Merging Results From Isolated Search Engines

Nick Craswell, David Hawking and Paul Thistlewaite

Co-operative Research Centre For Advanced Computational Systems
Department Of Computer Science
Australian National University
{nick,pbt}@cs.anu.edu.au, david.hawking@cmis.csiro.au

The authors wish to acknowledge that this work was carried out within the Cooperative Research Centre for Advanced Computational Systems established under the Australian Government's Cooperative Research Centres Program.

Abstract. Two new techniques for merging search results are introduced: Feature Distance ranking algorithms and Reference Statistics. These techniques are compared with other published methods, using TREC effectiveness evaluations based on human relevance judgements and input rankings from 5 different search engines over 5 disjoint document collections. The new techniques are found to be more effective than existing methods in an isolated-server environment such as the World Wide Web. In addition, Feature Distance algorithms are found to be as effective in an isolated-server environment using Reference Statistics as they are in an integrated-server environment.

1 Introduction

The problem addressed in this paper is as follows:

> *A document ranking $R = \langle D, o \rangle$ consists of a set of documents D and an ordering o. Given N rankings $R_1 \ldots R_N$, generate a single ranking $R_m = \langle D_m, o_m \rangle$ such that $D_m = D_1 \cup \ldots \cup D_N$ and o_m is an effective ranking, meaning that it tends to rank relevant documents above irrelevant ones.*

Typically, the N incoming lists are search results from document sets $C_1 \ldots C_N$. Instead of creating a single central index of all documents $\bigcup C_i$, a merging strategy generates R_m using information collated at query time pertaining just to the current query and documents D_m.

In this paper, a merging strategy based on downloading document contents and using a set of reference collection statistics is introduced, along with a new ranking algorithm designed for use in merging. The new techniques are evaluated in terms of retrieval effectiveness using a TREC [Voorhees and Harman, 1997] framework, and compared to other merging methods. Efficiency questions are not considered here.

2 Background

2.1 Merging Architecture

This section describes the networked information retrieval architecture considered in this paper. It is assumed that a number of search servers (search engines) are available, each of which indexes documents from one or more document servers. A basic search is an interaction between a the user's client software (e.g. a web browser) and a search server, where the client makes a request in the form of a query (q) and the search server responds with a ranked list of documents (R). The user may then view documents by downloading them from the appropriate document servers.

In an environment where a large number of search servers are available, it is possible to employ a special client known as a *metasearcher* [Lawrence and Giles, 1998, Gauch et al., 1996, Selberg and Etzioni, 1995, Dreilinger and Howe, 1997, Smeaton and Crimmins, 1996]. Metasearchers merge results from multiple search servers into a single ranked list using some *results merging* strategy. In addition, some form of *query translation* technology is necessary, to interact with different search servers, and some *server selection* method may be available for locating servers covering documents relevant to the user's query. The problems of query translation and server selection are not considered in this paper. However, a simple method of server selection is used here to measure "server promise" (see Section 4.2).

Merging is particularly useful when no single index covers all information of interest to the user. For example, a user might wish to find information on a medical treatment by searching a publicly available medical Web site, a subscription only Web site, a proprietary database on the local intranet, and a local CD-ROM. If it is a one-off search, building a single index of all documents is clearly impractical. However, using a metasearcher, the impression of a unified index can be given.

Merging strategies can be divided into two categories [Voorhees, 1995], integrated methods and isolated methods. Integrated methods require the servers to provide special information for use in merging, while isolated merging methods can be applied without any specialised merging information from servers.

2.2 Integrated Merging Methods

In ordinary retrieval from a unified collection, the most effective ranking algorithms [Harman, 1992] require collection statistics, particularly document frequencies. A document frequency — the number of documents containing at least one occurrence of a given term — can be used to weight the relative importance of query terms, or to estimate relative probabilities of document relevance. Integrated merging strategies use special protocols and server functionality to collate collection statistics, allowing comparable document scores to be generated.

One approach is to collate collection statistics at the search servers [Viles and French, 1995] or at the client [Mazur, 1994, Krester et al., 1998, Callan et al., 1995]. In the latter case, the client provides statistics to the servers with each query. In both cases, the servers use these collection statistics, along with a homogeneous ranking algorithm, to generate comparable document scores. The client then generates the merged ranking by sorting the documents in descending order of score. The advantage of this approach is that it allows servers to generate comparable document scores. The disadvantages are that: 1) the servers must all comply with some statistic propagation protocol, 2) non-trivial communication must take place before query time, and 3) the set of servers whose statistics are included must be decided before query time.

