Utilizing Supporting Evidence to Improve Dynamic Requirements Traceability

Jane Cleland-Huang, Raffaella Settimi, Chuan Duan, Xuchang Zou
Center for Requirements Engineering
DePaul University
{jhuang, rsettimi, duanchuan, xzou}@cs.depaul.edu

Abstract

Requirements traceability provides critical support throughout all phases of a software development project. However practice has repeatedly shown the difficulties involved in long-term maintenance of traditional traceability matrices. Dynamic retrieval methods minimize the need for creating and maintaining explicit links and can significantly reduce the effort required to perform a manual trace. Unfortunately they suffer from recall and precision problems. This paper introduces three strategies for incorporating supporting information into a probabilistic retrieval algorithm in order to improve the performance of dynamic requirements traceability. The strategies include hierarchical modeling, logical clustering of artifacts, and semi-automated pruning of the probabilistic network. Experimental results indicate that enhancement strategies can be used effectively to improve trace retrieval results thereby increasing the practicality of utilizing dynamic trace retrieval methods.

1. Introduction

The importance of requirements traceability for supporting critical software engineering activities such as requirements validation and change management is broadly recognized. However typical industrial practices in which traceability matrices are manually constructed and maintained tend to be costly to implement and are therefore perceived by many organizations to be financially non-viable [9,15]. To address this problem several researchers have investigated the use of dynamic retrieval methods to automate the process of generating traceability links [1,11,12,18]. These approaches are generally based on information retrieval methods that link artifacts according to the occurrence of terms in both the requirement and set of searchable documents or on the grammatical structuring of the terms [19]. Results from these approaches are promising because they clearly demonstrate the feasibility of replacing traditional trace methods with dynamic ones, but unfortunately they also suffer from precision problems.

In the trace retrieval problem it is important for the user to identify all relevant artifacts. If a query returns 70% of the critical links but fails to find the remaining 30%, then the query could be ineffective in supporting impact analysis, and a critical side effect of a proposed change could go unnoticed [18]. To counter this difficulty, trace retrieval strategies must favor recall over precision, where recall measures the number of correctly retrieved documents out of the entire set of correct documents, and precision measures the number of correctly retrieved documents out of the set of retrieved documents. As these metrics trade-off against each other, the decision to favor recall typically results in lower precision levels.

Prior work in this area has shown that at recall levels close to 90%, precision varies from 10% to 45% [1,7,12,18]. Although these results may initially seem poor, they actually can drastically reduce the amount of work an analyst needs to perform in order to conduct a trace. A manual trace that may previously have required a search through hundreds of artifacts to find the impacted ones, could now require an evaluation of only a fraction of those documents. Although this is no silver bullet, it does provide a significant and viable improvement on current practice [6,24].

This paper describes several techniques for incorporating additional information in order to improve trace retrieval results. An objective function was established for the retrieval algorithm to achieve recall levels at 90% or higher and precision at 20% or higher on all datasets. At these precision levels an analyst would need to evaluate an average of five retrieved documents in order to find one correct one. In addition to evaluating the overall precision for all feasible queries, the variation in precision for individual queries is also considered. Huffman-Hayes et al referred to a similar concept as believability [12] and proposed a series of metrics for its measurement. We focus on achieving consistent precision

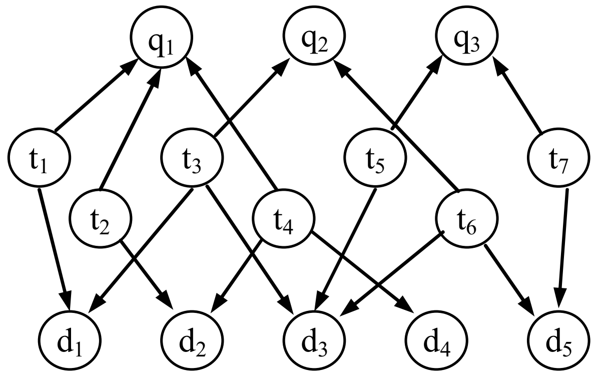


Figure 1. Directed acyclic graph for a probabilistic network retrieval model

across multiple queries, so that the analyst will have a more consistent expectation about the behavior of the tool.

Three different enhancement strategies based on *hierarchical modeling*, *artifact clustering*, and *semi-automated graph pruning* are discussed. In section 2, a basic probabilistic network algorithm is described to provide a baseline against which the three enhancements are compared. This algorithm was initially utilized in our previous work to retrieve non-functional goals that were potentially impacted during a design change [7]. Section 3 explains the importance of enhancement strategies and introduces the datasets used to evaluate the techniques described in this paper. Sections 4, 5, and 6 then describe the three enhancement strategies and report experimental results from applying each technique against the three different data sets. Section 7 provides an analysis of the results and a discussion of underlying project characteristics that are conducive to dynamic requirements trace retrieval.

2. The basic retrieval process

The basic retrieval algorithm uses a probabilistic network model to evaluate the relevance of a document to a specific query. The probabilistic network retrieval model based on prior work by Wong and Yao [21,22] consists of a graphical and a quantitative component. The graphical component is a directed acyclic graph (DAG) *G(V,E)* where the nodes *V* are random variables, and the arcs *E* link pairs of nodes and represent associations between variables. The DAG encodes the model's conditional independence assumptions among the variables [13]. The model assumes a probability distribution over the variables V that describes the interdependencies among the variables. Figure 1 provides an example of a DAG for a probabilistic network retrieval model. The model is defined over three sets of variables (or nodes): a set of *n* documents $\{d_1, d_2, \ldots, d_n\}$, a set of *k* terms $\{t_1, t_2, \ldots, t_k\}$ called index terms or keywords, and a set of *m* queries $\{q_1, q_2, \ldots, q_m\}$. The set of index terms $\{t_1, t_2, \ldots, t_k\}$ is derived from the document collection during a preprocessing step, in which non-useful words such as conjunctions, adverbs, and pronouns are eliminated and the remaining words are stemmed to their common grammatical roots by removing prefixes and suffixes [8].

