Supporting Data Structures Courses Using a Portfolio Search Engine
利用學習歷程搜尋引擎支援資料結構課程

Ping-Hsing Don, Chen-Chung Liu

Department of Computer Science and Engineering, Nanya Institute of Technology

Graduate Institute of Network Learning Technology, National Central University

董炳信*　劉晨鐘**

*南亞技術學院資工系 副教授 **國立中央大學網學所 副教授

Abstract

Data Structures is a foundational course for undergraduate computer science and engineering students, requiring students to learn about various data types and their implementation. Teachers generally give programming assignments to students. However, it is critical to these novices that successfully learn concepts and programming skills, when they only study examples in the teaching materials. Therefore, students usually input keywords in content-based search engines to retrieve documents (or web pages) including the keywords for reference. Although these search engines are convenient, they do not generally produce ideal query results for students. Portfolios of previous students can help new students taking the same course. However, these portfolios are generally discarded at the end of the course. This study uses the portfolios of past Data Structures students, personal construct technique and Bayesian Belief Network mechanism to construct a portfolio search engine. Query results generated in this study indicate that the portfolio search engine always produces significantly better precision and recall than the content-based search engine.

摘要

資料結構是計算機科學與工程大學生的基礎課程，它要求學生了解各種資料型態和它們的執行情況。教師通常會給學生一些程式作業，但是這些新手若只研究教材中的例子是無法成功的學習資料結構的概念和程式技巧。因此，學生通常會利用內容搜尋引擎輸入一些關鍵詞去檢索相關文件（或網頁）來參考。雖然這些搜尋引擎很方便，但是通常不會產生理想的查詢結果。以前學生的學習歷程可以幫助修相同的課的新學生，但是

這些學習歷程在課程結束後一般都會被丟棄。本研究利用以前學生修資料結構課的學習歷程，個人建構技術和貝氏信念網絡機制，去構建一個學習歷程搜尋引擎。在本研究下的查詢結果顯示這個搜尋引擎總是比內容搜索引擎產生很高的查準率和查全率。

Keywords: Bayesian Belief Networks、 Data Structures、 personal construct、 search engine
關鍵詞：貝氏信念網絡、資料結構、個人構念、搜尋引擎

I. Introduction

Data Structures is a foundational course for undergraduate computer science and engineering students, requiring students to learn about various data types and their implementation. Teachers adopt examples to illuminate how and why data structures are utilized, and then give programming assignments to students, to motivating them to solve a problem by different approaches, in terms of their running time, space usage and implementation complexity. However, it is critical to these novices that successfully learn concepts and programming skills relating to abstract data structures, when they only study examples in the teaching materials. Some students fail their Data Structures courses because they do not complete their programming assignments, or complete them in a substandard fashion (Lawrence, 2004).

To complete their assignments successfully, students require relevant information to improve their cognitions and programming skills, and therefore generally input keywords in content-based search engines to retrieve documents (or web pages) including the keywords for reference. These search engines are easy and convenient for students to use, and save students time. However, search engine queries frequently have ambiguous or semantic keyword problems (Abhishek & Hosanagar, 2007). These search engines can give the better query results when adopting complex queries to solve the keyword problems. However, users cannot reasonably be expected to be familiar with the correct syntax, tags and delimiters supported by search engines (Adali et al., 1997). Up to 70% of users only adopt one query keyword (Butler,2000). Therefore, various methodologies have been developed for building a thesaurus (Miller, 1995; Han et al., 1998; Tseng, 2001). However, 71% of users felt significant frustration while searching (Wyle, 2001), because some information have implicit or deeper concepts which are not indicated by keywords. Keywords can not precisely reveal the concepts in documents (Ghazfan et al.,1995), thus the precision and recall of the retrieved information is low (Zamir & Etzioni , 1999). Therefore, the queries still do not produce the results required by students.

