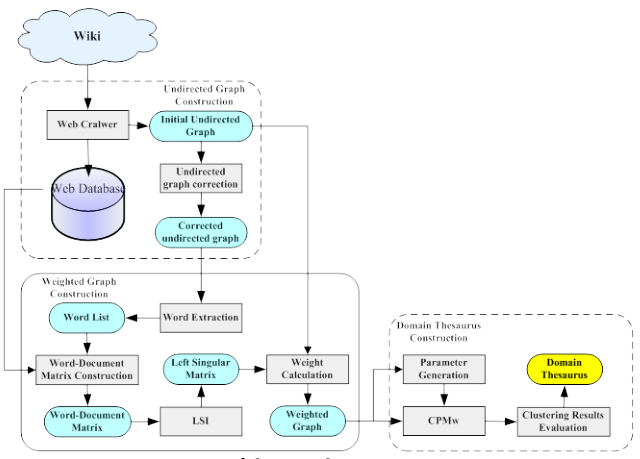
$$I(c) = \left(\prod_{\substack{i<j \\ i,j \in C}} w_{ij} \right)^{2/k(k-1)} \qquad (5)$$

where w_{ij} represents the weight of edge (v_i, v_j).

For each clique found in CPM, CPMw computes clique intensity using Equation 4, and only retains the cliques whose clique intensity is greater than a specified threshold *I*. The choice of parameter *I* is also very important. CPMw fixed parameter *k*, and then clustered using the edge weight as threshold from the largest one to the minimum one. If the largest community is about twice as big as the second largest one, the parameter *I* can be considered preferably.

4. Domain Thesaurus Construction

This chapter details our domain thesaurus construction method. This method is comprised of three parts including: undirected graph construction, weighted graph construction and domain thesaurus construction. The process of out method is shown as Figure 2.



Wiki
Undirected Graph Construction
Web Cralwer
Initial Undirected Graph
Web Database
Undirected graph correction
Corrected undirected graph
Weighted Graph Construction
Word List
Word Extraction
Word-Document Matrix Construction
Left Singular Matrix
Weight Calculation
Word-Document Matrix
LSI
Weighted Graph
Domain Thesaurus Construction
Parameter Generation
Domain Thesaurus
CPMw
Clustering Results Evaluation

Fig. 2 Process of domain thesaurus construction

This method first carries out depth limited bread first traversal of the Wiki from a specified URL to build the hyperlink structure graph on a particular area, and then using web database and word list to construct word-document matrix. After that we calculate the similarity between nodes using LSI, and set it as the weight of the corresponding edge. Finally, we utilize CPMw to cluster the weighted graph, evaluate clustering results, and generate the final domain thesaurus.

A. Undirected Graph Construction

Wiki contains four kinds of web pages: the entry page, non-created entry page, special page and list page. The entry page is used to define and describe a particular concept. The URL addresses of these four kinds of web pages all have fixed formats. As for the entry page, the URL is always in line with the "http://zh.wikipedia.org/wiki/*" format, and '*' denotes a concept. Web crawler will only extract the hyperlink of the entry page, and it is easy to that using regex which meets the above rules. In addition, Web crawler will save the crawled pages into web database, and this database will be used for building vocabulary-document matrix.

Let the current page title is T_i and the link address is U_i, and we denotes the parsed out link and anchor text as T_j and U_j, then the format of the initial undirected graph saved as a text file is "T_i U_i T_j U_j". Initial undirected graph still needs to be corrected by undirected graph correction module. The reason for doing this is that there may be two problems in initial undirected graph. For example, music page contains a link "rhythm and blues", but in fact the link corresponds to the page title as "rhythm and blues". In addition there will be the phenomenon of a same URL having different anchor texts, which will lead to a URL may correspond to two or more nodes in the graph.

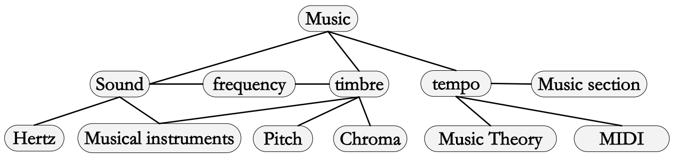


Fig. 3 Undirected graph about "Music"

We propose the following method to solve the above problems: sequentially scan the initial undirected graph; for each link U_i , record its first appeared anchor text T_i; replace all the anchor texts of U_i as T_i. After this step, we can get the corrected undirected graph. The format of the corrected undirected graph is like "T_i T_j". Figure 3 shows a small part of the corrected undirected graph about "music".