As shown in Figure 1, the probabilistic model assumes the existence of arcs only between terms and documents, and between terms and queries. The DAG contains an arc from a document node to a term node, whenever the term belongs to the document. Similarly for queries, arcs are drawn between a query and each term contained in the query. Note that the graph does not contain direct arcs between queries and documents, since we assume that documents are conditionally independent on the queries given the index terms. In other words, the relationship between a document and a query is assumed to depend only on the terms co-occurring in both the document and the query. For computational reasons, index terms are assumed to be pair-wise disjoint.

The probability model is defined over a concept space *U*, called *Universe of Discourse*, in which terms $\{t_1, t_2, \ldots, t_k\}$ represent the singleton concepts, and each document d_i and each query q_j are random variables that represent propositions within the concept space. The probability $pr(d)$ of a proposition is interpreted as the degree of coverage of *U* by the knowledge contained in proposition *d*. The probability $pr(d_j|q)$ of a given document being linked (or relevant) to a specific query is defined as the degree of coverage for document d_j provided by *q*.

The conditional independence assumptions of the probabilistic network model allow us to compute the conditional probability $pr(d_j|q)$ of each potential document/query link as follows:

$$pr(d_j \mid q) = \left(\sum_{i=1}^{k} pr(d_j \mid t_i)\, pr(q, t_i) \right) / \, pr(q)$$

The conditional probabilities on the right-hand side of the equation can be estimated from the frequencies of occurrence of term t_i in the document d_j or in the query *q*.

Thus for each term t_i in d_j, the probability $pr(d_j \mid t_i) = \dfrac{freq(d_j, t_i)}{\sum_k freq(d_j, t_k)}$ represents the degree of information provided by t_i to describe the concept d_j, and can be assumed to be directly proportional to the frequency of occurrence of t_i in d_j in relation to the size of the document. In other words, this formula calculates the number of times a term is used in proportion to the size of the document.

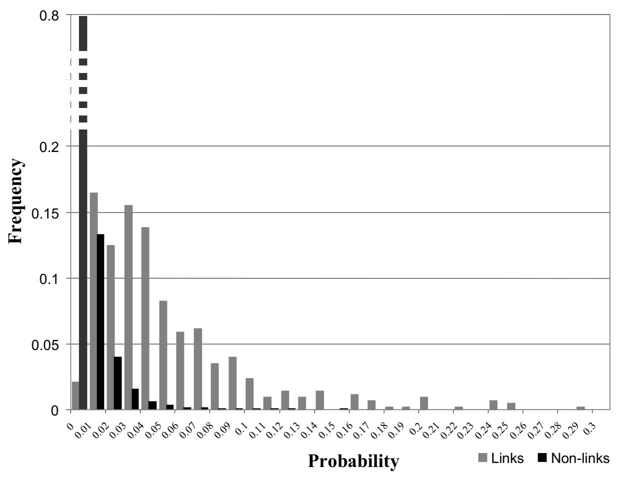


Figure 2. Probability distribution of links and non-links

The probability $pr(q,t_i)$ is computed as $pr(q,t_i) = \frac{freq(q,t_i)}{n_i}$ where n_i is the number of documents in the collection containing term t_i. This formula therefore considers the number of times the term is used in the query in proportion to the number of documents in which it occurs.

The probability of a query q is computed as $pr(q) = \sum_i pr(q,t_i)$, using simple marginalization techniques. The probability $pr(q,t_i)$ can be interpreted as the degree of relevance of the term t_i to describe the query concept q. In our formulation such probability value is inversely proportional to the number of documents containing the index term. Thus less frequent index terms correspond to larger probability values for $p(q,t_i)$, since words that appear in fewer documents are considered to be more informative in defining the relevance of a document to the query. This is a typical assumption in information retrieval algorithms such as the vector space model algorithm that use tf-idf ranking strategies [17].

Given a query q, the probabilistic retrieval tool computes the probability $pr(d_j|q)$ for each document d_j in the specified collection. The documents are ranked according to their degree of relevance to the query q, by sorting them in decreasing order of probability values $pr(d_j|q)$. A threshold value is typically established so that all documents with probability of relevance higher than the given threshold will be retrieved. In this paper, the threshold values were selected by optimizing the objective function "*maximize Recall + Precision, where Recall > T%",* where T% is a target recall chosen by the user. This is equivalent to finding the threshold value which maximizes both recall and precision while maintaining a sufficiently high recall level to effectively support requirements traceability. A training set can be used to select the threshold values [23]. In the experiments described later in this paper we identify the threshold value that returns recall of 90% and 95% and then report the precision achieved at each of these levels.

In the remainder of this paper the terms *true-link* and *non-link* are used in the following manner. A *true-link* represents a dependency relationship between a query and a document confirmed by consensus of knowledgeable developers. Conversely a non-link represents a query-document pair for which expert opinion has determined that no relationship exists. The term *candidate link* is also used to describe links returned by the tool but not yet evaluated by the analyst.