Because portfolios contain learning works as well as the processes to develop these works (Burch, 1999), these portfolios with valuable learning experiences can enable peers to consider problems carefully and enhance his learning ability (Lawrence, 2004; Liu & Tsai, 2005; Moster & Sudzina, 1996; Borko et al., 1997; Duffy et al., 1999). These portfolios also can help the following students take the same course after days (Hogan & Pressley, 1997). However, these portfolios are generally discarded at the end of courses. This study reuses the portfolios in the past Data Structures course and constructs a portfolio search engine to improve the performance of content-based search engines. The engine substitutes personal constructs (Kelly, 1995) for keywords in students' works, and also adopts the reasoning ability of the Bayesian Belief Network (BBN) (Cheeseman & Stutz, 1996) mechanism to expand the concept of the query keyword while searching. The query results in this study indicate that the proposed portfolio search engine always produces significantly better precision and recall than the content-based search engine when students use only one query keyword.

The remainder of this study is structured as follows. Section II introduces background information about the database source of search engines. Section III describes the portfolio search engine in detail. Section IV presents the experiments and results. Section V draws conclusions.

II. Portfolios in Peer Assessment System

54 students submitted 54 works for the same programming assignment to the Peer Assessment System (PAS) (Liu & Tsai, 2005) on web in our past Data Structure course. Each student's work included program code and a short summary of the code using MS Word or PDF. These 54 students were randomly divided into eight assessment groups, each with six or seven students. Each student assessed his group works using the repertory grid techniques (Shaw & Gaines, 1995) in PAS. The repertory grid technique is a reflective device for raising self-awareness and encouraging understanding of the perspectives of others (Pope & Keen, 1981). PAS successfully elicited personal concepts (constructs in Kelly's terminology) concerning data structures from assessing students' works. Each repertory grid in PAS included elicited concepts, both explicit and implicit, and assessed scores on these concepts in each work (Liu & Tsai, 2005). This study reused these portfolios in our past Data Structure course as the database source of the portfolio search engine. The portfolios including elicited constructs, assessed scores on elicited constructs of 54 repertory grids and 54 students' works, were accumulated to PAS at the end of the course.

III. Modules in Portfolio Search Engine

Students must judge the suitability of alternative implementations of data structures for their assignments. Because the 54 works in PAS have different program codes with concepts and programming skills, later students can use them to understand their own knowledge and support their assignments. However, content-based search engines adopt keywords in works (or documents) to represent characteristics of works. The keywords in works are explicit concepts. Some works, especially program codes, have implicit or deeper concepts that are not shown by keywords. These implicit concepts also can be adopted to represent the characteristics of works. The elicited concepts (or constructs) in PAS can be substituted for keywords in works. Besides, the Bayesian Belief Network (BBN) can use analysis and reasoning ability to expand the construct of query keyword. Therefore, this study can adopt personal constructs and BBN mechanism to solve the keyword problems while searching. Figure 1 shows the system flowchart of the portfolio search engine. The engine comprises three main modules, namely the construct causality, document index and query keyword process modules, as depicted in the dotted block diagrams.

