A Graph-Based Information Retrieval Model

Ohm Sornil[*]

Abstract

The goal of information retrieval is to effectively retrieve documents relevant to users' queries. A variety of models and techniques have been proposed over the past 50 years. In this research, we introduce a novel, graph-based approach to information retrieval. Its computation is fast and scalable, and its structure is flexible to incorporate many performance enhancement techniques. The performance evaluation according to the HARD track of TREC 2003 is reported.

Keywords information retrieval, graph, random walk with restart

1. Introduction

Information retrieval (IR) concerns with representations, storage, organization, and access to documents. The goal of an information retrieval system (generally called a search engine) is to retrieve documents relevant to users' information needs, typically presented to the system in form of queries. It has now been realized that IR plays a very important role in the desktop and World Wide Web applications.

Before a search system is in operation, words are recognized and extracted from documents in the collection, passed through the stopword elimination, stemming, and other processes according to the features of the system, such as phrase construction, term canonicalization, term expansion, etc. An index is then built from those terms with an indexing process.

Users employ a user interface provided by the system to specify their information needs in forms of queries. Queries are processed according to query operations, e.g., stopword elimination, stemming, and further transformed into the representation used by the system, normally in a similar format to that of the index. Then the searching process employs the index to identify a set of documents that potentially are relevant to the query. During the search, the system may give a matching score for each document. Once a

[*] Assistance Professor, Department of Computer Science, School of Applied Statistics, National Institute of Development Administration, Bangkok, Thailand, Email: osornil@as.nida.ac.th

set of documents has been identified and scored, they are ranked upon their scores and presented to the user through the interface. After, the result set is returned to the user, some systems provide methods for the user to refine his query by examining the results, marking those documents in the result set that are considered relevant, and resubmitting it into the system through the relevance feedback process. The system then calculates another result set and returns it to the user.

A major task of an information retrieval system is to predict which documents are relevant, and which are not, according to a user's information need. This decision usually depends upon similarity calculation intrinsic to the information retrieval model which attempts to establish an ordering of the documents retrieved. A number of models have been proposed over 50 years, for example:

Vector model (Salton, 1971): Both the query and each document are represented as vectors in the term space. A measure of the similarity between the two vectors is computed.

Probabilistic model (Robertson & Sparck Jones, 1976): A probability based on the likelihood that a term will appear in a relevant document is computed for each term in the collection. For terms that match between a query and a document, the similarity measure is computed as the combination of the probabilities of each of the matching terms.

Language model (Baeza-Yates & Ribeiro-Neto, 1999): A language model is built for each document, and the likelihood that the document will generate the query is computed.

Inference Networks (Turtle & Croft, 1991): A Bayesian network is used to infer the relevance of a document to a query. This is based on the evidence in a document that allows an inference to be made about the relevance of the document. The strength of this inference is used as the similarity coefficient.

Boolean Indexing: A score is assigned such that an initial Boolean query results in a ranking. This is done by associating a weight with each query term so that this weight is used to compute the similarity coefficient.

Latent Semantic Indexing (Furnas et al., 1988): The occurrence of terms in documents is represented with a term-document matrix. The matrix is reduced via Singular Value Decomposition (SVD) to filter out the noise found in a document so that two documents which have the same semantics are located close to one another in a multi-dimensional space.

Neural Networks (Wilkinson & Hingston, 1991): A sequence of neurons in a network fire when activated by a query triggering links to documents. The strength of each link in the network is transmitted

to the document and collected to form a similarity coefficient between the query and the document. Networks are trained by adjusting the weights on links in response to predetermined relevant and irrelevant documents.

Genetic Algorithms (Baeza-Yates & Ribeiro-Neto, 1999): An optimal query to find relevant documents can be generated by evolution. An initial query is used with either random or estimated term weights. New queries are generated by modifying these weights. A new query survives by being close to known relevant documents and queries with less fitness are removed from subsequent generations.

Fuzzy Set Retrieval (Ogawa et al., 1991): A document is mapped to a fuzzy set (a set that contains not only the elements but a number associated with each element that indicates the strength of membership). Boolean queries are mapped into fuzzy et intersection, union, and complement operations that result in a strength of membership associated with each document that is relevant to the query. This strength is used as a similarity coefficient.

Random walk algorithms on graphs have been used in citation analysis, social networks, and the analysis of the link structure of the Web. Generally, a graph-based ranking algorithm is a way of deciding on the importance of a vertex within a graph, by taking into account global information recursively computed from the entire graph, rather than relying only on local vertex-specific information. The basic idea implemented by a random walk algorithm is that, when one vertex links to another one, it is basically casting a vote for other vertex. The higher the number of votes that are casted for a vertex, the higher the importance of the vertex.