Finally, we have to note that web crawler will limit the search depth, and in this paper we usually set it as 2 or 3. There are two reasons for doing this: (i) as the search depth increases, the subject of the link will drift from the initial theme. For example, "Music" page contains "Sound", but "Sound" page contains "Ultrasonic", "Infrasound" etc, which are unrelated with "Music"; (ii) large depth limit will lead to huge undirected graph, which will cause a heavy time and space overhead of the subsequent processing.

B. Weighted Graph Construction

First, we use word extraction module to build word list which contains all the vocabularies that appeared in the corrected undirected graph. Then word-document matrix construction module will build the word-document matrix using web database and word list. Element a_{ij} in the matrix represents the number of occurrences that word v_i appeared in document D_j.

Next, we use formula (2) (3) (4) to carry out matrix singular value decomposition. After this step, we can get left singular matrix T_k, and each row in T_k corresponds to an eigenvector of a word. Then weight calculation computes the similarity of two nodes as the weight of the edge. Here we use cosine similarity. For an edge $<v_i, v_j>$ in the graph, assuming there corresponding left singular vectors are T_{ki} and T_{kj}, then the calculation of the weight can be represented as follows:

$$sim(v_i, v_j) = \cos\theta = \frac{\sum_{m=1}^{k} w_{im} \times w_{jm}}{\sqrt{(\sum_{m=1}^{k} w^2{}_{im})(\sum_{m=1}^{k} w^2{}_{jm})}} \tag{6}$$

w_{im} and w_{jm} are the m-dimensional value of T_{ki} and T_{kj}. Figure 4 shows the constructed weighted undirected graph about "Music".

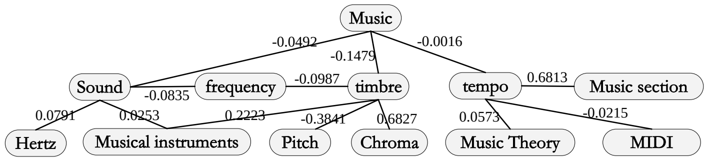


Fig. 4 Weighted undirected graph about "Music"

Finally we explain our selection method of the parameter k in equation (4): For the eigenvalues calculated using formula (2), the sum of the top-k largest eigenvalues should be just over 90% of the sum of all the eigenvalues. This method can efficiently reduce the dimension of the original high-dimensional matrix. For the word-document matrix about "Music", the dimension of original uncompressed singular matrix T is 8954×125. When using our parameter selection method, the dimension of left singular matrix can be compressed into 8954×53.

C. Domain Thesaurus Construction

It is not difficult to find in figure 4 that the similarity calculated using LSI sometimes is inconsistent with the semantic relationship. For example, the similarity between "music" and "tone", "tone" and "pitch" are both negative. For this case, we make a little adjustment to CPMw algorithm. We first compute the value of $\prod_{i<j;\, i,j \in c} w_{i,j}$. If this value is negative, then abandon clique C. Otherwise, calculate $I(c)$ using Equation (5), and retain it only when $I(c)$ is greater than the threshold I.

One advantage of CPMw is that even if the edge weight between two nodes is negative, these two nodes may still be clustered together, as long as the number of negative weight is even in the clique. CPMw fixed parameter k, and then clustered using the edge weight as threshold from the largest one to the minimum one. If the largest community is about twice as big as the second largest one, the parameter I can be considered preferably. Paper [17] describes that in general k values from 3 to 6. Obviously the efficiency of this method is low, so this article uses a different method of parameter I choosing method.

First we fixed the value of k, and then sorted the edge weight in descending order. Let the number of edges in the graph is N, the parameter generation module will select the first 0.9×N, 0.7×N, 0.5×N, 0.3×N and 0.1×N weight as the candidates of parameter I. The reason for doing so is that the threshold I will neither be too large (too large threshold will case all cliques will not meet), nor be too small (too small will case all cliques will meet). So it must be between the maximum and minimum. Generally, parameter *k* ranges from 3 to 6. Therefore, the improved CPMw algorithm needs totally 20 times of clustering in different parameters.