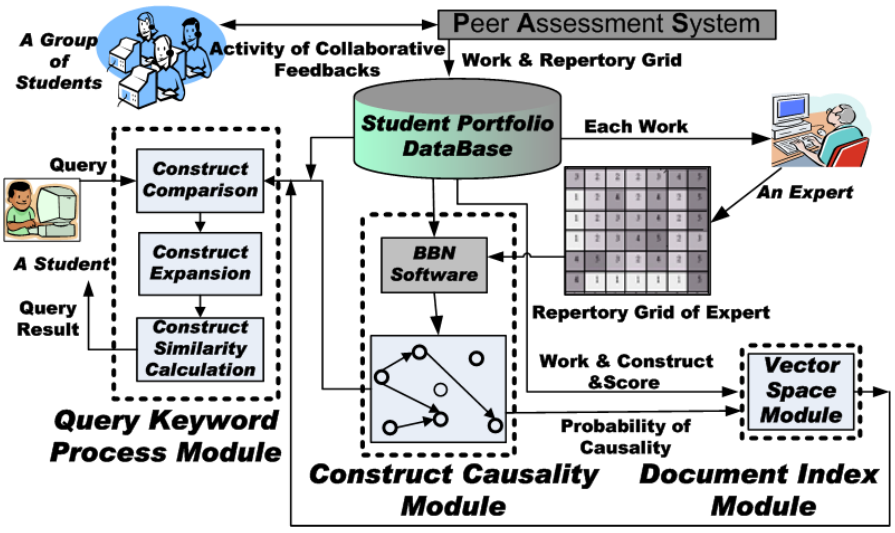

Figure 1 System flowchart of portfolio search engine

A. Construct Causality Module

A student wishing to search works with a specific construct (or concept) for reference generally inputs an appropriate keyword in a content-based search engine. If no work in the database has the keyword, then the search engines do not retrieve any work, even though some works have constructs with similar meanings to the construct of the query keyword. For instance, when a student inputs the keyword *Good Algorithm*, the search engine does not retrieve works with the construct *Execution Fast*. However, *Good Algorithm* and *Execution Fast* not only relate to similar data structure properties, but are also related. If a search engine has the causality

network of the construct *Good Algorithm*, then it can adopt the causality to retrieve these works with the construct *Execution Fast*. Therefore, a construct causality network can reduce the query keyword problems, and thus improve the query results.

BBN has better precision and recall than other information retrieval systems (Ghazfan et al., 1995). BBN can handle uncertainty in information retrieval, support the relationship in domain knowledge and allow learning of construct causality by Bayesian statistics. Therefore, the analysis and reasoning ability of BBN can be adopted to build the construct causality network. BBN can adopt the constructs and assessed scores on constructs as variables to reason the causality and probability of constructs in each student's work. However, the construct causality networks of all students are independent. The repertory grid of expert can be used to build the relationship among construct causality networks in all students' works. Therefore, a teacher assessed each student's work by six fundamental constructs such as *Binary Search*, *Dynamic Memory* and *User Interface* to obtain a repertory grid of expert. Figure 2 shows the total construct causality network resulting from inputting the repertory grids of all students and an expert into the BBN mechanism.

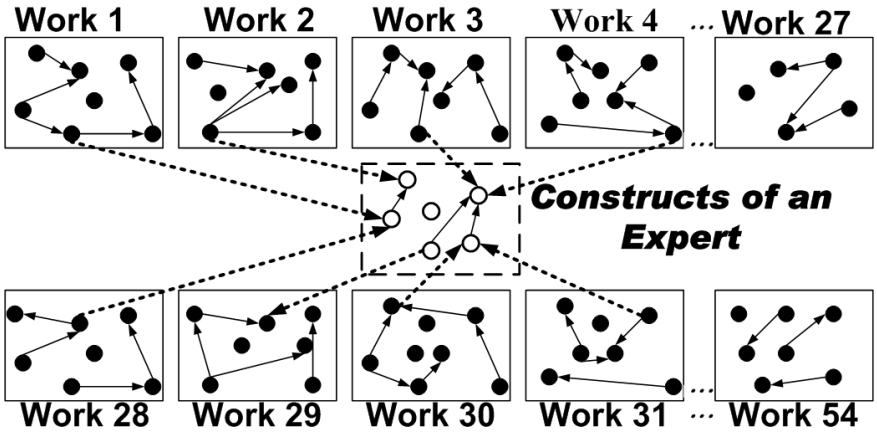

Figure 2 Total construct causality network

The dotted block diagram in Fig. 2 denotes the construct causality of an expert, and each fixed block diagram denotes the construct causality in each student's work. The dotted arrows lines created by the BBN mechanism denote the dependent relations among an expert's constructs and each work's constructs. This study adopted the construct causality module to support the following document index and query keyword process modules for analysis and reasoning.