This research proposes a graph-based information retrieval model (GIR), employing a graph structure that captures occurrences of terms in documents as well as correlations among terms, and the similarity between a document and the query are calculated by a specific type of graph random walk algorithm, called Random Walk with Restart (RWR). The model can be extended to incorporate relevance feedback naturally. Its effectiveness is evaluated using TREC collections; its performance is measured and compared to the performance figures of systems participating in the TREC program.

The rest of this article is organized as follows. Section 2 reviews related work on graph random walk algorithms in information retrieval. Section 3 describes the GIR model. In Section 4, experimental evaluations and results are discussed. Finally, Section 5 provides concluding remarks for the research.

2. Related Work

There are some information retrieval systems utilizing graphs in some fashions. Rule Based Information Retrieval by Computer (RUBRIC) (Tong et al., 1987) uses production rules to capture user query concepts. A query is specified in a hierarchical fashion which can be viewed as a set of production rules. Each rule has an associated weight, assigned by the user. RUBRIC determines relevant documents through a goal-driven search and creates a rule tree with the user's query topic at the root, subtopics are represented as branches, and the keywords to be matched with contents of documents are at the leaves. Score of each node is calculated by a fuzzy evaluation of the AND/OR operations.

Kazai, et al. (Kazai et al., 2000) present a retrieval model representing documents as graphs. A document is represented as a tree whose nodes are components of the documents and whose edges represent the relationships between those components. The model uses fuzzy aggregation, an approach based on the fuzzy representation of linguistic quantifiers, and ordered weighted averaging operators to assign scores to documents.

Ogilvie and Callan (Ogilvie & Callan, 1997) introduce a document retrieval system using a tree-based generative language model. Nodes of the tree correspond to document components, such as titles, sections, and paragraphs. Each node in the tree is associated with a language model. The model for a leaf node is estimated directly from the text presented in the document components associated with the node. An internal node is estimated by a linear interpolation of the child nodes. This approach allows for structured queries and ranking of document components. One benefit of the model includes guidance from language modeling on how to estimate the probabilities used in ranking.

Quintana et al. (Quintana et al., 1992) propose a Hypertext Retrieval System (HRS) that allows users to retrieve information from hypertext documents. This model is based on semantic contents of documents. Queries and information in hypertext documents are automatically converted to conceptual graphs. The information retrieval problem is turned into a graph matching problem. This technique is useful for the retrieval of information in large knowledge domains where a user needs to find specific information but does not know the organization of the hypertext documents or words used in the documents.

Montes-y-Gomez et al. (Montes-y-Gomes et al., 2001) adapt the idea of conceptual graphs to represent the contents of documents and query. The model compares two conceptual graphs (document

graph and query graph) by matching and measuring their similarity. The algorithm consists of two parts which are finding the intersection of the two graphs and measuring their similarity which corresponds to the size of their intersection graph. This measure is suitable for text comparison since it considers not only topical aspects of the phrases but also the relationships among elements in the texts.

In terms of graph ranking algorithms in information retrieval, their main uses are in ranking documents from link information in hypertext environments. Hypertext analysis is studied mostly to support Web search engines, in particular the ranking of the retrieved Web pages. A hyperlink from page A to page B is a recommendation of page B by the author of page A or the same topic being presented in both Web pages A and B. The hyperlink analysis supports the relevance evaluation of pages to a user query.

HITS (Hypertext Induced Topic Search) (Kleinberg, 1999) is a well known algorithm introduced by Kleinburg. The algorithm computes two scores for each web page recursively which are: hub and authority scores. Pages with high authority scores are considered to have content relevant to the query, whereas pages with high hub scores are considered to contain links to relevant content. The intuition is that a page which points to many other pages is a good hub, and a page that many pages point to is a good authority. Good authorities are those pointed to by good hubs, and good hubs are pointed to by good authorities. This mutually reinforce relationship is at the core of the algorithm.

PageRank (Brin & Page, 1998) is a prominent technique for estimating importance scores for Web pages. The PageRank value is computed by weighting each hyperlink to the Web page proportionally to the quality of the Web page containing the hyperlink. The PageRank values are calculated recursively with arbitrary initial values. PageRank algorithm is used at the heart of the Google search engine to compute page importance for every Web pages based on linkages among pages. Google uses this technique to employ the global hyperlink structure of the Web and produce better rankings of search results compared to using the content of the page alone.

Yang and Chan (Yang & Chan, 2005) develop a modified multimedia PageRank algorithm for retrieving multimedia Web objects including Web pages, images and videos. The method analyzes hyperlinks and embedded links among multimedia objects together with weights for different types of links using the PageRank algorithm.