3. Enhancement Strategies

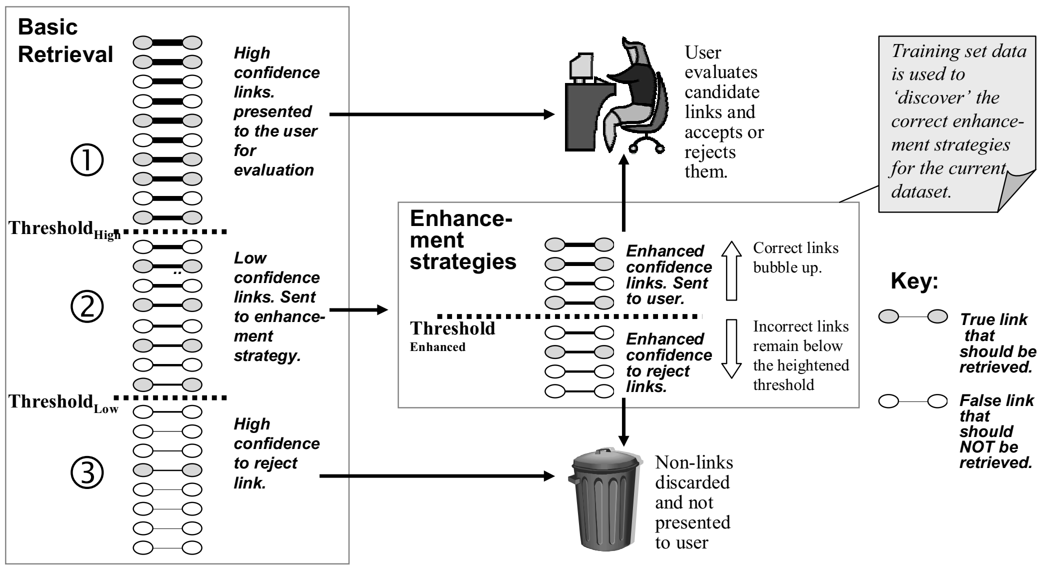
An analysis of the probability scores for true-links vs. non-links in a typical distribution such as that depicted for the Ice Breaker System in Figure 2, indicates that the majority of non-links are found at low probability values, while the majority of true-links occur at high values. An analyst could therefore have reasonably strong confidence that high probability values represent true links and that low probabilities indicate non-links. However at mid-level probability values it is much more difficult to differentiate between true-links and non-links. The enhancement strategies described in this paper are designed to address this problem by increasing the confidence that the artifact-pairs occurring within this mid-probability range are in fact either true-links or non-links.

In the trace retrieval problem the threshold is deliberately set low so that a high percentage of true-links will be recalled. Consequently, pairs of artifacts whose relevance scores appear below the threshold value can be safely assumed, with high degree of confidence, to be non-links. Pairs of artifacts whose relevance scores are just above the threshold will be assumed to be candidate links with very low confidence. As illustrated in Figure 3, it is therefore these pairs of artifacts that are sent to the hierarchical and clustering enhancement strategies.

The three enhancement strategies described in this paper were initially conceived through analyzing the root causes of omission and commission errors that occurred in a previous experiment [18], and then hypothesizing about systematic approaches for correcting these errors without introducing a significant number of new errors.

3.1 Data Sets

This study focused on the retrieval of links between requirements and UML class diagrams in order to simulate the type of traceability performed during an impact analysis query. As many UML case tools such as Together®, by TogetherSoft™ [20] provide explicit traceability between design models and code, tracing to

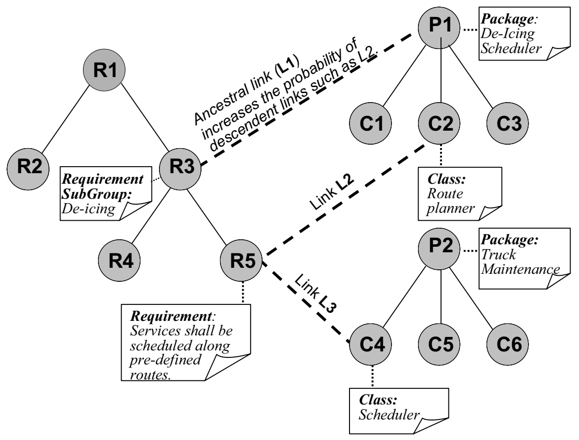


Basic Retrieval
①
High confidence links. presented to the user for evaluation
Threshold$_{High}$
②
Low confidence links. Sent to enhance-ment strategy.
Threshold$_{Low}$
③
High confidence to reject link.
User evaluates candidate links and accepts or rejects them.
Training set data is used to 'discover' the correct enhance-ment strategies for the current dataset.
Enhance-ment strategies
Enhanced confidence links. Sent to user.
Correct links bubble up.
Threshold$_{Enhanced}$
Enhanced confidence to reject links.
Incorrect links remain below the heightened threshold
Non-links discarded and not presented to user
Key:
True link that should be retrieved.
False link that should NOT be retrieved.

Figure 3. Enhancement strategies are aimed at low-confidence links

UML diagrams implies the ability to trace to code. Three data sets were selected for the experiments.

The Ice Breaker System (IBS) was initially described in [16] and enhanced with requirements mined from documents obtained from the public work departments of Charlotte, Colorado [3]; Greeley, Colorado [10]; and the Region of Peel, Ontario [14]. IBS manages de-icing services to prevent ice formation on roads, receiving inputs from a series of weather stations and road sensors within a specified district, and using this information to forecast freezing conditions and schedule dispersion of salt and other de-icing materials. It maintains maps of the district, plans de-icing, manages the inventory of de-icing materials; maintains, dispatches, and tracks trucks in real time; and issues and tracks work orders. The Ice Breaker system consists of 180 functional requirements, 72 classes, and 18 packages.



R1
R2
R3
R4
R5
Requirement SubGroup: De-icing
Requirement: Services shall be scheduled along pre-defined routes.
Ancestral link (L1) increases the probability of descendent links such as L2
Link L2
Link L3
P1
Package: De-Icing Scheduler
C1
C2
C3
Class: Route planner
P2
Package: Truck Maintenance
C4
C5
C6
Class: Scheduler

Figure 4. Contextual information obtained from the artifact hierarchy

The Event-Based Traceability (EBT) system, which was initially developed at the International Center for Software Engineering at the University of Illinois at Chicago [4,5], provides a dynamic traceability infrastructure based on the publish-subscribe scheme for maintaining artifacts during long-term change maintenance. It is composed of 54 requirements, 60 classes, and 8 packages.