B. Document Index Module

Content-based search engines extract all keywords from each document (or page), and then build the index for every keyword. The vector space model (Salton, 1971) represents each document by index terms ,and each index term may be a word or phrase for understanding the content of documents (Baeza-Yates & Ribeiro-Neto, 1999). The vector space model can be adopted to process each student's work, because each index term can denote each construct in students' works. TF-IDF (Salton & McGill, 1983) is a popular calculating method for weighting words for text mining. Each word weight in a document is calculated from two factors, TF (Term Frequency) and IDF (Inverted Document Frequency). Each assessed score (1–5) on each elicited construct in PAS can denote the strength of the construct in each work. Therefore, this study adopted the assessed score on construct to denote TF as the word (or construct) weight in each work. When the query keyword and each work are presented by the vector form, the included angle between them represents their similarity. A larger angle denotes a larger difference between the query keyword and the work.

274 personal constructs concerning data structures were elicited from the 54 students' works in PAS (Liu & Tsai, 2005). Therefore, each work was transferred to 274 dimensions in document vector given by d_j : <s_1, s_2, ..…, s_i, …..., s_{274}>, where d_j denotes the document vector of work j; s_i denotes the assessed score on construct in document vector d_j and the status of construct i. Because a work of a student only was assessed by the group members of the student with group elicited constructs, the assessed scores on elicited constructs (or index terms) in document vector of the work was assigned by the repertory grid of the student. However, the work was not assessed by other groups' members. Therefore, the assessed score of 3 was assigned to other elicited constructs (or index terms) in document vector, representing these constructs that did not appear in the work and can not represent the status of the constructs.

To effectively discriminate whether a construct is representative of data structures in works, original assessed scores (1-5) were revised and given in the range shown in Table 1. A positive score indicates that a construct is well represented in a work. A negative score indicates that a construct is poorly represented in a work. For instance, the document vector d_A of work A is <2,1,1,2,1,−2,−1,0,0,0,0,0,0,…..>. The first seven index items in document vector d_A denote the constructs in work A. The scores of first seven index items in document vector d_A denote the strength of these constructs in work A. However, the other constructs are not show in work A, because work A is not assessed by other groups' members with their elicited constructs. Therefore, the scores of other index terms in document vector are zero, representing these constructs that did not appear in work A.

Table 1 Contrast table of revised scores

Original Score	5	4	3	2	1
Revised Score	2	1	0	-1	-2

The BBN mechanism then adopts the construct causality module to expand the document vector d_A of work *A*. For instance, in Fig.3, work *A* has an assessed score of 2 on the index term *Bubble Sort*, because it has the construct *Bubble Sort*. Because the expert's *Binary Search* is related to the construct *Bubble Sort* in work *A*, BBN can reason that the scores of *Execution Fast* and *Good Comparative Skill* constructs in work *B* and *C* are 2, because they have highest probabilities (80% and 70%) in the construct causality network. If the fourth index term of the document vector d_A of work *A* denotes the construct *Bubble Sort*, then the ninth index term denotes the construct *Execution Fas*t, and the eleven index term denotes the construct *Good Comparative Skill*. Therefore, the document vector d_A of work *A* is expanded to <2,1,1,2,1,−2,−1,0,**2**,0,**2**,0,0, …..>. The portfolio search engine adopts the construct causality module and document index module for the following operations in the query keyword process module.

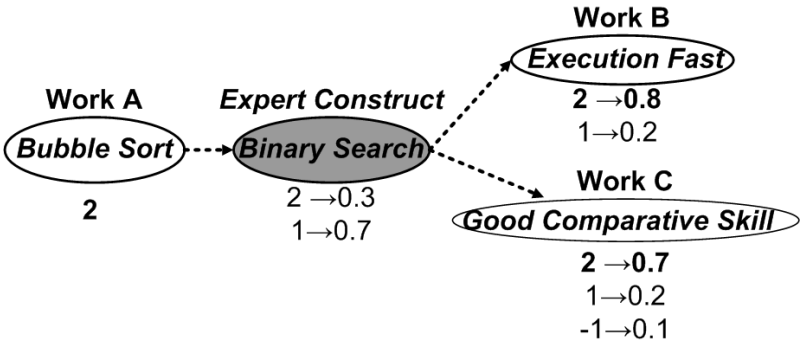

Figure 3 Construct causality and probability

C. Query Keyword Process Module

When a portfolio search engine receives a query keyword, the query keyword process module first compares the query keyword with all constructs of works in the database. If a conforming construct is found in students' works, then the query keyword is processed as the query vector by the conformed construct. The query keyword process module then calculates the similarity between the query vector and the document vector of each work. Students' works is then displayed in order of similarity. As revealed in Fig. 1, this module comprises three operational modules as follows:

1.Construct Comparison Module

This module first compares the query keyword with all constructs of works in the database. If no constructs found in the database conform to the query keyword, then this module returns a 'fail' message to the

student and ends the search. Therefore, the search engines do not retrieve any work. By contrast, if a conforming construct is found in the database, then the query keyword is represented by the conformed construct, and is formed as the query vector. For instance, when a student inputs the keyword *Bubble Sorting*, then the search engine compares the keyword *Bubble Sorting* with all constructs in the database. If a work contains a conforming construct such as *Bubble Sort*, then the query vector is formed as <2>, because each index term in the query vector representing each conforming construct is set to 2. The query vector is then expanded by the following operations.

2.Construct Expansion Module

The BBN expands constructs in other works which are connected with the conformed construct to the query vector. These connected constructs were assigns fitting scores to the index terms in the query vector, which depend on these connected constructs with the highest probability in the construct causality network. For instance, Fig. 3, shows a conforming construct *Bubble Sort* in work *A*, producing a query vector of <2>. BBN then adopts the causality network of construct *Bubble Sort* to reason other connected constructs such as *Execution Fast* and *Good Comparative Skill* in work *B* and *C*, and expands them to the query vector. These connected constructs (or index terms) in the query vector are assigned the value 2, because they have the highest probability (80% and 70%). Therefore, the query vector <2> is expanded as <2, 2, 2>, and is then compared with all document vectors of works in the database for similarity by the following construct similarity calculation module.

3.Construct Similarity Calculation Module

The portfolio search engine only retrieves students' works with the same index terms (or constructs) as the query vector when calculating the construct similarity according to their cosine of the query vector and each document vector (Salton & Lesk,1968). The following formula denotes the construct similarity calculation.

$$sim(d_j, q) = \frac{d_j \times q}{|d_j| \times |q|}$$

Where d_j denotes document vector of work *j*; *q* denotes the query vector. For instance, if two works have the same constructs (or index terms) as the query vector *q* <2, 2, 2>, then one document vector is <…., 2, …., 2,…., 2, …….>, and another is <…., 1, …., 1,…., 1, …….>. These two document vectors are transferred as <2, 2, 2> and <1, 1, 1> to calculate the construct similarity to the query vector. However, the above two document vectors have the same similarity of 1 (100%) when they are compared with query vector *q* <2, 2, 2>. However, the document vector with index value 2 is more similar to the query vector than is the document vector with index value 1. Therefore, the construct similarity calculation formula multiplies the length of the document

vector d_j, which is called *construct strength* ($|d_j|$), then the similarities of the above two document vectors to the query vector q <2, 2, 2> can be separated. After calculating the similarity of each document vector to the query vector, the system returns a set of students' works with similarity in the order from high to low.

IV. Experiment and Discussion

The subjects of this study were 48 undergraduate students of the Department of Computer Engineering and Science at Yuan Ze University in Taiwan enrolled in the Data Structures course. The teacher gave a programming assignment to these students during the course. The assignment was different from the assignment in the past semester, but required many of the same concepts and programming skills. To compare the performance of different search engines, content-based (CB), construct-based (CI) and portfolio (BB) search engines were adopted in this experiment. CB is based on keyword indexing, CI is based on construct indexing, and BB is based on construct indexing and a Bayesian Belief Network (BBN) mechanism. These search engines adopted the accumulated portfolios in PAS as the database source. Each student was asked to input one query keyword concerning data structures in these search engines while searching. Two different findings were obtained from this experiment.

