Baselines in Requirements Tracing

Senthil Karthikeyan Sundaram
Computer Science Department
University of Kentucky
skart2@uky.edu

Jane Huffman Hayes
Computer Science Department
University of Kentucky
hayes@cs.uky.edu

Alexander Dekhtyar
Computer Science Department
University of Kentucky
dekhtyar@cs.uky.edu

ABSTRACT
We summarize the results of our requirements tracing work to date, focusing on our empirical results with open source datasets. Specifically, we describe the problem of after-the-fact requirements tracing for Verification and Validation (V&V) analysts, we provide a brief overview of Information Retrieval methods we have applied as well as measures used to evaluate them, we describe our tracing tool, and we present the results of a number of empirical studies. Two of the open source datasets that we have used are available to the research community at http://promise.site.uottawa.ca/SERepository/.

Categories and Subject Descriptors
D.2.1 **[Requirements/Specifications]**: Tools
D.2.5 **[Testing and Debugging]**: Tracing
H.3.3 **[Information Search and Retrieval]**: relevance feedback, retrieval models

General Terms
Experimentation, Human Factors, Verification.

Keywords
Requirements tracing, traceability, information retrieval, relevance feedback.

1. INTRODUCTION
Ensuring that requirements have been addressed at each phase of the lifecycle is an important part of the verification and validation of a system. In addition, the ability to assess change impact is an important activity that we undertake in software engineering. In order to achieve these two important tasks, a detailed, accurate Requirements Traceability Matrix (RTM) is needed. Unfortunately, such RTMs are often not built to the appropriate level of detail, or are not maintained, thus requiring "after-the-fact" tracing. Building and maintaining an RTM can be tedious, cumbersome, and error-prone. Current approaches to after-the-fact tracing have many shortcomings, detailed in [4, 5].

Fortunately, requirements tracing can be represented as an Information Retrieval (IR) problem. Requirements tracing, as described in [5], consists of document parsing, candidate link generation, candidate link evaluation, and traceability analysis. Our work focuses on candidate link generation. We have found that IR methods can be used to quickly generate candidate links that are accurate (as measured by: the percentage of actual matches that are found (recall), and the percentage of correct matches as a ratio to the total number of candidate links returned (precision)). We have implemented these methods in a tool, **RE**quirements **TR**acing **O**n target (RETRO). We have evaluated the methods using two open source NASA datasets, Moderate Resolution Imaging Spectroradiometer (MODIS) [7, 9], and CM-1 [8]. This paper summarizes the results obtained in our experiments. Some of the results we mention have been reported previously in [5], while other results are reported here for the first time.

The paper is organized as follows: Section 2 presents an overview of the methods implemented in RETRO and the measures used to evaluate the methods. Section 3 discusses the datasets used for evaluation. Section 4 presents our tool. Section 5 discusses experimental setup while Section 6 shows the results obtained from evaluation. Finally, conclusions and future work are presented in Section 7.

2. USING INFORMATION RETRIEVAL FOR REQUIREMENTS TRACING
In this section, we present a short description of information retrieval, the IR methods applied, and the measures used to evaluate the methods.

2.1 Information Retrieval
As discussed in [5], the problem of requirements tracing can be viewed as determining, for each pair of elements from high- and low-level documents, whether the elements match. The problem of Information Retrieval can be stated as follows: given a document collection and a query, determine the set of documents from the collection that are similar to the query [1]. We can see that there is a striking similarity between requirements tracing and IR. In the case of requirements tracing, the high level elements or requirements act as queries and the low level elements act as the document collection.

PROMISE'05 May 15, 2005 St. Louis, Missouri, USA.

2.2 Information Retrieval Methods Used in RETRO

We have implemented and evaluated a variety of IR methods. In this paper, we discuss *tf-idf vector retrieval*, *tf-idf retrieval with simple thesaurus, and Latent Semantic Indexing (LSI)*. Each method has been implemented with an appropriate analyst feedback processing method [1].

2.2.1 Tf-Idf Model

A document *D* in the collection is represented as a vector $d=(w_1,..,w_N)$ of keyword weights, based on the vocabulary $V=(k_1,..k_N)$ of the entire collection. The weight, w_i, is calculated as the product of the *term frequency (tf)*, the number of times k_i occurs in *D,* and the *inverse document frequency (idf),* determined by the number of documents in the collection in which k_i occurs. Similarly, the query is also converted into a vector, $q=(q_1,..,q_N)$, and the similarity (also called relevance) *sim(d,q)* is computed as the cosine of the angle between the vectors d and q:

$$sim(d,q) = \cos(d,q) = \frac{\sum_{i=1}^{N} w_i \cdot q_i}{\sqrt{\sum_{i=1}^{N} w_i^2 \cdot \sum_{i=1}^{N} q_i^2}}.$$

2.2.2 Tf-Idf + Simple Thesaurus

This method is an extension of the Tf-Idf model with a simple thesaurus of keywords and phrases. A simple thesaurus T is a set of triplets <t,t',α>, where t and t' are matching thesaurus terms and α is the similarity coefficient between them (e.g., <"fault", "error", 0.85>). Thesaurus terms can be either single keywords or key phrases – sequences of two or more keywords. In this method, the vocabulary of the document collection now contains all thesaurus terms that are key phrases (i.e., not single keywords). The similarity (relevance) formula is modified to account for the thesaurus matches as:

$$sim(d,q) = \cos(d,q) = \frac{\sum_{i=1}^{N} w_i \cdot q_i + \sum_{<ki,kj,\alpha_{ij}>\in T} \alpha_{ij}(w_i \cdot q_j + w_j \cdot q_i)}{\sqrt{\sum_{i=1}^{N} w_i^2 \cdot \sum_{i=1}^{N} q_i^2}}.$$

