Probabilistic Data Fusion on a Large Document Collection

David Lillis, Fergus Toolan, Rem Collier, and John Dunnion

School of Computer Science and Informatics
University College Dublin
Ireland
{david.lillis, fergus.toolan, rem.collier, john.dunnion}@ucd.ie

Abstract. Data Fusion is the process of combining the output of a number of Information Retrieval algorithms into a single result set, to achieve greater retrieval performance. *ProbFuse* is a probabilistic data fusion algorithm that has been shown to outperform the CombMNZ algorithm in a number of previous experiments. This paper builds upon this previous work and applies *probFuse* to the much larger Web Track document collection from the 2004 Text REtreival Conference.

1 Introduction

In the past, multiple solutions have been proposed to solve the Information Retrieval (IR) problem of identifying documents that have relevance to particular user queries. However, individual IR systems will typically return different documents in response to a given query, even when the retrieval performance of the individual systems is similar [1].

Much research has been carried out into combining the outputs of a number of different IR systems into a single result set, in order to achieve greater retrieval performance than any individual system. This is known as "data fusion" when the underlying systems have access to the same document collection [2].

ProbFuse is a novel data fusion algorithm that uses the probability that particular documents are relevant to a given query in order to produce a fused result set. In the past, *probFuse* has been shown to outperform the common CombMNZ algorithm on small document collections [3] and also on data taken from Text REtrieval Conferences (TREC) [4]. In order to run *probFuse*, values for two variables, t and x, must be determined. In previous work, these values have been set empirically in the course of our experiments. This paper, however, uses the variables selected during experiments on the TREC-3 and TREC-5 document collections [4]. The aim is to demonstrate that these values are not collection-specific and that it is possible to determine values that will achieve superior performance on many document collections without the necessity of calculating different values for each.

In addition, the document collection used (the Web Track from the TREC-2004 conference) is much larger than those used in previous work. This has

the effect that the relevance judgments are far less complete (i.e. only a small fraction of the documents in the collection have been judged to be relevant or nonrelevant to each query, leaving most document unjudged). Because the *probFuse* algorithm relies on training data to calculate the probability that particular documents are relevant, we aim to investigate the effects of this decreased level of completeness on fusion performance.

Section 2 is a general description of the data fusion problem. In section 3 we outline work that has been carried out by other researchers in the past to tackle the data fusion problem. Section 4 describes the *probFuse* data fusion algorithm. In section 5 we outline an experiment to demonstrate the effectiveness of *probFuse* on the Web Track collection from the 2004 TREC Conference, including a comparison with the standard CombMNZ algorithm [5]. Finally, section 6 closes with our conclusions and intended future work.

2 Problem Description

When performing data fusion, there are three "effects" that may be leveraged in order to achieve greater retrieval performance [6]. The "Chorus Effect" describes a situation where a document is regarded to be relevant by a number of the underlying systems whose results are being fused. Fusion techniques that rank documents higher based on this agreement will tend to achieve better performance in this situation. The Chorus Effect is the key difference between the treatment of the data fusion and collection fusion tasks. In data fusion, the presence of a document in numerous result sets can be used as evidence of relevance, as each of the underlying systems has access to the same document collection. Where the document collections are disjoint, documents will only be returned by, at most, a single input result set. Experiments by Lee [7] have shown that the Chorus Effect is a significant factor when performing data fusion.

The most relevant documents are likely to be returned near the beginning of each result set. Where this is the case, fusion algorithms that "skim" the top documents from each of its input result sets and combine these to form the fused output will perform well. This is known as the "Skimming Effect".

The third "effect" to be taken into account is the "Dark Horse" effect. This describes a situation where one underlying system produces results that are of an unusually high (or low) standard. There is an apparent contradiction between this and the Chorus Effect. Whereas the Chorus Effect argues in favour of taking as many input result sets into account as possible, the Dark Horse Effect favours the identification of a single result set that is of a higher quality to the others.