CI performed better than that of CB, but BB performed the best, as revealed in Fig. 4. For instance, when a student inputs the keyword *log2n*, CB did not retrieve any work, because no work contained the keyword *log2n*. CI could retrieve some works, because some works contained conforming constructs such as *Execution time* ***log2n***. Why BB could retrieve more works than CI? In Fig. 5, the conforming construct *Execution time* ***log2n*** has a relationship with the construct *Binary Search* of the expert in the causality network of construct *log2n*. Hence, BB could retrieve works containing the *Searching Skill* and *Execution Speed* constructs by the BBN mechanism. Thus, BB performed the best.

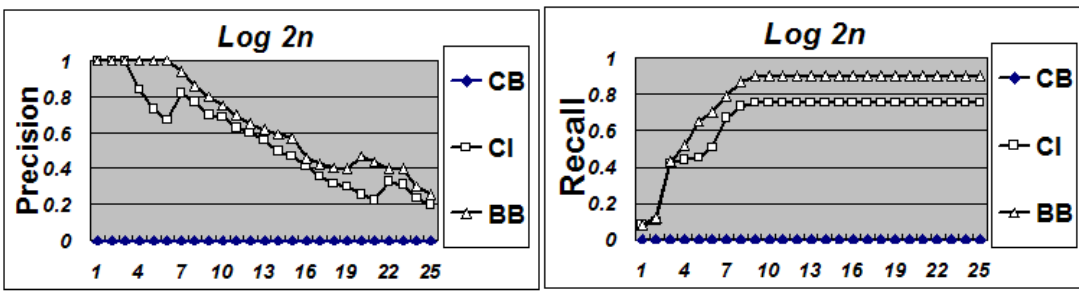

Figure 4 Precision and recall of keyword *log2n*

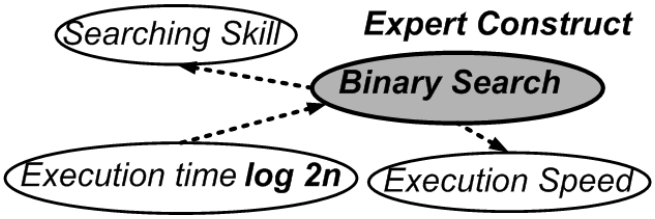

Figure 5 Causality network of construct *log2n*

CI performed the best, followed by BB, as revealed in Fig. 6. For instance, when a student inputs the keyword *Dynamic Memory*, CB retrieved works containing the keyword *Dynamic Memory*. CI performed better than CB, because more works in the database included the construct *Dynamic Memory*. Why was the performance of BB not the best? As depicted in Fig. 7, although the construct *Correct* ***Dynamic Memory*** *Location* in a work connected with the expert's construct *Dynamic Memory*, it also connected with the expert's construct *User Interface* in the causality network of construct *Dynamic Memory*. However, the expert's *Dynamic Memory* and *User Interface* constructs are unrelated. Therefore, incorrect connections to the construct *User Interface* in the causality network of construct *Dynamic Memory* led to incorrectly expanded constructs such as the construct *Exception Process* in other works, reducing the performance of BB.