Mihalcea and Tarau (Mihalcea & Tarau, 2006) introduce the TextRank graph-based ranking model for keyword and sentence extraction from natural language texts. The algorithm takes into account edge weights while computing the score associated with a vertex in the graph.

Graph-based ranking algorithms are used mostly in calculating term weights to represent the contribution of a term in its context. Blanco and Lioma (Blanco & Lioma, 2007) introduce the use of a random walk algorithm to weight terms in the tf-idf weighting scheme by adapting the TextRank algorithm. In this model, edge weights are not taken into account. Hassan et al. (Hassan et al., 2006) propose a random walk model considering term dependencies and their relations to the surrounding context for term weighing. Islam et al. (Islam et al., 2008) exploit the relationship of local information of a vertex (term position) and global information (information gain) and term dependency to produce term weights.

For similarity ranking, Thammasut and Sornil (Thammasut & Sornil, 2006) propose a graph-based IR system whose query can be expressed as a graph of topics/subtopics. Once the query graph is merged with the document relationship graph, a random walk algorithm is applied to determine the score of each document node. The emphasis of the work is on expressing queries as graphs, and it does not incorporate term relationships in the structure.

3. GIR: A graph-based information retrieval model

This section introduces the proposed graph-based information retrieval model (GIR). The idea is to turn the information retrieval into a graph problem. The model creates a collection graph which represents content of documents and relationships among terms, built prior to the query time. Once a user supplies a query into the system, a query node is created and connected to query terms in the graph. A graph random walk algorithm is then applied to calculate the relevance of each document to the query.

3.1 Document Processing

After words are extracted from a document, they are preprocessed in three important steps: stopword elimination (Frakes and Baeza-Yates, 1992), stemming by Porter's algorithm (Porter, 1980), and indexing.

3.2 Collection Graph Construction

Collection graph $G = (V,E)$, shown in Figure 1, is a weighted, undirected graph whose vertices $V(x) \in V$ are organized in two layers: document and term layers. A document as well as a term is represented as a vertex or node in the graph. We denote $V(d_j)$ and $V(t_i)$ as the vertex of a document d_j and that of a unique term t_i in the vocabulary, respectively. $V(t_i)$ is connected to $V(d_j)$ when term t_i appears in document d_j. A term relationship connection $E(t_i, t_k) \in V \times V$ is placed between $V(t_i)$ and $V(t_k)$ if the

bonding strength between the two terms is greater than a certain threshold. $V(t_i)$ has relationship connections placed to top k other term vertices that it has strong relationships with.

The weight w_{t_i,d_j} between $V(t_i)$ and $V(d_j)$ can be computed using one of the family of weighting algorithms known as *tf.idf* - these algorithms are based on the term frequency ($tf_{i,j}$) which is proportional to the frequency of t_i in d_j and inverse document frequency (idf_i) which is proportional to the inverse of the number of documents containing t_i (df_i). Empirical studies (Salton and Buckley, 1988) indicate that *tf.idf* approaches, while simple, are very effective. The document term weight w_{t_i,t_j} and query term weight $w_{t_i,q}$ are defined as follows:

$$w_{t_i,t_j} = \frac{\mathrm{tf}_{i,j} \cdot \log \frac{N}{\mathrm{df}_i}}{\sum_{i=1}^{|V|} (\mathrm{tf}_{i,j} \cdot \log \frac{N}{\mathrm{df}_i})^2}$$

and

$$w_{t_i,q} = \left(0.5 + \frac{0.5 \cdot tf_{i,q}}{\max_i tf_{i,q}} \right) \cdot \log \frac{N}{df_i}$$

where N is the total number of documents in the collection.

The graph structure includes the relationships/correlations among terms. To calculate these values, we adapt the association approach to automatic thesaurus construction (Kaji et al., 2000). Instead of extracting associations in the document level which may include lots of noises, in this research the calculations are carried out in the paragraph level. That is, paragraphs are regarded as the text units for association. The connection weight c_{t_i,t_k} between $V(t_i)$ and $V(t_k)$ represents the strength of the relationship between term t_i and term t_k, calculated as:

$$c_{t_i,t_k} = \log_2 \frac{df_{i,k} / \sum_{i,k} df_{i,k}}{\{df_i / \sum_i df_i\} \cdot \{df_k / \sum_k df_k\}}$$

where df_i is the occurrence frequency of term t_i, and $df_{i,k}$ is the co-occurrence frequency of terms t_i and t_k.