3 Background Research

Previous approaches to data fusion tend to fall into two broad categories. Some approaches use the rank in which a document appears in each result set to produce the fused output. Others use the scores assigned to each document by the underlying IR systems, generally utilising a normalisation step to map these

scores into a common range. Two such score-based techniques were proposed by Fox and Shaw [5]. CombSum ranks documents according to the sum of the normalised scores assigned to them by the underlying systems. CombMNZ multiplies the CombSum score by the number of result sets in which the document is returned. A study by Lee [7] achieved positive results by applying CombMNZ to the TREC-3 dataset. Real-world meta search engines such as *MetaCrawler*[8] have used CombSum as their fusion algorithms. The Linear Combination model is a common approach to score-based fusion [6], in which the scores assigned to documents by the underlying systems are multiplied by weights associated with each system.

An early rank-based technique is *interleaving* [9], which takes a document from each result set in turn in a round-robin fashion. Weighted variations have also been proposed [10]. Other rank-based approaches include a variation of CombMNZ proposed by Lee [7] and the *Borda-fuse* [11] and *Condorcet-fuse* [12] voting algorithms developed by Aslam and Montague. Manmatha et al. were able to calculate the probability that a document was relevant based on its position within the result set using Bayes' Rule [13].

Other approaches that use data other than ranks or scores have also been proposed. For example, some use the contents of the documents returned [14, 15]. Others require the underlying systems to provide metadata about the documents being returned, rather than merely assigning a ranking score [16].

4 Probability-Based Fusion

This section describes *probFuse*, a data fusion algorithm that uses the probability that documents are relevant to a query to produce the final fused result set.

The inputs to *probFuse* are the result sets returned by a number of different input systems in response to particular queries. Each input system has access to the same document collection.

The *probFuse* algorithm divides the input result sets into segments. The number of segments into which to divide each result set, x, is determined empirically. For a result set containing 1000 documents, if $x = 100$, each segment will contain 10 documents. An example of the segmentation of documents can be found in [3].

The first stage of applying *probFuse* is a training stage to determine the probability that a document returned in a particular segment by a particular input system is relevant. In a training set of Q queries, $P(d_k|m)$, the probability that a document d returned in segment k is relevant, given that it has been returned by retrieval model m, is given by:

$$P(d_k|m) = \frac{\sum_{q=1}^{Q} \frac{|R_{k,q}|}{|k|}}{Q} \tag{1}$$

where $|R_{k,q}|$ is the number of documents in segment k that are judged to be relevant to query q, and $|k|$ is the total number of documents in segment k.

Because this calculation takes all of the documents in each segment into account, this is known as *probFuseAll*.

In previous work, a variation of this calculation, known as *probFuseJudged* achieved slightly better retrieval performance where relevance judgments were incomplete [4]. When using *probFuseJudged*, $P(d_k|m)$ is given by

$$P(d_k|m) = \frac{\sum_{q=1}^{Q} \frac{|R_{k,q}|}{|R_{k,q}|+|N_{k,q}|}}{Q} \tag{2}$$

where $|N_{k,q}|$ is the number of documents in segment k that are judged to be nonrelevant to query q.

ProbFuseJudged only takes documents in each segment that have been judged either relevant or nonrelevant into account when calculating the probability of relevance. In contrast, *probFuseAll* takes all of the documents in a segment into account, assuming unjudged documents to be nonrelevant.

Once these probabilities have been calculated for each segment number for each input system, fusion can take place. The ranking score S_d for each document d is given by

$$S_d = \sum_{m=1}^{M} \frac{P(d_k|m)}{k} \tag{3}$$

where M is the number of retrieval models being used, $P(d_k|m)$ is the probability of relevance for a document d_k that has been returned in segment k by retrieval model m, and k is the segment that d appears in (1 for the first segment, 2 for the second, etc.). If a particular document d is not included in a result set at all, $P(d_k|m)$ is considered to be zero.

Using the sum of the probabilities to generate the final ranking score for each document makes use of the Chorus Effect. The division by k gives greater weight to documents appearing in early segments and so utilises the Skimming Effect.

5 Experiments and Evaluation

This section describes an experiment to evaluate the performance of *probFuse* on the Web Track collection from the TREC-2004 conference.