Finally, artifacts in the third data set were reconstructed from the well documented Light Control system (LC) developed for the University of Kaiserslautern [2]. This system controls the lights in a building based upon user defined lighting schemes, building occupation, and current exterior illumination. Our version of this system consisted of 36 requirements, 25 classes, and 5 packages.

For each of these data sets UML diagrams were developed in Poseidon, exported to XMI and then parsed into the Poirot:Tracemaker tool. Requirements were entered directly into Poirot's database. For analysis purposes, a traceability matrix was constructed that identified the true-links that the retrieval algorithm should attempt to retrieve. This matrix was used to evaluate the performance of each of the enhancement strategies in terms of recall and precision metrics. The following sections describe each enhancement and report on experimental results.

4. Hierarchical Enhancement

Many artifacts are arranged in a hierarchical format in which the words used to name and describe the higher level artifacts capture the general meaning of their lower-level components. As an example, consider the situation

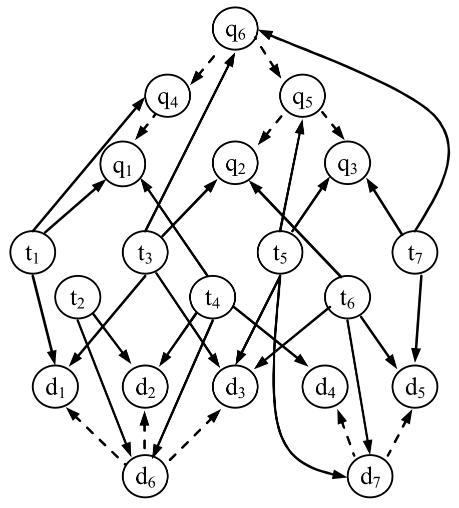


Figure 5. Directed acyclic graph G(V,E) for the hierarchical probabilistic network model

in Figure 4, in which requirements are labeled R1-R5 and classes and packages are labeled C1-C6 and P1-P2 respectively. R3 represents a subgroup of requirements entitled "De-Icing". One of the requirements in this group, R5, states that "Services shall be scheduled along pre-defined routes". In this case, the hierarchical information contained in R3, provides the necessary context in which to understand that R5 actually describes de-icing services. Similar contextual information can be found on the document side of the hierarchy such as in section titles for design documents or packages containing classes. At the leaf nodes, links are generated between R5 and C2 (Link L2), and between R5 and C4 (Link L3). In fact, the L3 link is incorrect because the Scheduler class (C4) actually refers to scheduling trucks for maintenance and is not related to de-icing. In contrast link L2 represents a true link. The basic retrieval algorithm retrieves both of these links at similar probability values and is unable to filter out the incorrect link.

In contrast, the hierarchical algorithm strengthens link L2 according to the probability of the ancestral link (L1) that is generated between R5's and C2's ancestors. By attempting to strengthen the probability of true-links, this approach enables the threshold value to be raised and therefore improves overall precision by filtering out additional non-links.

This concept is incorporated into the probabilistic retrieval model, by including the interrelationships that occur between ancestral documents and queries in the model. The DAG associated to this new model is depicted in Figure 5, and has arcs linking artifacts that are directly related within an artifact hierarchy. These arcs are shown as dashed lines. For instance, if a class diagram belongs to a specific package, the node representing the class diagram will be directly linked to the package node. Similarly for requirements, a node corresponding to a higher level requirement module will be directly linked to sub-requirements nodes that belong to that module.

The additional dependencies introduced in the model are taken into account when computing the probabilities associated to the variables. In the hierarchical model, the conditional probability $pr(d_j|q)$ of a potential link between a document d_j and a query q can be computed as

$$pr(d_j \mid q) = \Big(\sum_{g \in pa_D(d_j)} \sum_i pr(d_j, g \mid t_i) pr(q, t_i) \Big) / pr(q)$$

where $pa_D(d_j)$ denotes the set of parents of d_j and contains all document nodes that have arcs pointing to the d_j node. The probability $pr(d_j,g|t_i)$ for any parent g of d_j is computed as follows:

$$pr(d_j, g \mid t_i) \propto \frac{freq(d_j, t_i)}{\sum_k freq(d_j, t_k)} + \alpha_D \sum_{g \in pa_D(d_j)} \frac{freq(g, t_i)}{\sum_k freq(g, t_k)}$$

The probability $pr(q,t_i)$ is defined as:

$$pr(q, t_i) \propto \left(freq(q, t_i) + \alpha_Q \sum_{h \in an_D(q)} pr(q \mid h) freq(h, t_i) \right\} / n_i$$

The weights α_D and α_Q in the formulas above are chosen in (0,1). Initial experimentation with the weighting factors indicated that both α_D and α_Q should be set equal to 0.5.

The term $an_Q(q)$ denotes the set of query nodes that are ancestors of q. For any ancestor node of q, say h, there exists a path of directed arcs from h to q. The conditional probability $pr(q \mid h)$ depends on the proximity between q and its ancestor h according to the hierarchical topology, and is assumed to decrease with the distance between q and h. This is based on the assumption that ancestors that are closer to the query q are likely to provide stronger information about the query, compared to other ancestors that lay farther away in the query hierarchy. In our experiments we compute the conditional probability as

$$pr(q \mid h) = 1/(A+1)$$

where A is the number of arcs in the shortest path between the ancestor node h and the node q.