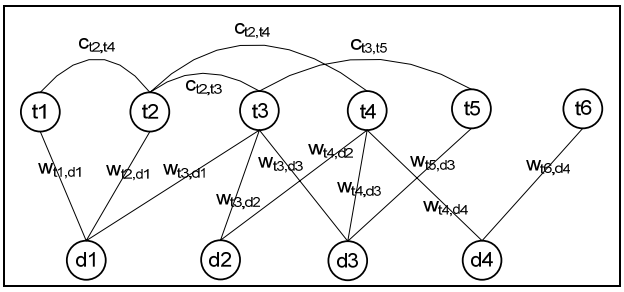

Figure 1. A collection graph

3.3 Similarity Calculation

After the collection graph is constructed, when a user supplies a query into the system, a query node *V(q)* is added to the graph, connecting to nodes corresponding to terms in the query, as shown in Figure 2.

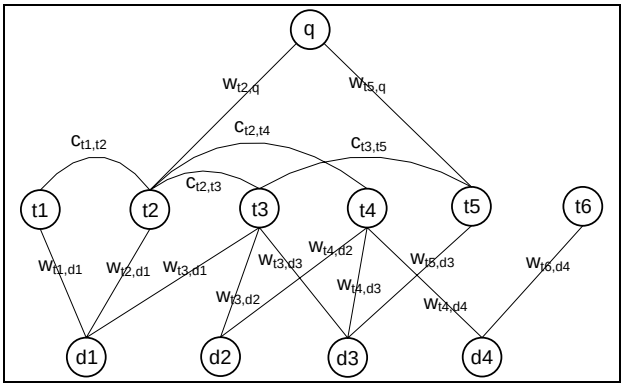

Figure 2. Structure of the graph after a query is supplied by a user

Then, the random walk with restarts (RWR) algorithm (Pan et al., 2004) is applied to calculate a similarity score between the query and each document. Documents are then ranked according to the scores, and the results are returned to the user in a descending order.

The RWR algorithm is selected since it is simple and able to bias toward the query node. The percentage of time the RWR walker spends on a document node is proportional to the closeness of the document node to the restart node. For information retrieval, we want to rank documents with respect to the query, thus the algorithm fits the problem naturally.

The algorithm operates as follows: to compute the similarity of vertex $V(d_j)$ to vertex $V(q)$, consider a random walker that starts from node $V(q)$. At each step, the walker chooses randomly among available edges, with one modification: before he makes a choice, he goes back to node q with probability c. Let $p(V(d_j))$ denotes the steady state probability that the random walker will find himself at node $V(dj)$, then $p(V(d_j))$ is the affinity (similarity/closeness) of $V(d_j)$ with respect to V(q). The system then ranks the scores in a descending fashion and returns the top M documents to the user.

The similarity scoring algorithm can be described as follows:

Initial State: The algorithm is initialized by

$U_i(0) = 0,\ 0 \le i \le n-1$ where $U_i(t)$ is the score of node i at iteration t.

$V_i(0) = 0,\ 0 \le i \le n-1$ where $V_i(t)$ is the restart of node i at iteration t.

$V_q(0) = 1$, where $V_q(0)$ is the restart of node q which is the query topic.

$U_i(0) = V_i(0), 0 \le i \le n-1$ where $U_i(t)$ is the score of node i at iteration t.

A is the adjacency matrix of the graph and is column-normalized.

c is the restart probability of restarting the random walk from the node which corresponding to the query topic.

Activation and Update State: Output of each node is calculated as follows :

$U_i(t+1) = (1-c)\, AU_i(t) + cV_i(t)$

Stable State: Repeat the iteration until convergence.

The stable state is achieved when sum of error at every node in the graph falls below a given threshold (e.g., 0.001).

Output State: After the graph converges, the resulting score for each node becomes the final similarity score of the corresponding document.

The RWR has an ability to bias toward the restart node. By setting the restart node as the query, RWR is able to rank documents according to the query topic node. The percentage of time the RWR walks spends on a document node is proportional to the closeness of the document node to the query node. This means ranking the document node with the respective similarity to the query. On the other hand, the

popular PageRank method (Brin & Page, 1998) may not be appropriate for our task, since the ranking it produces does not bias toward any particular node.

The RWR algorithm together with the graph structure represent a form of built-in thesaurus. It is unlike a conventional thesaurus however, in that terms are grouped on the basis of common documents rather than common meanings. This means that in a set of documents concerning graphs, "node" and "edge" are more likely to be linked than "node" and "vertex". The hope is that two terms that are common to a number of documents discuss the same subject.

3.4 Efficiency of the Algorithm

Let $\vec{u}_i$ and $\vec{v}_i$ be *Nx1* probability vectors, we can see that according to RWR, the following equation holds:

$$\vec{u}_i(t) = (1-c)A\vec{u}_i(t-1) + c\vec{v}_i(t).$$

We can show that

$$\vec{u}_i(t) = c(I-(1-c)A)^{-1}\vec{v}_i(t)$$

where *I* is the *N x N* identify matrix.