In previous research on *probFuse*, the values for x, the number of segments into which to divide each result set and t, the percentage of queries to use for training purposes have been calculated empirically using the document collection that was being evaluated [4, 17]. The aim of these experiments was to demonstrate that values could be identified that would cause *probFuse* to achieve greater performance than CombMNZ. For this experiment, we have the taken the values that were shown to perform best on the data from the TREC-3 and TREC-5 conferences. Each result set was divided into 25 segments and 50% of the available queries were used as training data. This means that the values are independent of the document collection being used in this case.

The goal of this experiment is to demonstrate that values for the number of segments and the training set size that have been calculated by using one document collection can be used to achieve high performance on other document collections. We demonstrate this by using the popular CombMNZ fusion algorithm as a baseline. CombMNZ has been shown to achieve high performance on data fusion tasks [7] and has become the standard data fusion technique against which to compare new algorithms [11].

An additional goal is to investigate the effects of using a document collection where relevance judgments are extremely incomplete.

The TREC-2004 Web Track data includes 74 topfiles. Each of these topfiles contains result sets returned by one IR system for each of 225 queries. Five runs were performed and the evaluation results below are the average of the scores over all five runs. For each run, 6 random topfiles were selected, ensuring that no topfile was used in multiple runs.

For each experimental run, the order of the queries was randomised and fusion was then performed using *probFuseAll*, *probFuseJudged* and CombMNZ. This was done five times for each run and the evaluation scores associated with each run is the average of these. Doing this ensured that performance was not influenced by the order of the queries.

Evaluation of the fused result sets was carried out using the Mean Average Precision (MAP) and bpref measures. MAP is the mean of the precision scores obtained after each relevant document has been retrieved. Relevant documents that are not included in the result set are given a precision of zero [18]. MAP assumes that documents that have not been judged are nonrelevant. The bpref measure evaluates the relative position of relevant and nonrelevant documents, ignoring documents that are unjudged. It was proposed by Buckley and Voorhees to cater for situations where relevance judgments are incomplete [18].

Table 1 shows the results for the *probFuseAll* and *probFuseJudged* algorithms for each of the five experimental runs. The information presented in this table is illustrated graphically in figure 1.

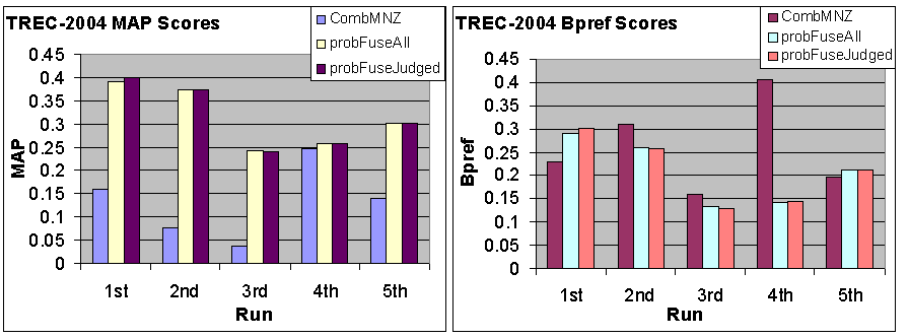

Fig. 1. TREC-2004 MAP and bpref scores for $t = 50\%$ and $x = 25$

Table 1. TREC-2004 performance of five individual runs for $t = 50\%$ and $x = 25$ using *probFuseAll* and *probFuseJudged*

	CombMNZ		*probFuseAll*	
	MAP	bpref	MAP	bpref
first	0.16042	0.22868	0.39154 (+144.07%)	0.29016 (+26.88%)
second	0.07808	0.31030	0.37536 (+380.74%)	0.25848 (-16.7%)
third	0.03846	0.15788	0.24418 (+534.89%)	0.13236 (-16.16%)
fourth	0.24544	0.40436	0.25862 (+5.37%)	0.14048 (-65.26%)
fifth	0.14130	0.19550	0.30278 (+114.28%)	0.21084 (+7.85%)
Average	0.13274	0.25934	0.31450 (+235.87%)	0.20646 (-12.68%)