For experimental purposes, the hierarchical approach was applied first to the entire data set, and secondly as an enhancement to the basic approach by implementing it against only the smaller band of lower confidence links depicted in Figure 3. For all experiments reported in this paper, the actual low confidence band was discovered through trial and error. Future work will extend our current automated approach for detecting optimal threshold values.

4.1 Results from Hierarchical Enhancement

Results from the hierarchical enhancement are shown in Table 1. For the IBS dataset precision showed little or no improvement when the hierarchical approach was applied to the entire dataset however it returned overall gains in precision of 8.84% and 11.29% at recall levels of

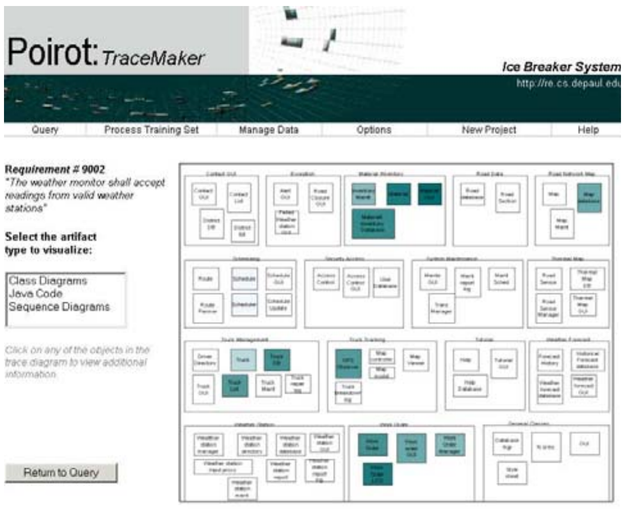
Table 1. Results from hierarchical enhancement showing recall goals at 90% and 95%

Retrieval strategy	IBS		EBT		LC	
	Recall	Precision	Recall	Precision	Recall	Precision
Basic	90.47%	20.43%	90.97%	17.75%	90.11%	37.61%
	95.01%	16.81%	95.13%	16.36%	93.40%	32.57%
Hierarchical applied to entire dataset	90.25%	20.02%	89.58%	12.02%	90.11%	31.78%
	95.01%	18.14%	95.14%	12.18%	95.6%	30.53%
Basic + Hierarchical applied to low confidence links	90.48%	31.72%	90.87%	18.30%	90.11%	36.77%
	95.69%	25.65%	95.83%	17.42%	95.6%	30.53%

95% and 90% respectively when it was applied in combination with the basic approach to only the band of lower confidence links. This confirmed our conjecture that enhancements would be more effectively applied against the band of prospective links returned at mid-level ranges of probability.

To put this into perspective, precision levels of around 20% returned by the basic retrieval algorithm for the IBS dataset, would require an analyst to look through an average of five links in order to find one good one, while a precision level of around 30% returned by the hierarchical algorithm, would require the analyst to look through an average of approximately three links in order to find one good one.

For the EBT data sets, minimal improvements in precision were observed, with an increase of approximately 1% at both 90% and 95% recall levels. An observation of the EBT dataset showed that the hierarchical information, especially on the document side, was not that strong, and package names had not been created as meaningfully as they might have been. This observation confirms that strong hierarchical information is needed for this approach to be effective.



Requirement # 9002
"The weather monitor shall accept readings from valid weather stations"

Select the artifact type to visualize:

Click on any of the objects in the trace diagram to view additional information

Figure 6. Visual portrayal of a requirement displayed in Poirot:Tracemaker

Similarly, the LC dataset did not return any improvement in precision through the hierarchical enhancement. However it is worth noticing that in the LC data set the basic algorithm was unable to retrieve more than 93.40% of correct traces. This was due to the fact that some of the correct document/query links could not be retrieved because the document and query pairs have few or no common terms and therefore the associated probability values were zero. However, using a combination of the basic and the hierarchical approach enabled retrieval up to 95.6% of the corrected links with 30.53% precision level. The hierarchical approach achieved this higher recall by utilizing additional structural information in order to capture links missed by the basic algorithm.

5. Clustering Enhancements

Clustering enhancements are based on the premise that links tend to occur in clusters. This is clearly depicted in Figure 6, which visualizes the generated links from an individual requirement to the entire set of classes in the IBS system. The visual display is part of the Poirot:TraceMaker tool and depicts generated links shaded according to the strength of their probability values. Darker shades represent stronger probability values, while lighter shades represent lower values. This diagram clearly illustrates the clustering effect which is typical of many queries.

Based on our initial observations of omission and inclusion errors [18], we hypothesized that if a link exists between a query and a document, and if that document is part of a logical cluster of documents, then there would be a higher probability that additional links should exist between the same query and other documents in that cluster. A similar observation can be made for links between individual documents and clusters of queries. These concepts are named *document side clustering* and *requirement side clustering* respectively and are illustrated in Figure 7. This approach differs from the previous enhancement strategy of hierarchical information because the enhancement is based on sibling artifacts rather than ancestral information. Furthermore, clusters can also be formed in other ways such as from logical groupings of work products such as a UML class and its related code and test cases. For the purposes of these experiments requirements-side clusters were formed from groups of lower level requirements and document-side clusters from classes bundled together into packages.