It can be shown easily that the overall cost of retrieving documents in response to a query is linear on the number of edges, or

T(n) = maxIteration x O(E)

= O(E)

From experiments, the number of iterations to converge (*maxIteration*) is relatively moderate (e.g., approximately 20), or it can be set to have an upper bound (e.g., 100) which is in the order of *O(1)*, thus the cost of retrieval is *O(E)*. Since the matrix is sparse, i.e., a small subset of terms appears in a document as well as the limited *k* number of connections from a term vertex. The algorithm is efficient. In addition, there are literatures on fast large-matrix inversion or fast approximations to solve the equation which makes this approach scalable (Pan et al., 2004).

3.5 Relevance Feedback

Relevance feedback can easily be incorporated into this architecture. In the manual relevance feedback, the user marks documents he considers relevant to his query from the initial/previous result set.

In automatic relevance feedback, top R from the total of M documents returned in the previous run are assumed to be relevant and fed back into the system to recalculate the ranking scores. In this research, Ide_Dec-Hi (Ide, 1971) approach to vector-based relevance feedback is modified, as follows:

$$w_{t_i,q}(new) = \max\left\{0, \min\left\{1, \left(\alpha \cdot w_{t_i,q}(old) + \beta \sum_{\forall d_j \in D_r} w_{t_i,d_j} - \gamma \cdot \max_{nonrelevant}(w_{t_i,d_j})\right)\right\}\right\}$$

where $w_{t_i,q}(new)$ is the new query term weight recalculated from the relevance feedback, $w_{t_i,q}(old)$ is the original/previous query term weight, D_r is the set of documents marked as relevance by users or is the top *k* documents for the automatic method, and α, β, γ are constant. Figure 3 shows an example of the automatic relevance feedback when $R = 2$, i.e., d_2 and d_3 are the top two documents. *Ide_Dec_Hi* is chosen since it is found to give good performance while the calculation is simple.

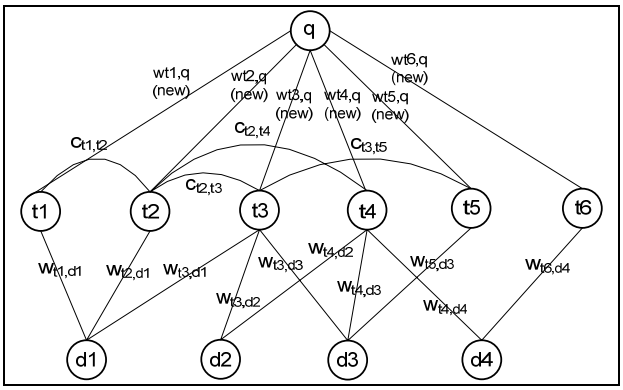

Figure 3. Graph structure when an automatic relevance feedback is incorporated

4. Experimental Evaluation

This section describes experiments carried out to measure the effectiveness of GIR. Four different configurations of the proposed technique are evaluated according to a standard evaluation approach on a standard information retrieval corpus, and the results are measured and analyzed.

4.1 Test Collections

The corpus used for our research is the High Accuracy Retrieval from Documents (HARD) track of TREC 2003 (National Institute of Standards and Technology, 2009) since it fits our objective of the research. The HARD track provides 20 training topics with 100 judged documents for each topic, and uses a set of 48 topics for evaluation. For the evaluation, two judgment results are distributed: the hard

evaluation and the soft evaluation. In the hard evaluation, a document is relevant if not only the document is relevant to the topic but also the matches the metadata of the topic. To aid in this effort, the HARD track provides extra information for systems to utilize in their functions or for human to better form queries. In the soft evaluation, a document is considered relevant when it is relevant to the topic, regardless of the topic's metadata. In this research, the soft evaluation judgment is adopted since we do not use any extra information provided.

4.2 Performance Metrics

Performance metrics employed in this study follow what are used in the HARD track evaluation which consist of: mean average precision (MAP), precision at 10 relevant documents, R-precision, precision-recall graph, and non-interpolated average precision over all relevant documents for each query. These measures are based on two fundamental measures which are recall and precision.