At last, clustering results evaluation will assess the 20 different clustering results. We made a little change to the evaluation method of CPMw: (i) calculate the 20 ratios of the largest community and second largest one respectively; (ii) let the i-*th* ratio is σ_i, and sort $|\sigma_i -2|$ in descending order; (iii) denote the ratio corresponding to the smallest $|\sigma_i -2|$ as $\sigma_{\min}$; (iv) judge whether there is a clustering result in the remaining 19 clustering results meeting $\||\sigma_i -2|-|\sigma_{\min} -2\|<0.1$; (v) if there is no clustering result meeting that, then return the $\sigma_{\min}$ corresponding one as the final clustering result. If there is one, assume it is the cluster result *m*, and then jump to the next step; (vi) if the ratio of the size of cluster result *m* and $\sigma_{\min}$ corresponding cluster result is greater than 1.2, then we return cluster result *m* as the final cluster result. Otherwise, return $\sigma_{\min}$ corresponding one as as the final cluster result; (vii) if clustering results satisfying the conditions in step (vi) is more than one, then select the largest of one to return.

We prefer large clustering result to small clustering result, because larger clustering result can better retain and reflect the slight difference in word semantics. This is helpful for understanding the polysemy of the word. At last, the domain thesaurus can be easily constructed using the cluster that contains the specified word, for example "Music".

5. Experiments

This chapter will test and analyze the performance of our domain thesaurus method quantitatively.

A. Experimental data and metrics

First, we built Wiki hyperlink structure graphs about five areas, which included: "music", "car", "computer", "mobile phone" and "Internet. We set the search depth limit of the web crawler as 2. The statistics on these five undirected graphs are shown in Table 1.

TABLE I Statistics on the undirected graph

Concept	Number of nodes	Number of edges
Music	8954	12137
Car	3666	4612
Computer	10441	19242
Mobile Phone	12551	21042
Internet	13828	24232

For different areas, we used our proposed method, TextRank, Co-occurrence, LSI and lfibf to construct domain thesauruses respectively. Referencing the evaluation method proposed in paper [14], this paper uses concept precision (CP) to evaluate the performance of the above algorithms. Given a query word, the algorithms return the most relevant top-50 words respectively. The correlation of the words returned and query word will be scored by users. The correlation ranges from 1 to 5. 5 point indicates very relevant, and 1 point indicates very irrelevant. CP is defined as follows:

$$CP = \frac{\text{Number of retrieved relevant concepts}}{\text{Number of total retrieved concepts}} \tag{7}$$

The word is defined as relevant if the correlation is scored 4 or 5. We used the above five concepts as the query words, and 250 words were returned. We invited 5 people scoring the correlation of these 250 words respectively. The final score is the average of the five people's scoring.

Given any query word, the method returning the most relevant top-50 words is: (i) get all the words which are in the same cluster with the query word; (ii) use left singular matrix and cosine similarity to calculate the similarity between query word and the other words; (iii) sort these words in descending order by the similarity, and return the first 50 words.

B. Experimental Results and Analysis

Table 2 shows the parameters which the optimal clustering results selected by the clustering results evaluation module correspond to.

TABLE II Parameters on the optimal clustering results

Concept	*k*	*I*	Number of Clusters	Ratio of the the largest community and the second one
Music	4	0.2731	20	1.78
Car	3	0.4121	16	2.5
Computer	6	0.1732	4	1.58
Mobile Phone	3	0.7052	28	1.79
Internet	6	0.0972	5	2.22

It is not difficult to see from the table that we can get a satisfactory result just trying 5 different clique intensity thresholds *I* using our parameter generation method. While using the original algorithm, we need to try out all the weights in the weighted graph.

Table 3 shows part of the clustering result about "Music". As can be seen, "Music" can belong to more than one cluster, and the semantic of each cluster is different. A concept can belong to multiple clusters, thus able to retain and reflect the difference of semantics between the words, which is the biggest advantage of CPMw algorithm.

TABLE III Part of the clustering result about "Music"

Cluster1:	Music, Rhythm, Melody, Interval, Harmony, Music Theory
Cluster2:	Music, Europe, Renaissance, Baroque Music, Concerto Grosso, Polyphonic Music, Neoclassical Music, Romantic Music
Cluster3:	Music, Folk Song, Vocal, Folk Music
Cluster4:	Music, Rock and Roll, Pop Music, Modern Music, Country Music

Table 4 shows the CP value of different algorithms. As can be seen from the table, the CP value of our method is significantly better than the other algorithms from top-10 and top-50. Part of the reason is that the method used in this paper utilizes both the link structure information and text information, and other algorithms only use one of them. For example, LSI and Co-occurrence algorithm only used the text information, while TextRank and lfibf algorithm only took advantage of the link structure information. In the rest of the four algorithms, the CP value of TextRank is better than the others, but there is still a wide gap with our method.