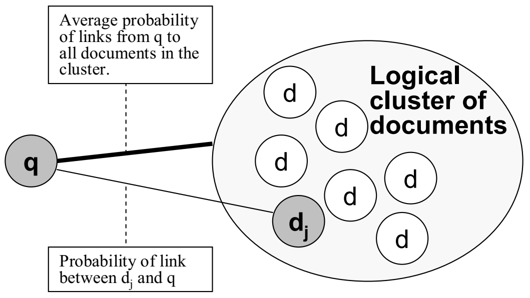
This concept was initially tested by post-processing the retrieved links. The following steps described in terms of document side clustering were implemented, where the low confidence band is defined between $Threshold_{high}$ and $Threshold_{low}$ and the results of the basic algorithm are only accepted for those probabilities returned greater than $Threshold_{high}$. Low confidence links were retrieved only if the average probability between the query and document's cluster was greater than $Threshold_{Enhanced}$. The algorithm is depicted below and can be applied to query side clustering by reversing the document and query terms in the algorithm.

```
If $Pr(d_j|q)_{basic}$ > $Threshold_{High}$ then
    retrieve artifact pair $d_j$,q.
else
    if $Pr(d_j|q)_{basic}$ > $Threshold_{Low}$ and
        $\Sigma$ Pr($d_k$|q)]/$N_k$ >
        $Threshold_{Enhanced}$ then
        retrieve artifact pair $d_j$,q
    else
        reject artifact pair $d_{j,q}$
```

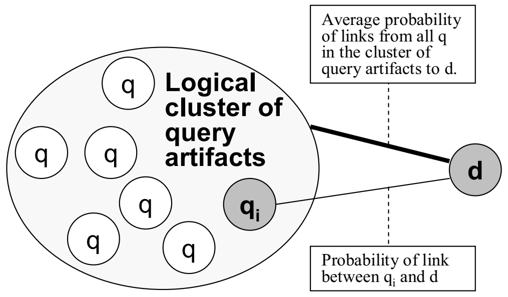
where N_k = the number of documents in cluster k of which d is a member.

5.1 Results from Clustering Enhancement

Results from the clustering enhancement, displayed in Table 2, returned the least overall improvements of the three techniques, with no significant improvements observed in either the EBT or the LC data. For the IBS system minimal improvements in recall and precision were observed when document-side clustering was applied against the entire document set, but significant improvements were observed when it was applied to the subset of low confidence links. At approximately 95% and 90% recall, document-side clustering resulted in improvements in precision of 10.38% and 9.6% respectively. Interestingly query-side clustering led to general improvements of approximately 5% and 8% when applied against the entire document set, but this phenomenon did not hold over the other data sets.


Average probability of links from q to all documents in the cluster.
Logical cluster of documents
q
d
d_j
Probability of link between d_j and q

a. Document side clustering


Average probability of links from all q in the cluster of query artifacts to d.
Logical cluster of query artifacts
q
q_i
d
Probability of link between q_i and d

b. Query side clustering

Figure 7. Clustering

Table 2. Results from the clustering enhancement showing recall goals at 90% and 95%

Retrieval strategy	IBS		EBT		LC	
	Recall	Precision	Recall	Precision	Recall	Precision
Basic	90.47%	20.43%	90.97%	17.75%	90.11%	37.61%
	95.01%	16.81%	95.13%	16.36%	93.40%	32.57%
Document side clustering on entire dataset	90.1%	21.1%	89.77%	14.99%	90.11%	37.61%
	95.5%	19.5%	95.45%	11.00%	93.40%	32.57%
Basic + Document side clustering on low confidence links	90.00%	30.20%	89.77%	14.99%	90.11%	37.61%
	95.20%	27.68%	95.45%	11.00%	93.40%	32.57%
Query side clustering on entire dataset	90.5%	29.5%	89.77%	14.99%	90.11%	37.61%
	95.00%	22.00%	95.45%	11.00%	93.40%	32.57%
Basic + Query side clustering on low confidence links	90.50%	30.0%	89.77%	14.99%	90.11%	38.3%
	95.50%	22.00%	95.45%	11.00%	93.40%	32.57%

6. Graph Pruning Enhancement

Unlike the previous two strategies, the graph pruning enhancement is not intended to improve precision across all queries and documents, but is focused on particular areas in which the retrieval precision is particularly poor. An example of a localized precision problem is found in the IBS system where the word “schedule” was used to refer to both de-icing schedules and truck maintenance schedules. As a result, any query that included the word “schedule” tended to return artifacts from both the de-icing and the truck maintenance domain. Precision for these queries was therefore much lower than for average queries and could be improved by prohibiting links such as those between de-icing schedule requirements and truck maintenance classes.

The graph pruning process described in this paper utilizes initial decisions made by the analyst against the training set data, in order to discover where to place constraints and to improve the precision of problematic areas. Rather than attempting to establish individual constraints which would take excessive effort, our approach creates constraints between groups of requirements and groups of documents.

As with the case of the clustering enhancement, pruning results were compared against the results of the basic algorithm. For this initial series of experiments query-side groups were established from the lowest level requirement sub-trees, and document-side groups were again formed from packages and their component classes.

Constraints could be established manually by the analyst however this would require an excessive amount of effort in even a medium sized project. To understand the scope of this problem, consider the IBS project in which artifacts were assembled into 19 requirements groups and 18 packages, representing 342 potential pairs of query-document groupings and 342 potential decision points for the analyst to consider! The constraint creation process was therefore fully automated in our study, although we believe that results could potentially be improved through user feedback.

The approach utilizes results obtained from an initial training set in order to automate the process of identifying the subset of query-document groupings that are likely to spawn false links. Constraints were established when the following two conditions held. First, an analyst had evaluated one or more links between the two groups and had rejected all of them. Second, the basic retrieval algorithm had actually generated candidate links between the two groups. The second condition minimizes the number of generated constraints which could reduce the analyst’s work if they decided to evaluate the links and offer feedback. As this experiment was conducted without user feedback, it simulated the situation in which the user accepted all the constraints proposed by the algorithm. Two variations of the experiment were conducted. In the first case the probability of links between constrained groupings was set to zero meaning that these links were completely disallowed, while in the second case the probability of constrained links was reduced by 50%.