A different approach requires each server to supply collection information with its search results [Walczuch et al., 1994, Gravano et al., 1997, Kirsch, 1997]. The client can then combine collection information from the individual servers into overall collection information, and employ some ranking algorithm to generate the merged list. Document information for use in ranking may be provided along with the collection statistics, or obtained through downloading the documents from document servers. This approach requires no pre-query synchronisation, but may only be applied when servers support the necessary statistic communication protocol.

The above integrated approaches use different definitions of "the collection": 1) the documents indexed by a fixed set of servers, and 2) the documents indexed by the servers currently being searched. In either case, if the same document is covered by more than one search server it will be counted more than once in the collection statistics. In both cases, efficiency and breadth of application are reduced by the requirement for extra communication and server functionality. The gain in effectiveness enabled by the additional information is examined in Table 2.

2.3 Isolated Merging Methods

Isolated merging methods use information which is readily available from search servers and document servers, without requiring any special server functionality. Four sources of merging information are examined here:

1. The ordinal rank assigned to a document by a search server,
2. The score assigned to a document by a search server,

3. A server promise metric, provided by some server-selection method (such as the one described in [Hawking and Thistlewaite, 1998]), and
4. The contents of the document itself, downloaded by the client from a document server.

All the isolated merging methods described in this section have been implemented (or approximated) and are evaluated in the present study.

In *score based* merging methods, documents are ranked in order of server-assigned scores, or some transformation of those scores. The raw scores produced by servers using heterogeneous ranking algorithms and non-shared collection statistics are not comparable, but they are included in the evaluation for completeness. To make the scores more comparable, they can be scaled so that the scores from each server range between two set values [Selberg and Etzioni, 1995]. In addition, a server promise weight can be used, weighting scores more highly if they originate from a server judged to be more useful [Gauch et al., 1996].

In *rank based* merging methods, server-assigned ordinal ranks are used in generation of the merged list. Ranks may be simply interleaved [Smeaton and Crimmins, 1996]. Alternatively the gap between documents from a list may be made inversely proportional to server promise, as in Yuwono interleaving [Yuwono and Lee, 1997], or an N-sided die, weighted in proportion to server promise, may be used to determine the order of documents. In the latter approach [Voorhees, 1995] incoming rankings of varying lengths are used, in contrast with the fixed length lists used in evaluations here, consequently an approximation of Voorhees interleaving is evaluated here (see Section 4.2).

In *content based* methods, the client downloads the documents D_m from document servers and analyses them in order to produce a ranking. The metasearcher Inquirus [Lawrence and Giles, 1998] does this, employing a ranking algorithm without collection statistics to generate the merged ranking. One advantage of content based methods is that even if indexes are out of date because of document changes and deletions, the merged ranking will be based on current document contents (which will also be cached for viewing by the user). The disadvantage is that the documents must be downloaded, incurring time and bandwidth costs for each search.

3 New Merging Techniques

In this paper two new techniques are introduced; use of Reference Statistics and the Feature Distance ranking algorithm.

3.1 Reference Statistics

Use of collection statistics can improve merging effectiveness, but no method for efficiently collating collection statistics is available without use of integrated servers. Instead of attempting to collate collection statistics, a reference statistic database can be used, containing all the relevant statistics for *some* set of

documents. This set may be some proportion of the documents to be searched (10% is used here), or a completely separate collection. Using a set of Reference Statistics, a metasearcher such as Inquirus could employ a more effective ranking algorithm by substituting a reference statistic wherever a collection statistic would normally appear.

3.2 Feature Distance Ranking Algorithms

One problem with with content based merging methods is the time taken and cost involved in downloading the documents. One alternative in such a case is to download only the beginning of each document, thereby reducing the time between transmission of document content requests and the beginning of content analysis. With this in mind, a ranking algorithm was developed which gives a higher weighting to term occurrences near the start of the document. If this algorithm was effective in the full text case, the degradation in effectiveness in the partial download case would hopefully be small.