Recall is a measure of the ability of a system to present all relevant documents, defined as:

$$Recall = \frac{number\ of\ relevant\ documents\ retrieved}{number\ of\ relevant\ documents\ in\ collection}.$$

Precision is a measure of the ability of a system to present only relevant documents, defined as:

$$Precision = \frac{number\ of\ relevant\ documents\ retrieved}{number\ of\ documents\ retrieved}.$$

4.3 System Configurations

Four configurations of GIR are studied, as shown in Table 1. The GIR-NoRelNoSig configuration uses binary link weights and does not include term relationship. The GIR-RelNoSig configuration includes term relationships with binary link weights. The GIR-RelSig configuration includes term relationship with real-valued connection weights between 0.0 and 1.0. The GIR-RF configuration is GIR-RelSig with the automatic relevance feedback process as described above. In general, users will examine and thus mark (or rate) only a small number of documents from the first few pages in the results, thus in this study we assume that the top 10 documents from the initial run are automatically fed back into the model. All configurations of GIR are run and compared with the median and maximum performance values from the results of TREC 2003's HARD track.

Table 1. GIR configurations studied in this research

Configuration	Term Relationships	Link Weights	Relevance Feedback
GIR-NoRelNoSig	No	0 or 1	No
GIR-RelNoSig	Yes	0 or 1	No
GIR-RelSig	Yes	Between 0.0 and 1.0	No
GIR-RF (top 10)	Yes	Between 0.0 and 1.0	Yes

4.4 Experiments and Results

After After a document is pre-processed by lexical analysis, stopword elimination according to (Frakes & Baeza-Yates, 1992), stemming by the Porter's algorithm, and indexing. Each GIR configuration is evaluated on the HARD track's corpus.

First, using the GIR-RelSig configuration we experiment to find out how different values of the parameters *c* and *k* affect the performance of the model by varying values of each parameter one at a time and measuring the value of the mean average precision on the training data set.

Pan et al. (Pan et al., 2004) conjecture that what determines a good value for the restart probability is the diameter of the graph. That is, we want our random walker to have a non-trivial chance to reach the outskirts of the whole graph. Thus, if the diameter of the graph is *d*, the probability that he will reach a point on the periphery is proportional to $(1-c)^d$.

For Web graph, the recommended value for *c* is typically 0.15 (Albert et al., 1999). The diameter of the web graph is approximately *d = 19* (Taher Haveliwala & Jeh, 2003) which implies that the probability *p* for the random walker to reach a node in the periphery is roughly: $p = (1-0.15)^{19} = 0.045$. In our architecture, we have a three layered-graph, the diameter is roughly *d = 3*. If we assume the same value of *p* for our model, we have:

$(1-0.15)^{19} = (1-c)^3$

$c \approx 0.65$

We thus initially set the value of c at 0.65, vary the value of k, and measure the mean average precision. The results suggest that the value of k between 15 and 25 yields the best performance, and after the value of k beyond 25 the effectiveness begins to drop. Next, we set the value of k at 20 and vary the value of c. The results suggest that value of c after 0.5 does not give significantly different performance. Therefore, in further experiments we will use the following parameters: $c = 0.6$ and $k = 20$.

Next, all 4 configurations are evaluated according to the HARD track of the TREC 2003 program. Table 2 shows the results for all configurations. We can see that term relationships are helpful in the retrieval as the average precision increases from 0.2518 for GIR-NoRelNoSig to 0.3117 for GIR-RelNoSig (+23.79 %), and term significance captured by weights is also found effective as we can see an increase from 0.3117 for GIR-RelNoSig to 0.3721 for GIR-RelSig (+19.38%).

Table 2. MAP, R-precision, and precision at 10 relevant documents for different configurations

Configuration	**Avg. Prec**	**R-Prec**	**Prec at 10**
GIR-Baseline	0.2518	0.2569	0.4318
GIR-CorrBinary	0.3117	0.2961	0.4946
GIR-Full	0.3721	0.3229	0.5423
GIR-RF	0.3541	0.3112	0.5028
TREC Median	0.2841	0.2994	0.4729
TREC Max	0.4069	0.425	0.65

In terms of ranking ability, both precision at 10 relevant documents and R-precision show similar patterns of relative performance, ordered from the lowest to the highest performance in the following order: GIR-NoRelNoSig, GIR-RelNoSig, and GIR-RelSig.

Compared to the TREC median performance, the GIR-RelSig configuration performs better in all aspects. GIR-RelNoSig outperforms the median in the average precision, but there is no clear winner in

terms of ranking ability. Though GIR-RelNoSig has better precision at 10 relevant documents, it has worse R-precision value than the median. Compared to the TREC maximum performance, none of the configurations studied performs better than it. However, the relative performance of GIR configurations and TREC performance must be used with care since in the TREC maximum performance the best results for different queries may come from different systems. That is, the maximum performance line is not constructed from a single system. In addition, some systems participating in TREC employ extra metadata information or extra human knowledge to assist during the search.