6.1 Results from Graph Pruning

Initial observations indicated that the size of the selected training set has a direct impact upon the results. A smaller training set requires less effort on the part of the analyst but tends to return results at lower levels of recall and higher levels of precision. As the training set size increases, the actual knowledge of query-document links provides greater insight into the feasible and infeasible links and results in increased recall while maintaining higher levels of precision. This suggests that automated pruning should only be used once sufficient knowledge has been obtained from analyst feedback. Although a more formal usability study is needed to determine how long it takes a user to analyze a training set and to identify the optimal balance between effort and accuracy of the retrieval results, for this study training set sizes of 20%, 30%, and 50% were selected for the IBS, EBT, and LC systems respectively. These training set ratios reflected the individual size of each data set and were designed to require a similar amount of effort on the part of the analyst for each project.

The results from the graph pruning reported in Table 3 were very promising, especially in light of the fact that

Table 3. Results from the pruning enhancement keeping thresholds at levels that returned 90% and 95% recall levels prior to pruning

Retrieval strategy	IBS (20% training set)		EBT (30% training set)		LC (50% training set)	
	Recall	Precision	Recall	Precision	Recall	Precision
Basic	90.47%	20.43%	90.97%	17.75%	90.11%	37.61%
	95.01%	16.81%	95.13%	16.36%	93.40%	32.57%
Probability values set at 50%	89.79%	24.40%	90.97%	20.25%	90.11%	40.59%
	95.01%	19.05%	95.14%	18.41%	93.41%	33.46%
Probability values set to zero	73.70%	33.16%	76.39%	26.13%	80.22%	49.32%
	77.78%	28.25%	80.55%	24.73%	82.42%	41.90%

user input was not elicited to evaluate the generated links. One of the first observations is that EBT results were finally within the objective range of 90% recall and 20% precision.

In the first experiment for which probabilities of constrained links were set to zero, the recall suffered dramatically in each dataset, dropping by approximately 10-15% in each case. Although precision also improved dramatically the recall levels below 90% were considered unacceptable. In the second set of experiments, for which probabilities of constrained links were set at 50%, the results indicated little or no reduction in recall yet marked improvement in precision. Several repetitions of the experiment all returned similar results.

Although the overall precision improvements for each dataset, depicted in Table 3 were less than 5% in each case, the graph-pruning enhancement was actually designed to achieve uniformity across individual queries, and so these results are also analyzed in terms of precision per query. Following pruning there was much less variation in the precision returned for each individual query. The average precision per query for IBS increased from 20.7% to 22.7%, for EBT from 10.8% to 23.7%, and for LC from 43% to 47%. These results do not include queries for which no actual links were identified in the traceability matrix. The most dramatic result was seen in the EBT system, which was the dataset that had been least responsive to either the basic retrieval algorithm or the hierarchical or clustering enhancements.

Figure 8 depicts the change in precision for individual queries following the pruning process. The graph shows only those queries that either gained in precision or suffered from reduced precision. In the IBS and LC datasets approximately 50% of the queries were unaffected, while in the EBT dataset less than 15% were unaffected. As the graph clearly depicts, a significant number of individual queries achieved improvements in precision, and in the case of the EBT dataset, approximately 6% of queries exhibited precision improvements greater than 60%. These results are significant because improving consistency in precision across multiple queries can increase the analyst's trust in the reliability of the retrieval tool.

7. Analysis of the results

The three enhancement strategies described in this paper demonstrated varying abilities to enhance the retrieval of requirements traces. One of the clearest observations is that the IBS dataset was more conducive to the enhancements than the other two datasets. Analysis of this dataset showed that the artifacts were structured into meaningful groupings that provided useful contextual clues. This was the only dataset that improved under all three enhancement strategies.

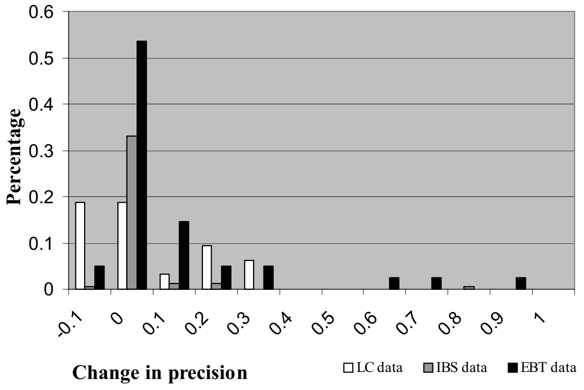


Figure 8. Gain in precision for individual queries achieved from pruning

The LC dataset exhibited the highest levels of recall and precision under the basic algorithm but was unresponsive to clustering enhancements. Analysis of the way artifacts were structured revealed low compatibility of organizations on the query and document side of the hierarchies. Requirements were organized more in terms of roles and locations, while the design documents were organized more in terms of functionality of the lighting features regardless of roles and locations. Stronger corollaries between both hierarchies might lead to better retrieval results from the enhancement strategies. However, it should be noted that although in this case, the hierarchical enhancement did not significantly improve precision in general, it did enable recall to reach the target level of 95% by recalling several previously missed artifacts.

Finally, the EBT system returned both the lowest basic results and showed the least response to the hierarchical and clustering enhancement strategies. However, this dataset was extremely responsive to the pruning technique and showed marked improvements in precision per query after constraints had been set between various artifact groupings. An analysis of the EBT dataset showed that a number of terms in the artifacts had been given dual meanings, which resulted in retrieval of an excessive number of unwanted artifacts. In many cases these unwanted artifacts were filtered out by the pruning process.

7.1 Conclusions

An analysis of the improvements provided by each enhancement strategy indicated significant overlap between the hierarchical and clustering techniques. A further analysis of the IBS dataset indicated that while maintaining existing recall levels previously established as optimal to achieve 95% recall for each individual algorithm, 87% of true-links were retrieved by both algorithms, 8% by only the hierarchical algorithm, 3% by only the clustering algorithm, and 2% were missed by both. Additional work is needed to determine how the hierarchical and clustering algorithms could be applied synergistically.