Feature distance algorithms , are based on the occurrence of *features* (query terms) in the document. They are based on intuition that a feature is less important if: a) it appears near the end of the document, b) it is not near other features, c) it is a term which has has occurred many times in the document already, and d) it is a common term in the collection. Therefore, the contribution (w) of a feature to the overall document score is determined by its total character offset (l) into the document, the distance in characters (d) between it and the previous feature, the number of times (n) that term has occurred so far, and the document frequency of the term (df). Experiments were conducted with several different feature distance weighting functions. Functions w_A and w_B, each with slightly different properties, were selected for presentation in Sections 5 and 6.

$$w_A = \frac{1}{n \cdot \sqrt{d \cdot df} \cdot \ln(l)}$$

$$w_B = \frac{1}{n^{1.1} \cdot \ln(d) \cdot \ln(df + 1) \cdot \ln(l)}$$

In both cases, the contributions of all the documents features F are summed, so the overall score achieved by the document is $score = \sum^{F} w$.

4 Experimental Framework

The Text REtrieval Conference (TREC) [Voorhees and Harman, 1997] provides an effectiveness evaluation framework, including sets of documents, topics and relevance judgements. The effectiveness of a merging method is defined as its ability to rank relevant documents above irrelevant ones in R_m. In order to measure this, five lists of 30 documents are generated, one from each TREC-6 collection. These incoming lists $R_1 \ldots R_5$ are already ranked in order of likely relevance to a TREC-6 topic, and the merging method is employed to generate

the merged ranking with respect to the same topic. Then human judgements of which documents are relevant to that topic are used to judge the effectiveness of the merged ranking.

The effectiveness measure used here is *average precision*:

$$Average\ Precision = \frac{\sum_{i=1}^{num\_rel\_ret} \frac{i}{rank(i)}}{num\_rel}$$

where $rank(i)$ is the rank of the ith relevant document, *num_rel_ret* is the number of relevant documents in the ranking being evaluated, and *num_rel* is the number of relevant documents in the collections being searched

It is usual when performing TREC tests to evaluate performance over 50 topics. The mean effectiveness of a merging method over 50 topics is defined here as an observation. However, the effectiveness of a merging method may depend on the techniques used for retrieval. For this reason, results from five different retrieval systems per collection were simulated. This allows 5^5 different configurations of incoming lists, each including results from between one and five systems, but always including one list from each collection. In order to avoid bias towards any particular merging method, all results presented in Section 5 cover all 3125 possible observations.

4.1 Details of Method

Collections All five TREC-6 collections were used in this evaluation; the LA Times, Foreign Broadcast Information Service, the Financial Times, the US Federal Register and the US Congressional Record.

Server Algorithms Five TREC-6 runs were used for generating input rankings. Each run contains the top 1000 ranked documents for each of 50 topics. Runs were from:

- University of California Berkeley, based on logistic regression (`Brkly22`),
- Cornell University, based on the vector space model (`Cor6A3cll`),
- City University, based on the Okapi probabilistic model (`city6at`),
- MDS RMIT, using a limited-context vector space model (`mds602`), and
- Queens College CUNY, using a modified probabilistic model trained using a spreading activation network (`pirc7At`).

Input Rankings Five input rankings (i.e. $R_1 \ldots R_5$) were generated from each of the above official TREC runs by extracting the best 30 documents from each of the five collections. Although rankings generated in this fashion will differ slightly from top 30 rankings which would have been produced by individual runs on each collection, they are based on highly effective, complex and varied retrieval techniques. In addition, the results were produced independently of the current study, and are available for use by other researchers in replicating these results.

Topics TREC topics are system-independent English language statements of user information need. Topics were used both for generation of the TREC runs described above and in the content-based merging methods. In the latter case, a set of unstemmed, weighted query terms was extracted from each topic, with term weights corresponding to the number of times the term occurred in the topic.

4.2 Details of Merging Methods

Random A merging method which generates o_m randomly was included in the evaluation to provide a baseline for effectiveness.

Rank and Score Based Methods Included in this evaluation are the rank and score based methods described in Section 2.3. Every effort has been made to correctly implement the techniques described. However, true Voorhees interleaving uses variable length input lists and a specific server promise measure not available here. The "V Interleaving" results presented in Section 5 are a modified version of Voorhees methods, based on a different server promise measure and using fixed length lists. Although the lists all start off with the same length, selection weights are made proportional to collection promise, at a level less than or equal to the length of the list. As documents are removed from the list, the weight is reduced by one, unless the weight is one and the list is not yet empty.