	CombMNZ		*probFuseJudged*	
	MAP	bpref	MAP	bpref
first	0.16042	0.22868	0.39920 (+148.85%)	0.30082 (+31.55%)
second	0.07808	0.31030	0.37340 (+378.23%)	0.25558 (-17.63%)
third	0.03846	0.15788	0.24132 (+527.46%)	0.12748 (-19.26%)
fourth	0.24544	0.40436	0.25924 (+5.62%)	0.14204 (-64.87%)
fifth	0.14130	0.19550	0.30284 (+114.32%)	0.21120 (+8.03%)
Average	0.13274	0.25934	0.31520 (+234.9%)	0.20742 (-12.44%)

The bpref scores do not show a consistent pattern when comparing *probFuse* and CombMNZ. Both *probFuseAll* and *probFuseJudged* achieve higher bpref scores on the "first" and "fifth" runs, whereas CombMNZ performs better on the others. The degree by which one technique outperforms the other varies also. Whereas for the "fourth" run, CombMNZ's bpref score is over 60% higher than either *probFuse* variant, its score for the "first" run is lower by over 25%.

The MAP data contrasts sharply with this. Using this measure, the scores for *probFuse* are higher than those of CombMNZ in all cases, and drops below a 100% increase only for the "fourth" run. The average improvement in MAP score over CombMNZ is over 230% for both *probFuseAll* and *probFuseJudged*.

In previous experiments on smaller TREC datasets, *probFuseJudged* achieved slightly better performance than *probFuseAll*. One of the aims of this experiment is to investigate whether the scores for the two variations of *probFuse* diverge as the relevance judgments' level of completeness decreases. From table 1, we can see that there is no significant difference between the scores for *probFuseAll* and *probFuseJudged* using either evaluation measure. In fact, the difference between the scores does not exceed 4% in any case. Thus we can conclude that the decision to base the probability scores on all documents or just judged documents does not have a significant effect on fusion performance, even in cases where the level of completeness of the relevance judgments is very low.

The contrast between the evaluation results for MAP and bpref is of interest. Whereas the MAP scores clearly indicate that the performance of *probFuse* is superior to that of CombMNZ, the bpref scores are inconclusive. In order to explain this, it is useful to examine the distribution of relevant, nonrelevant and

unjudged documents in the result sets created by *probFuseAll*, *probFuseJudged* and CombMNZ. Figure 2 illustrates these distributions. This figure shows the percentage of judged relevant, judged nonrelevant and unjudged documents at various positions in the result sets. Each data point represents the this percentage for a group of 10 documents. For example, data points for position 0 on the x-axis represent documents from position 0 to position 9. All result sets produced by *probFuseAll*. *probFuseJudged* and CombMNZ during our experiments were used for this analysis.

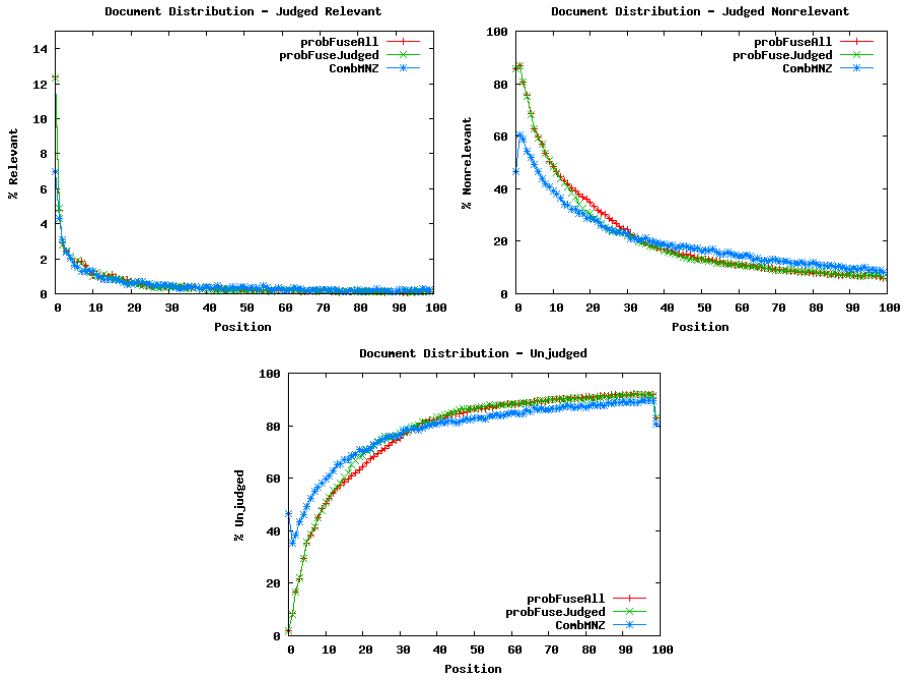

Fig. 2. Distribution of judged relevant, judged nonrelevant and unjudged documents