Next, we consider the effect of incorporating automatic relevance feedback function in GIR. To our surprise, the results show that automatic relevance feedback where top 10 documents from the previous search is fed back into system to recalculate similarity scores (GIR-RF) does not improve but instead hurts the performance. This may be due to the fact that using automatic relevance feedback makes the model overly focus on documents surrounding those top documents while the test corpus contains mostly news documents which vary in word selections and styles of writing, thus decreases the recall. However, it is still better than the TREC median performance.

Overall, the experiments show that query expansion through term relationship works well, and taking into account differences in term significances through weights benefits the retrieval.

Figure 4 shows the precision-recall graphs for the GIR configurations. We can see that in the high precision region, performance of different GIR configurations clearly reflect what appear in Table 2. However, in the high recall region, those differences are less clear, for example, GIR-RelSig, GIR-FB, and GIR-RelNoSig nearly overlap. Finally, Figure 5 shows the topic-by-topic comparison of TREC median and maximum to GIR-RelSig.

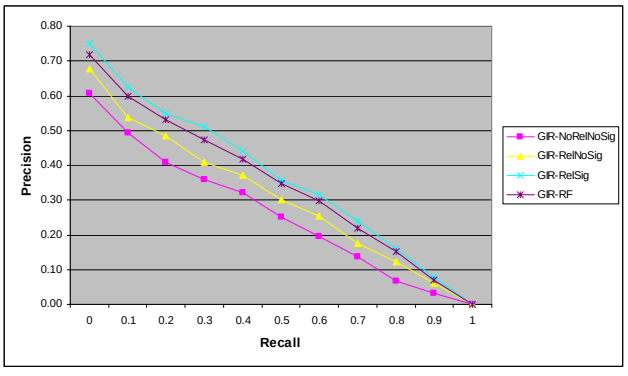

Figure 4. Precision-recall graph for different GIR configurations

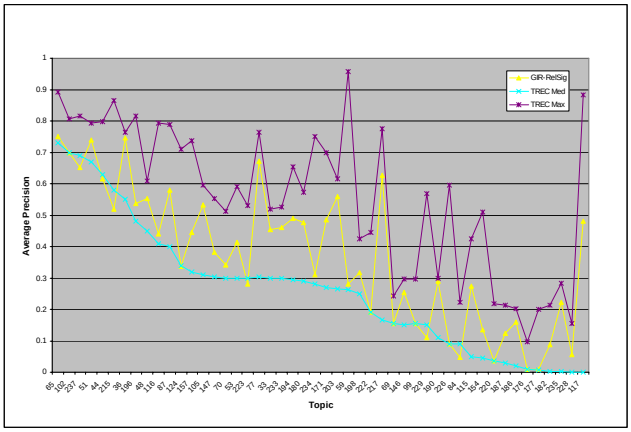

Figure 5. Comparisons of GIR-Full, TREC median, and TREC maximum (topic by topic)

5. Conclusion

In this article, a novel information retrieval model based on graph random walk (GIR) is proposed. The architecture of the model comprises three layers of nodes. In the first layer, each node represents a single document in the collection. In the second layer, a node represents a term extracted from documents in the collection. Term relationship connections are placed between terms with sufficient relationship strengths, calculated from passage-level excerpts. GIR is evaluated according to TREC 2003's HARD track. The results show that including term relationships and term significance weightings are useful for retrieval purpose. GIR is found to perform far better than the TREC median performance, and overall it performs close to the TREC maximum performance. However, incorporating automatic query expansion to the model is found not much helpful. The proposed architecture for information retrieval can be extended in various ways. It can directly be applied to text summarization by instead of ranking excerpts based solely on document similarity, the term layer also is considered, a random walk algorithm is then used to assign important scores for excerpts, and the top n excerpts are extracted as summary. It also can be applied to Web searching by seamlessly adding directed edges between documents to incorporate hyperlinks into the model. The effect of relevance feedback and other performance enhancements should be studied further. In addition, a few techniques for efficient implementation of the random walk with restart algorithm have been proposed (Tong et al., 2006), and their effectiveness for information retrieval should be investigated.

Acknowledgement

This research was supported by Research Center, National Institute of Development Administration, Bangkok, Thailand.

References

A. Albert, H. Jeong, and A.-L. Barabasi. Diameter of the World Wide Web. Nature, 401:130-131, 1999.

R. Baeza-Yates and B. Ribeiro-Neto. Modeling (chapter 2). In R. Baeza-Yates and B. Ribeiro-Neto, editors, Modern Information Retrieval. AWL England, 1999.

R. Blanco and C. Lioma. Random Walk Term Weighting for Information Retrieval. SIGIR'07, Amsterdam, Netherland, July 23-27, 2007.

S. Brin and L. Page. The Anatomy of A Large-Scale Hypertextual Web Search Engine. Computer Networks and ISDN Systems, 1998, 30(1-7).