Ranking Algorithms Using Reference Statistics or collection statistics obtained through integrated servers it is possible to apply any of a large range of ranking algorithms. The following ranking formulae are used in evaluations here:

- Feature Distance w_A and w_B (described above)
- the Inquirus ranking algorithm

$$R = c_1 N_p + \left(c_2 - \frac{\sum_{i=1}^{N_p-1} \sum_{j=i+1}^{N_p} min(d(i,j), c_2)}{\sum_{k=1}^{N_p-1} (N_p - k)} \right) \Big/ \frac{c_2}{c_1} + \frac{N_t}{c_3}$$

  where R is the document's score, N_p is the number of query terms present in the document (each term is counted once), N_t is the number of query term occurrences in the document, d(i,j) is the minimum distance (number of characters) between the ith and jth query terms, and c_1, c_2 and c_3 are constants
- simple $tf \cdot idf$

$$w_t = \frac{tf_d}{df}$$

- the Okapi BM25 [Walker et al., 1997] variant described in [Hawking et al., 1997]

$$w_t = tf_d \times \frac{\log(\frac{cs - df + 0.5}{df + 0.5})}{2 \times (0.25 + 0.75 \times \frac{dl}{avdl}) + tf_d}$$

– a version of the Okapi BM25 with no document frequency information, and length normalisation according to a constant rather than the true average document length:

$$w_t = tf_d \times \frac{1}{2 \times (0.25 + 0.75 \times \frac{dl}{4096}) + tf_d}$$

where w_t is the relevance weight assigned to a document due to query term t (this weight is multiplied by the query weight of t), tf_d is the number of times t occurs in the document, cs is the total number of documents, df is the document frequency of t, dl is the length of the document and $avdl$ is the average document length

Server Promise The server promise measure used here is the sum over all query terms of $w \cdot df \cdot icf$, where w is the query weight assigned to that term, df is the document frequency of that term in the collection and icf is the inverse of the number of collections containing that term.

Reference Statistics Reference collection statistics were taken from a ten percent sample of the collections searched, simply by choosing every 10th file encountered on disk. If there was no document frequency entry for a particular query term in the reference statistic database, the frequency was assumed to be one.

5 Results

All isolated methods are compared in results Table 1, and for each method the proportion of configurations for which feature distance w_A was superior is listed in the rightmost column. A more detailed comparison of two methods across 3125 configurations is presented in Figure 1. Changes in effectiveness of methods using collection statistics due to use of Reference rather than real statistics are documented in Table 2. Table 3 shows the effect on content based methods of downloading only part of each document.

6 Discussion

The average precision figures in Table 1 show that content based methods, in particular those incorporating Reference Statistics, were the most effective of the isolated-server merging methods. The difference in mean average precision between Okapi and Feature Distance w_A is only 0.006, but the latter is more effective in 93% of cases. The extent of these differences over all 3125 configurations is shown in Figure 1. This fairly uniform improvement suggests that further study of merging with Feature Distance algorithms is warranted. The other noteworthy result in Table 1 is that score based methods performed better

Method	Information	Average Precision		Configurations where w_A is better
		Mean	Standard Deviation	
Random	None	0.062	0.005	100%
Inquirus	C	0.085	0.008	98%
Raw Scores	S	0.123	0.038	100%
V Interleaving	RW	0.126	0.015	100%
$tf \cdot idf$	CX	0.127	0.012	100%
Yuwono Interleaving	RW	0.131	0.014	100%
Interleaving	R	0.132	0.011	100%
Scaled Scores	S	0.148	0.022	99%
Weighted Scaled Scores	SW	0.155	0.023	99%
Okapi (no df)	C	0.177	0.007	100%
Okapi	CX	0.185	0.009	93%
Feature Distance w_B	CX	0.189	0.011	69%
Feature Distance w_A	CX	0.191	0.010	0%

Table 1. Effectiveness of isolated merging methods (see Section 4.2 for details on methods). Means and standard deviations of average precision are over 3125 observations. The proportion of configurations for which w_A was superior is also listed. Merging information used; S: Scores, R: Ranks, W: Server promise weight, C: Document content, and X: Reference Statistics.