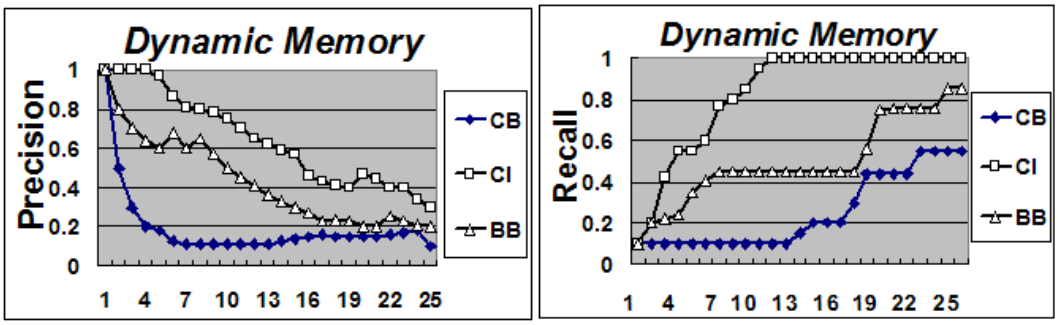

Figure 6 Precision and recall of keyword *Dynamic Memory*

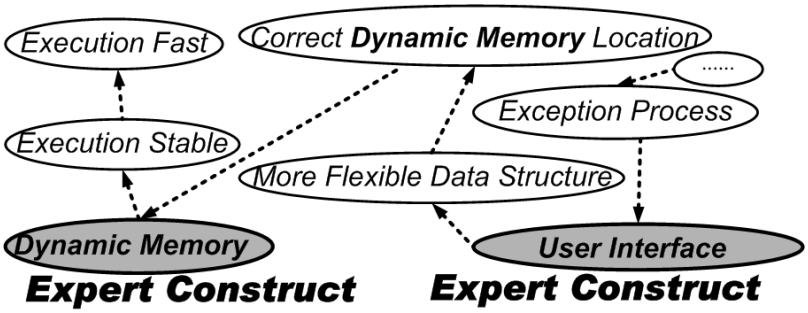

Figure 7 Causality network of construct *Dynamic Memory*

A hybrid index search engine (HI) was adopted to improve the performance of the portfolio search engine (BB). The HI reordered the works in descending order from the sum of the ranking of answer works retrieved by CI and BB. The following formula denotes the similarity of each work to the query keyword in HI.

$$d_{HI,q} = d_{CI,q} + k \times d_{BB,q}$$

Where $d_{HI,q}$ denotes the similarity of document vector d of a work with query vector q in HI; $d_{CI,q}$ denotes the similarity of document vector d of a work with query vector q in CI; $d_{BB,q}$ denotes the similarity of document vector d of a work with query vector q in BB; k denotes the weight of BB. Table 2 shows the average precision and recall of HI with various k values.

Table 2 Average performance with various k values

K value	1	0.5	0.4	0.3	0.2	0.1
Precision	0.57789	0.596888	0.603789	0.611502	0.635721	0.651859
Recall	0.412074	0.426354	0.432997	0.438941	0.455027	0.466621

As revealed in table 2, a small k can lead to a good overall precision and recall. Therefore, k = 0.1 is selected for hybrid index search engine (HI). The following figure 8 and 9 separately show the total average performances of precision and recall through each of the four search engines, again obtained by inputting the 48 query keywords of students to search engines. The new portfolio search engine (or HI) always had the best performance in this experiment.

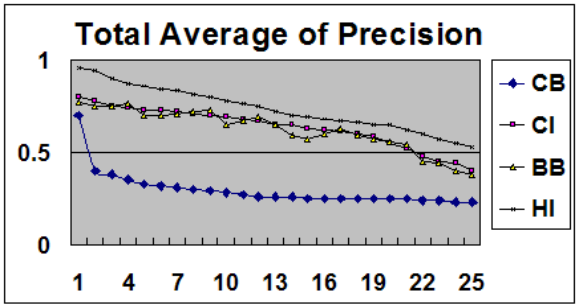

Figure 8 Total average performance of precision

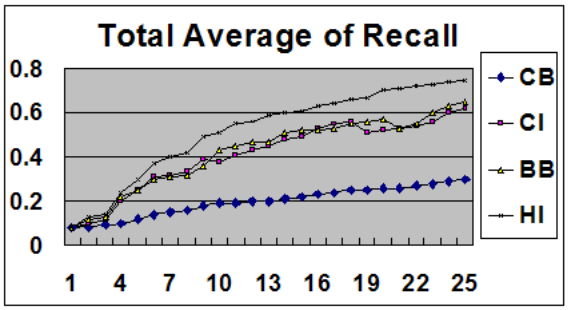

Figure 9 Total average performance of recall

V. Conclusion

Content-based search engines are limited in retrieving works, because implicit or deeper concepts in some works are not shown by keywords. This study found that portfolio search engines performed significantly better than content-based search engines, and could be applied in any computer science course. Teachers can also adopt the Bayesian Belief Network mechanism to observe construct causality in students' works to understand the defects in their personal constructs concerning data structures, and use this information to give each student adaptive guidance and adjust their teaching materials. Therefore, this study not only could help students learn by themselves, but also improve teaching processes to support the Data Structures course.