W. B. Frakes and R. Baeza-Yates. Information Retrieval: Data Structures and Algorithms, Prentice Hall, Englewood Cliffs, NJ, 1992.

G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter, and K. E. Lochbaum. Information Retrieval Using a Singular Value Decomposition Model of Latent Semantic Structure. In Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 465-480, Grenoble France, 1988.

S. Hassan, R. Mihalcea, and C. Banea. Random Walk Term Weighting for Improved Text Classification. In Proceedings of TextGraps: 2nd Workshop on Graph Based Methods for Natural Language Processing, ACL, 2006, 53-60.

E. Ide. New Experiments in Relevance Feedback. The SMART Retrieval System, pages 337-354. Prentice-Hall, 1971.

M. R. Islam, B. D. Sarker, and M. R. Islam. An Effective Term Weighting Method Using Random Walk Model for Information Retrieval. In Proceedings of the International Conference on Computer and Communication Engineering, Malaysia, 2008.

H. Kaji, Y. Morimoto, T.Aizono, and N. Yamasaki. Corpus-Dependent Association Thesauri for Information Retrieval. In Proceedings of the 18th Conference on Computational Linguistics, Germany, 2000.

G. Kasai, M. Lalmas, and T. Roelleke. Focused Structured Document Retrieval. In Proceedings of the 9th International Symposium on String Processing and Information Retrieval, 2000.

J. M. Kleinberg. Authoritative Sources in A Hyperlinked Environment. Journal of the ACM, 1999, 46(5): 604-632.

R. Mihalcea and P. Tarau. TextRank: Bringing Order into Texts. In Proceedings of Empirical Methods in Natural Language Processing. ACL, 2006, 404-411.

M. Montes-y-Gomez, A. López-López, and A. Gelbukh. Information retrieval with conceptual graph matching. In the Proceedings of the 12th International Conference of Database and Expert Systems Applications, 2001.

National Institute of Standards and Technology. Text REtrieval Conference. http://trec.nist.gov

Y. Ogawa, T. Morita, and K. Kobayashi. A Fuzzy Document Retrieval System Using the Keyword Connection Matrix and a Learning Method. Fuzzy Sets and Systems, 39:163-179, 1991.

P. Ogilvie and J. Callan. Language Models and Structured Document Retrieval. In Proceedings of the Initiative for the Evaluation of XML Retrieval Workshop, 2002.

J. Y. Pan, H. J. Yang, C. Faloutsos, and P. Duygulu. GCap: Graph-based Automatic Image Captioning. In Proceedings of the 4th International Workshop on Multimedia Data and Document Engineering (MDDE'04), Washington, DC, USA, 2004.

M. F. Porter. An Algorithm for Suffix Stripping. Program, 14(30) pp.130-137, 1980.

Y. Quintana, M. Kamel, and A. Lo. Graph-Based Retrieval of Information in Hypertext Systems. In Proceedings of the 10th annual international conference on Systems documentation, ACM Press, 1992, pp.157-168.

S. E. Robertson and K. Sparck Jones. Relevance Weighting of Search Terms. Journal of the American Society for Information Sciences, 27(3):129-146, 1976.

G. Salton. The SMART Retrieval System: Experiments in Automatic Document Processing. Prentice Hall Inc., Englewood Cliffs, NJ, 1971.

G. Salton and C. Buckley. Term-Weighting Approaches in Automatic Retrieval. Information Processing & Management, 24(5):513-523, 1988.

S. K. Taher Haveliwala and G. Jeh. An analytical Comparison of Approaches to Personalizing Pagerank. Technical report, Stanford University, 2003.

D. Thammasut and O. Sornil. A Graph-Based Information Retrieval System. In Proceedings of International Symposium on Communications and Information Technologies, 2006, pp. 743-748.

R. M. Tong, L. A. Appelbaum, V. N. Askman, and J. F. Cunningham. Conceptual Information Retrieval Using RUBRIC. In Proceedings of ACM Conference on Research and Development in Information Retrieval, 1987, pp.247-253.

H. Tong, C. Faloutsos, and J.-Y. Pan. Fast Random Walk with Restart and Its Applications. In Proceedings of International Conference of Data Mining, 2006.

H. Turtle and W. B. Croft. Evaluation of an Inference Network-Based Retrieval Model. ACM Transactions on Information Systems, 9(3):187-222, July 1991.

R. Wilkinson and P. Hingston. Using the Cosine Measure in a Neural Network for document retrieval. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, pages 202-210, Chicago, IL USA, October 1991.

C. C. Yang and K. Y. Chan. Retrieving Multimedia Web Objects Based on PageRank Algorithm. In Proceedings of the 14th