Method	Mean average precision		Configurations where integration is better
	Isolated	Integrated	
$tf \cdot idf$	0.127	0.144	100%
Okapi	0.185	0.187	99%
Feature Distance w_B	0.189	0.182	0%
Feature Distance w_A	0.191	0.191	40%

Table 2. Relative effectiveness of ranking algorithms in isolated and integrated environments (Reference Statistics are used in the isolated environment). Mean average precisions are over 3125 observations.

Method	Mean average precision		Configurations where full download is better
	Full download	First 4 kB	
Inquirus	0.085	0.127	0%
$tf \cdot idf$	0.127	0.129	38%
Okapi (no df)	0.177	0.166	100%
Okapi	0.185	0.172	100%
Feature Distance w_B	0.189	0.176	99%
Feature Distance w_A	0.191	0.173	100%

Table 3. Effectiveness of ranking algorithms using full and partial document download. Mean average precisions are over 3125 observations.

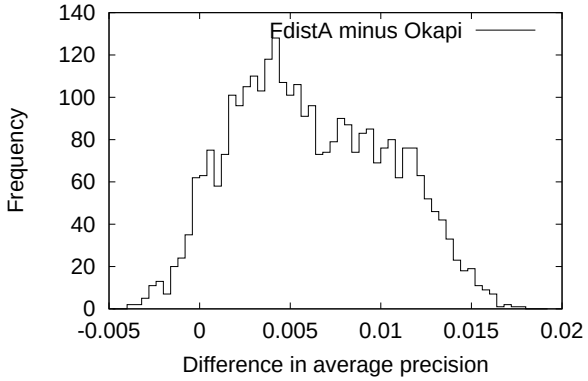

Fig. 1. Histogram of pairwise differences in average precision between Feature Distance w_A and Okapi over 3125 configurations.

than rank based methods, and when server promise weights were used, only the score based method improved.

Feature Distance algorithm w_A had the highest mean average precision both with integrated collection statistics and with Reference Statistics (see Table 2). While the w_A results remained constant, algorithm w_B actually improved with use of reference statistics. The reason for this, and the reason Okapi and $tf \cdot idf$ were less effective in almost all configurations, is not clear.

The Inquirus and $tf \cdot idf$ algorithms became more effective when operating over only the first 4 kB of each document (see Table 3). This may be because neither of these algorithms normalise on document length, and the 4 kB limit acted as a crude form of length normalisation. The other algorithms were less effective with partial download. Feature Distance algorithms were designed to be tolerant to partial downloads, but the degradation in effectiveness was worse (or no better) than for the Okapi algorithms.

The Okapi algorithm achieved an average precision of 0.185, considerably lower than the 0.288 officially achieved by the `city6at` run in TREC-6. However, the official TREC average precision is calculated over 1000 documents per topic rather than 150. Furthermore, the merging results contained 30 documents from each collection in all cases, even if some collections contained few or no relevant documents.

A number of parameters were not varied in this paper, such as the query generation method, choice of server promise measure, collection statistic sampling method and input ranking generation method. Changing these factors could affect the evaluation results in various ways which are not explored here. However, the results presented here suggest that the new methods warrant further investigation.

7 Conclusions

In an isolated-server environment, such as the World Wide Web, Reference Statistics allow more effective ranking algorithms to be applied when performing content based merging. In addition, Feature Distance ranking algorithms proved effective in both the isolated and integrated server cases. Experiments with content based ranking on partial documents showed that effectiveness is reduced in most cases, and Feature Distance methods are no better than other algorithms in this respect.

Experiments with Reference Statistics from different sources — such as a different collection or a previous version of the current collection — have not yet been carried out. Such experiments could indicate how generally applicable Reference Statistics are, and how the choice of statistics affects the quality of the final results. Experiments into the efficiency of downloading documents using a protocol such as HTTP could also be carried out, to indicate the trade-off between time taken and effectiveness in the partial download case. In addition, other applications of Reference Statistics could be explored, such as generation comparable relevance scores on the server side rather than the client side, or use in query driven automatic hypertext navigation.

Acknowledgements

Thanks to Richard Walker for TEX support and to NIST for TREC resources.

References

[Callan et al., 1995] Callan, James P., Lu, Zihong, and Croft, W. Bruce (1995). Searching distributed collections with inference networks. In [Fox et al., 1995], pages 12–20.