Little overlap was observed between the pruning technique and the other algorithms. This is because the hierarchical and clustering methods had a more general impact on the overall precision of the system links while the pruning enhancement specifically identified and addressed problematic queries.

Analysis of these results suggests that certain datasets may be more responsive to dynamic trace retrieval. The use of a project glossary and systematic hierarchy containing well defined terms can improve dynamic trace retrieval for the project. During the early stages of a project once an initial set of requirements and design artifacts have been developed it may be helpful to develop a small trace matrix and evaluate the project's ability to support dynamic traceability [23]. Currently the Poirot: Tracemaker tool implements both the basic and hierarchical retrieval algorithms, and as a result of the experiments reported in this paper, our future work will incorporate a more sophisticated pruning process into the probabilistic algorithm.

Acknowledgments

The work described in this paper was partially funded by NSF grant CCR-0306303. We also acknowledge the contributions of DePaul students Rick Strahl, Oussamma BenKhadra, and Cathy Herzon for their assistance in developing the Poirot:Tracemaker tool.

References

[1] G. Antoniol, G. Canfora, G. Casazza, A. De Lucia and E. Merlo. Recovering Traceability Links between Code and Documentation, *IEEE Trans. on Software Engineering*. Vol. 28, No. 10, 2002, pp. 970-983.

[2] E. Borger and R. Gotzhein, "Requirements Engineering Case Study 'Light Control,'" http://rn.informatik.uni-kl.de/recs, 2002.

[3] Charlotte Dept of Transportation, N.C, Information on road management services, http://www. charmeck.org/Departments/Transportation/Home.htm

[4] J. Cleland-Huang, C.K.Chang, and J.Wise, "Automating Performance Related Impact Analysis through Event Based Traceability", *Requirements Engineering Journal, Springer-Verlag*, Vol. 8, No. 3, Aug. 2003, pp. 171-182.

[5] J .Cleland-Huang, C.K.Chang, and M. Christensen, "Robust Requirements Traceability for Handling Evolutionary Change", *IEEE Trans. on Software Engineering*, Vol. 29, No. 9, Sept. 2003, pp. 796-810.

[6] J.Cleland-Huang, G. Zemont, and W. Lukasik, "A Heterogeneous Solution for Improving the Return on Investment of Requirements Traceability", *International Conference on Requirements Engineering,* Kyoto, Japan, 2004, pp. 230-239.

[7] J. Cleland-Huang, R. Settimi, O. BenKhadra, E. Berezhanskaya, S. Christina, "Goal-Centric Traceability for Managing Non-Functional Requirements", *International Conference on Software Engineering*, St. Louis, USA, May 2005, pp. 362-371.

[8] Frakes W.B., and Baeza-Yates, R., *Information retrieval: Data structures and Algorithms*, Prentice-Hall, Englewood Cliffs, NJ, 1992.

[9] O. Gotel, and A. Finkelstein, "An Analysis of the Requirements Traceability Problem," *Proceedings First Int'l Conf. Requirements Eng.,* 1994, pp. 94-101.

[10] Greeley City, Colorado, Information on road management, http://www.ci.greeley.co.us

[11] J. Huffman Hayes, A. Dekhtyar, and J. Osborne, "Improving Requirements Tracing via Information Retrieval", *IEEE International Requirements Engineering Conference,* Monterey, CA, Sept. 2003, pp.138-150.

[12] J. Huffman-Hayes, A. Dekhtyar, S. Karthikeyan Sundaram, S. Howard, "Helping Analysts Trace Requirements: An Objective Look", *International Requirements Engineering Conference*, Kyoto, Japan. Pp. 245-259.

[13] Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann, San Mateo, CA.

[14] Peel Region, Information on road management, http://www.region.peel.on.ca

[15] B. Ramesh, and M. Jarke, "Toward Reference Models for Requirements Traceability", *IEEE Trans.. on Software Engineering,* Vol. 27, No. 1, Jan 2001, pp. 58-92.

[16] S. Robertson, J. Robertson, *"Mastering the Requirements Process"*, Addison-Wesley, 1999.

[17] G. Salton and C. Buckley. Term weighting approaches in automatic retrieval. *Information Processing and Management*, 24(5), pp. 513-523, 1988.

[18] R. Settimi, J. Cleland-Huang, O. BenKhadra, J. Mody, W. Lukasik, and C. DePalma, C., "Supporting Change in Evolving Software Systems through Dynamic Traces to UML", *IEEE International Workshop on Principles of Software Evolution*, Kyoto, Japan, Sept. 2004, pp. 49-54.

[19] G. Spanoudakis, "Plausible and Adaptive Requirement Traceability Structures", Proc. of the 14th International Conference on Software Engineering and Knowledge Engineering, Ischia, Italy, 2002, pp.135-142.

[20] TogetherSoft™, http://www.togethersoft.com

[21] S.K.M.Wong, and Y.Y. Yao, "A probabilistic inference model for information retrieval" *Information Systems*, Vol. 16, No. 3, 1991, pp. 301-321.

[22] S.K.M. Wong and Y.Y. Yao, "On Modeling Information Retrieval with Probabilistic Inference", *ACM Transactions on Information Systems*, Vol. 13, No. 1, 1995, pp.38-68.

[23] X. Zou, R. Settimi, J. Cleland-Huang, C. Duan, "Empirical Study of a Probabilistic Network Approach for Retrieving Traceability Links", *Technical Report # 2005-003, School of Computer Science, Telecommunications, and Information Systems, DePaul University*, (March 2005).

[24] G. Zemont, *"Towards Value Based Requirements Traceability"*, Master Thesis, DePaul University technical report TR05-011, 2005.