Both variations of *probFuse* return more judged relevant documents than CombMNZ in early positions. This will have a positive effect on their MAP scores, as MAP rewards highly-placed relevant documents more than those that are returned further down the result set or not returned at all. CombMNZ returns a greater number of unjudged documents towards the top of the result sets it produces. This has a detrimental effect on its MAP score, as these documents are assumed to be nonrelevant for this measure. However, bpref ignores these documents. This means that they have no effect on the bpref score despite the fact that relevant documents are then pushed further down the fused result set. The tendency to return nonrelevant documents in early positions is higher for

probFuse. When evaluated using MAP, this is no different to returning unjudged documents. However, it does have a detrimental effect on bpref scores to return nonrelevant documents above those that have been judged relevant.

Table 2. Average number of relevant documents returned

Average Relevant Documents	882.68
probFuseAll	690.84 (78.27%)
probFuseJudged	670.98 (76.01%)
CombMNZ	661.96 (74.99%)

Table 2 shows the average number of relevant documents that were returned by each fusion technique over all the runs. The "Average Relevant Documents" shows the average number of relevant documents that were available for retrieval. This is an average because the order of the queries were changed, meaning that a different set of queries was being used for fusion each time. From this figure, we can see that overall recall was higher for *probFuse* than for CombMNZ, meaning that more relevant documents were retrieved in total.

Because 50% of the 225 available queries were used for training, this means that 113 were used for fusion each time. Given that the average number of available relevant documents is 882.68, this means that an average of only 7.81 relevant documents was available to be retrieved for each query. The variation of the number of available relevant documents is due to the fact that the specific queries being used for fusion change as the order of the queries is randomised. This data is significant when interpreting the results returned by bpref. If R relevant documents are available for a query, bpref only considers the first R nonrelevant documents. Consider a query for which there are 8 relevant documents available. If a fusion technique returns 8 nonrelevant documents in its first 20 results, returning a relevant document in 21st place is the same as returning the same document in 1,000th place or not at all. In contrast, MAP always considers a higher-placed relevant document to be superior, and also prefers a relevant document that has been returned in a result set to one that was not returned.

Additionally, since bpref ignores unjudged documents, this allows CombMNZ to return relevant documents further down its result sets without negatively impacting its bpref scores. However, this is reflected in the inferior MAP scores.

The tendency of *probFuseAll* and *probFuseJudged* to return a greater number of relevant documents in early positions is rewarded by the superior MAP scores shown in table 1. Typical users of IR systems expect to find relevant documents at the top of result sets. One study found that 85.2% of surveyed users examined only the top 10 results presented to them [19]. This indicates that the tendency of CombMNZ to return relevant documents in lower positions, which is not penalised by bpref, is not desirable behaviour in a real-world system.

The vastly superior MAP scores achieved by *probFuse* over CombMNZ, together with the higher overall recall and greater tendency to return relevant documents in early positions shown by *probFuse* support the conclusion that the *probFuse* algorithms display superior performance to CombMNZ.

6 Conclusions and Future Work

This paper presents the results of applying the *probFuse* data fusion algorithm on the large Web Track collection from the TREC-2004 conference. Evaluating the performance of *probFuseAll* and *probFuseJudged* using the MAP evaluation measure showed the *probFuse* algorithms achieving improvements of over 230% over CombMNZ. Additionally, *probFuse* achieved higher recall overall and tended to return more relevant documents at the top of the fused result sets produced.

The completeness of the relevance judgments for the Web Track collection is lower than that of others that have been used in previous experiments. Despite this, no significant difference was observed between using all documents in the calculation of the probability of relevance, rather than using just those documents that have been judged to be either relevant or nonrelevant.