[Dreilinger and Howe, 1997] Dreilinger, Daniel and Howe, Adele E. (1997). Experiences with selecting search engines using metasearch. *ACM Transactions on Information Systems*, 15(3):195–222.

[Fox et al., 1995] Fox, Edward A., Ingwersen, Peter, and Fidel, Raya, editors (1995). *Proceedings of ACM SIGIR '95*, Seattle, Washington. ACM Press.

[Gauch et al., 1996] Gauch, Susan, Wang, Guijun, and Gomez, Mario (1996). ProFusion*: Intelligent fusion from multiple, distributed search engines. *The Journal of Universal Computer Science*, 2(9):637–649.

[Gravano et al., 1997] Gravano, Luis, Chang, Kevin, Garcia-Molina, Hector, Lagoze, Carl, and Paepcke, Andreas (1997). STARTS - Stanford protocol proposal for internet retrieval and search. `http://www-db.stanford.edu/~gravano/starts.html`.

[Harman, 1992] Harman, Donna (1992). Ranking algorithms. In *Information Retrieval. Data Structures and Algorithms*, chapter 14, pages 363–392. Prentice Hall, Upper Saddle River, NJ.

[Hawking and Thistlewaite, 1998] Hawking, David and Thistlewaite, Paul (1998). Methods for information server selection. Accepted by ACM TOIS April 98.

[Hawking et al., 1997] Hawking, David, Thistlewaite, Paul, and Craswell, Nick (1997). ANU/ACSys TREC-6 experiments. In [Voorhees and Harman, 1997], pages 275–290.

[Kirsch, 1997] Kirsch, Steven T. (1997). Distributed search patent. U.S. Patent 5,659,732. Infoseek Corporation.

[Krester et al., 1998] Krester, De, Moffat, Shimmin, and Zobel (1998). Methodologies for distributed information retrieval. In *Proceedings of the 18th International Conference on Distributed Computing Systems.*

[Lawrence and Giles, 1998] Lawrence, Steve and Giles, C Lee (1998). Inquirus, the NECI meta search engine. In *Proceedings of the Seventh International World Wide Web Conference.*

[Mazur, 1994] Mazur, Zygmunt (1994). Models of a distributed information retrieval system based on thesauri with weights. *Information processing & management.*, 30(1):61.

[Selberg and Etzioni, 1995] Selberg, E. and Etzioni, O. (1995). Multi-service search and comparison using the MetaCrawler. In *Proceedings of the 1995 World Wide Web Conference.*

[Smeaton and Crimmins, 1996] Smeaton, Alan F. and Crimmins, Francis (1996). Using A data fusion agent for searching the WWW.

[Viles and French, 1995] Viles, Charles L. and French, James C. (1995). Dissemination of collection wide information in a distributed information retrieval system. In [Fox et al., 1995], pages 12–20.

[Voorhees, 1995] Voorhees, Ellen M. (1995). Siemens TREC-4 report: Further experiments with database merging. In Harman, D. K., editor, *Proc. Fourth Text Retrieval Conference (TREC-4)*, pages 121–130, Gaithersburg, MD. U.S. National Institute of Standards and Technology.

[Voorhees and Harman, 1997] Voorhees, E. M. and Harman, D. K., editors (1997). *Proceedings of the Sixth Text Retrieval Conference (TREC-6)*, Gaithersburg, MD. U.S. National Institute of Standards and Technology.

[Walczuch et al., 1994] Walczuch, Nikolaus, Fuhr, Norbert, Pollmann, Michael, and Sievers, Birgit (1994). Routing and ad-hoc retrieval with the trec-3 collection in a distributed loosely federated environment. In Harman, D. K., editor, *Proceedings of the Third Text Retrieval Conference (TREC-3)*, pages 135–144, Gaithersburg, MD. U.S. National Institute of Standards and Technology. NIST special publication 500-225.

[Walker et al., 1997] Walker, S, Robertson, S E, Boughanem, M, Jones, G J F, and Sparck-Jones, K (1997). Okapi at TREC-6. In [Voorhees and Harman, 1997].

[Yuwono and Lee, 1997] Yuwono, Budi and Lee, Dik L. (1997). Server ranking for distributed text retrieval systems on the internet. In Topor, Rodney and Tanaka, Katsumi, editors, *DASFAA '97*, pages 41–49, Melbourne. World Scientific, Singapore.