Reference

Abhishek, V. and Hosanagar, K. (2007), "Keyword generation for search engine advertising using semantic similarity between terms," *Proceedings of the ninth international conference on Electronic commerce (ICEC'07),* Minneapolis, MN, USA, ACM: 89-94.

Adali S., Bufi C. and Temtanapat, Y. (1997), "Integrated Search Engine," pp.140, 1997 IEEE Knowledge and Data Engineering Exchange Workshop (KDEX '97).

Baeza-Yates, R. and Ribeiro-Neto, B. (1999), *Modern Information Retrieval.* New York: The ACM Press.

Borko, H., Michalec, P., Timmons, M. and Siddle, J. (1997), "Student teaching portfolios: A tool for promoting reflective practice," *Journal of Teacher Education*, vol. 48, no. 5, pp. 345-357.

Burch, C. B. (1999), "Inside the portfolio experience: The student perspective," *English education*, vol. 32, no. 1, pp. 34-49.

Butler, D. (2000), "Souped-Up Search Engines," *Nature*, vol. 405, pp. 112-115.

Cheeseman, P. and Stutz, J. (1996), "Bayesian Classification (AutoClass): Theory and Results," in *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, pp. 61.

Duffy, M. L., Jones, J. and Thomas, S. W. (1999), "Using Portfolios to Foster Independent Thinking," *Intervention in School and Clinic*, vol. 35, no. 1, pp. 34-37.

Ghazfan, D., Srinivasan, B. and Indrawan, M. (1995), "Semantically Correct Bayesian Network based Information Retrieval," *Proceeding of the 5th Hellenic Conference in Informatics*, pp. 639-648.

Han, J. J., Choi, J. H., Park, J. J., Yang, J. D. and Lee, J. K. (1998), "An object-based information retrieval model: toward the structural construction of thesauri," *IEEE international Forum on Research and Technology Advances in Digital Library*, pp. 117-125.

Hogan, K. and Pressley, M. (1997), *Scaffolding Student Learning: Instructional Approaches and Issues*. MA: Brookline Books, pp. 7-11.

Kelly, G. A. (1995), *The Psychology of Personal Constructs*. New York: Norton.

Lawrence, R. (2004), "Teaching Data Structures Using Competitive Games," *IEEE Transactions on Education*, vol. 47, no. 4, pp. 459-466.

Liu, C. C. and Tsai, C. M. (2005), "Peer assessment through web-based knowledge acquisition: tool to support conceptual awareness," *Innovation in Education and Teaching International*, vol. 42, no. 1, pp. 45-61.

Miller, G. A. (1995), "WordNet: a lexical database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39-41.

Moster, M. P. and Sudzina, M. R. (1996), "Undergraduate case method teaching: Pedagogical assumption vs. the real word," *Presented at the annual meeting of the Association of Teacher Educators*, ST. Louise, MO. (ERIC document Recast Service, No, ED 395900)

Pope, M. L. and Keen, T. R. (1981), *Personal construct psychology and education*. London: Academic Press.

Shaw, M. L. G. and Gaines, B. R. (1995), "Comparing constructions through the web," *Computer Support for Collaborative learning*, pp. 300-307.

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

Salton, G. and Lesk, M.E. (1968), "Computer evaluation of indexing and text processing," *Journal of the ACM*, vol. 15, no. 1, pp. 8-36.

Salton, G. and McGill, M. J. (1983), *Introduction to Modern Retrieval.* McGraw-Hill Book Company.

Tseng, Y. H. (2001), "Fast co-occurrence thesaurus construction for Chinese news," *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 2, pp. 853-858.

Wyle, M. F. (2001), Why search when you can find. Available:http://www.euphorion.com/whitepapers/why-Search-Find.pdf

Zamir, O. and Etzioni, O. (1999), "Grouper: A Dynamic Clustering Interface to Web Search Results," in *WWW8 Proceedings*, pp. 1361-1374, Toronto.