To date, experiments to evaluate the effectiveness of *probFuse* have used training data from the same document collection as is used for fusion. In the future, we intend to investigate the results of using training data from one document collection and using this to perform fusion on another. Additionally, the result sets provided in the TREC data are of a fixed length of 1,000 documents. In order for *probFuse* to be applicable in real-world situations, it must be capable of performing well on variable-length result sets, as some queries will result in more documents being returned by the underlying IR systems than others.

We also intend to investigate methods of inferring the probability of relevance without the necessity for relevance judgments to be available. Previous work by Manmatha et al. [13] in estimating the probability that a document is relevant to a particular query may be useful in this regard.

References

1. Harman, D.: Overview of the first Text REtrieval Conference (TREC-1). In: SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (1993) 36–47
2. Aslam, J.A., Montague, M.: Bayes optimal metasearch: a probabilistic model for combining the results of multiple retrieval systems. In: SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2000) 379–381
3. Lillis, D., Toolan, F., Mur, A., Peng, L., Collier, R., Dunnion, J.: Probability-based fusion of information retrieval result sets. Artificial Intelligence Review (2006)
4. Lillis, D., Toolan, F., Collier, R., Dunnion, J.: ProbFuse: a probabilistic approach to data fusion. In: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, New York, USA, ACM Press (2006) 139–146

5. Fox, E.A., Shaw, J.A.: Combination of multiple searches. In: Proceedings of the 2nd Text REtrieval Conference (TREC-2), National Institute of Standards and Technology Special Publication 500-215. (1994) 243–252
6. Vogt, C.C., Cottrell, G.W.: Fusion via a linear combination of scores. Information Retrieval **1** (1999) 151–173
7. Lee, J.H.: Analyses of multiple evidence combination. SIGIR Forum **31** (1997) 267–276
8. Selberg, E., Etzioni, O.: The MetaCrawler architecture for resource aggregation on the Web. IEEE Expert (1997) 11–14
9. Voorhees, E.M., Gupta, N.K., Johnson-Laird, B.: The collection fusion problem. In: Proceedings of the Third Text REtrieval Conference (TREC-3). (1994) 95–104
10. Voorhees, E.M., Gupta, N.K., Johnson-Laird, B.: Learning collection fusion strategies. In: SIGIR '95: Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (1995) 172–179
11. Aslam, J.A., Montague, M.: Models for metasearch. In: SIGIR '01: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, ACM Press (2001) 276–284
12. Montague, M., Aslam, J.A.: Relevance score normalization for metasearch. In: CIKM '01: Proceedings of the Tenth International Conference on Information and Knowledge Management, New York, NY, USA, ACM Press (2001) 427–433
13. Manmatha, R., Rath, T., Feng, F.: Modeling score distributions for combining the outputs of search engines. In: SIGIR '01: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, ACM Press (2001) 267–275
14. Craswell, N., Hawking, D., Thistlewaite, P.B.: Merging results from isolated search engines. In: Australasian Database Conference, Auckland, New Zealand (1999) 189–200
15. Lawrence, S., Giles, C.L.: Inquirus, the NECI meta search engine. In: Seventh International World Wide Web Conference, Brisbane, Australia, Elsevier Science (1998) 95–105
16. Gravano, L., Chang, K., Garcia-Molina, H., Paepcke, A.: Starts: Stanford protocol proposal for internet retrieval and search. Technical report, Stanford, CA, USA (1997)
17. Lillis, D., Toolan, F., Mur, A., Peng, L., Collier, R., Dunnion, J.: Probability-based fusion of information retrieval result sets. In: Proceedings of the 16th Irish Conference on Artificial Intelligence and Cognitive Science (AICS 2005), Portstewart, Northern Ireland, University of Ulster (2005) 147–156
18. Buckley, C., Voorhees, E.M.: Retrieval evaluation with incomplete information. In: SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2004) 25–32
19. Silverstein, C., Henzinger, M., Marais, H., Moricz, M.: Analysis of a Very Large AltaVista Query Log. Technical Report 1998-014, Digital SRC (1998) http://gatekeeper.dec.com/pub/DEC/SRC/technical-notes/abstracts/src-tn-1998-014.html.