TABLE IV CP of different algorithms

Algorithm	Top-10	Top-20	Top-30	Top-40	Top-50
Our Method	96.0%	96.0%	94.7%	91.5%	87.2%
LSI	84.0%	74.0%	72.0%	69.5	66%
TextRank	80.0%	80.0%	76.7%	77.5%	74.0%
lfibf	78.0%	76.0%	72.7%	71.5%	68.4%
Co-occurrence	74.0%	69.0%	67.4%	65.5%	61.8%

6. Conclusion

Based on the anchor description and topic locality of hyperlink, we propose a dictionary construction method based on the Wiki link structure graph clustering. This method first carries out depth limited bread first traversal of the Wiki from a specified URL to build the hyperlink structure graph on a particular area, and then using the web database and word list to construct word-document matrix. After that we calculate the similarity between nodes using LSI, and set it as the weight of the corresponding edge. Finally, we utilize CPMw to cluster the weighted graph, evaluate clustering results, and generate the final domain thesaurus. The experimental results show that our proposed method is superior to the other algorithms.

Building domain thesaurus is just the first step of our work. In the future, we expect to build domain ontology automatically via Internet. Building ontology will not only need to extract the relevant vocabularies, but also need to identify the relationship between terms, such as "parent-of" "part-of", "instance-of" and so on. Besides, further optimizing the clustering speed of CPMw algorithm is also the problem that we will consider.

Acknowledgment

This work is supported by National Science and Technology Support Program #2011BAH11B01.

References

[1] Craswell, Nick, David Hawking, and Stephen Robertson. "Effective site finding using link anchor information." Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2001.
[2] Nakayama, Kotaro, Takahiro Hara, and Shojiro Nishio. "A thesaurus construction method from large scaleweb dictionaries." Advanced Information Networking and Applications, 2007. AINA'07. 21st International Conference on. IEEE, 2007.
[3] Fellbaum, Christiane. "WordNet." Theory and Applications of Ontology: Computer Applications (2010): 231-243.
[4] Dong, Zhen-Dong, Qiang Dong, and Chang-Ling Hao. "Theoretical findings of HowNet." Journal of Chinese Information Processing 21.4 (2007): 3-9.
[5] Chen, Hsinchun, et al. "Automatic thesaurus generation for an electronic community system." (1995).
[6] Schütze, Hinrich, and Jan O. Pedersen. "A cooccurrence-based thesaurus and two applications to information retrieval." Information Processing & Management 33.3 (1997): 307-318.
[7] Tseng, Yuen‐Hsien. "Automatic thesaurus generation for Chinese documents." Journal of the American Society for Information Science and Technology 53.13 (2002): 1130-1138.
[8] Mihalcea, Rada, and Paul Tarau. "TextRank: Bringing order into texts." Proceedings of EMNLP. Vol. 4. 2004.
[9] Page, Lawrence, et al. "The PageRank citation ranking: bringing order to the web." (1999).
[10] Kleinberg, Jon M. "Authoritative sources in a hyperlinked environment." Journal of the ACM (JACM) 46.5 (1999): 604-632.
[11] Chen, Zheng, et al. "Building a web thesaurus from web link structure." Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval. ACM, 2003.
[12] Yu Chuanming and Zhang Xiaoqing. "Learning Ontology from Wikipedia:Principles and Methods." Journal of the China Society for Scientific and Technical Information 30.3 (2011): 244-252.
[13] Ito, Masahiro, et al. "Association thesaurus construction methods based on link co-occurrence analysis for wikipedia." Proceedings of the 17th ACM conference on Information and knowledge management. ACM, 2008.
[14] Nakayama, Kotaro, Takahiro Hara, and Shojiro Nishio. "Wikipedia mining for an association web thesaurus construction." Web Information Systems Engineering–WISE 2007 (2007): 322-334.
[15] Strube, Michael, and Simone Paolo Ponzetto. "WikiRelate! Computing semantic relatedness using Wikipedia." Proceedings of the National Conference on Artificial Intelligence. Vol. 21. No. 2. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.
[16] Ruiz-Casado, Maria, Enrique Alfonseca, and Pablo Castells. "Automatic assignment of wikipedia encyclopedic entries to wordnet synsets." Advances in Web Intelligence (2005): 947-950.
[17] Farkas, Illés, et al. "Weighted network modules." New Journal of Physics 9.6 (2007): 180.