2.2.3 Latent Semantic Indexing (LSI)

LSI is a dimension reduction technique based on Singular Value Decomposition (SVD) of the term-by-document matrix that can be constructed by putting tf-idf vectors of all documents in a single matrix [3]. SVD transforms the original matrix into a product of two orthogonal matrices and a diagonal matrix of eigenvalues. By considering only the top k eigenvalues, we can obtain an approximation of the original matrix by a smaller matrix. Rows of the matrix can be compared to each other using the cosine similarity described above. For example, if L is a document-by-term weight matrix of dimension A x B, its SVD is written as L = TSD', where T is a matrix with orthogonal rows, D' is a matrix with orthogonal columns and S is a diagonal matrix of eigenvalues of L. We can trim the list of eigenvalues of L from rank(L) to a smaller number n and obtain a decomposition $L_n = TS_nD'$, where S_n is the diagonal matrix of size n x n with n largest eigenvalues of L on the diagonal. Rows of the matrix TS_n^2D' can be compared to each other using the cosine similarity as defined in Section 2.2.1. Use of the matrix TS_n^2D' instead of the original matrix L reduces the dimensionality of the document vectors from B to n. We have implemented LSI based both on tf-idf and tf-idf+thesaurus document collection matrices.

2.2.4 Relevance Feedback

To incorporate interactive work with analyst into RETRO, we have implemented relevance feedback for the IR methods studied. Relevance feedback works as follows: the analyst conveys to RETRO both positive (true link found) and negative (false positive found) information. The relevance feedback processor re-computes the vector q_{new} for the query q by adding to it positive information and subtracting negative information:

$$q_{new} = \alpha q + \left(\frac{\beta}{r}\sum_{d_j \in D_r} d_j\right) - \left(\frac{\gamma}{s}\sum_{d_k \in D_{irr}} d_k\right).$$

In the above formula, α, β and γ determine the importance of the original vector, positive information and negative information respectively. D_r is the set of documents deemed as relevant to the query q and D_{irr} is the set of documents deemed as irrelevant to the query q.

2.3 Measuring Effectiveness

This section discusses the primary and secondary measures used to evaluate tracing techniques. The effectiveness of IR techniques is typically evaluated using the primary measures of recall and precision as defined in [6]. In requirements tracing, recall measures if a technique was able to find all the high-low level requirement pairs that trace to each other, whereas precision signifies the number of extra pairs found by the technique that do not trace to each other. Secondary measures help to understand the structure of the candidate links better. In [6], we have developed and used a set of secondary measures. For brevity, we only examine Selectivity - the percentage of all possible high-low requirement pairs that is found in the candidate trace.

```
%
SDP3.3-2  L1APR03-I-2 L1APR01-I-2
%
SDP3.3-4  L1APR03-I-2 L1APR01-I-2 L1APR01-I-1
%
```

Figure 1. Sample answer set for MODIS dataset.

The DPU-BOOT CSC shall test and clear DRAM on power-on using the COLD_MEM_SIZE obtained from the SYSTEM_BLOCK.

Figure 2. Sample text for a requirement from CM-1 dataset.

2.3.1 *Evaluating Candidate Traces*

When evaluating candidate traces (whether computer- or human-generated), we want both precision and recall to be as high as possible. However, we use different thresholds for these two measures. Based on our observation, we note that a change from 10% to 20% in precision means that instead of 1 true link in 10, the trace contains 1 true link in 5, a savings of about 50% in terms of the number of links to verify for the analyst. We strive for high recall to prevent analysts from having to search for links not shown to them. This latter activity is much more time consuming and much less desirable as a task than simply vetting the results provided from an existing candidate trace. In addition, we want selectivity to be as low as possible (while keeping recall high). For recall, we consider results above 80% excellent, above 70% - good, and between 60% and 70% - acceptable. For precision, 20-30% is acceptable, 30-50% is good, and 50% and above – excellent.

3. DATASETS

Two NASA open source datasets were used to evaluate the IR techniques implemented in our tool, MODIS [7, 9] and CM-1 [8]. The MODIS dataset consists of 19 high level and 49 low-level requirements whereas the CM-1 dataset contains 235 high-level requirements and 220 design elements. ***We have manually traced both datasets and have verified the obtained traces.*** We refer to these as "answer sets" or "theoretical true traces." There were 41 and 361 true links found for the MODIS and CM-1 datasets, respectively. Figure 1 shows a subset of the answer set (true trace) for MODIS [7, 9]. The answer set contains the matching low level elements for each high level element, separated by a percentage symbol. For example, high level requirement SDP3.3-2 has two children: L1APR03-I-2 and LIAPR01-I-2. Figure 2 shows the text for an exemplary requirement from the CM-1 [8] dataset.

4. RETRO

RETRO consists of an IR toolbox, a GUI (shown in Figure 3), and a set of analysis tools. The IR toolbox is implemented in C++ and contains a variety of IR methods, adapted for the purpose of the requirements tracing task. The GUI component is implemented in JAVA and can be used to access the methods from the IR toolbox. The analyst can use the GUI to set up tracing projects, to analyze the candidate lists, and to provide feedback to the tool. The interface contains, at the top of the screen, the list of high level requirements (left) and the list of candidate links for the high level requirement selected, with relevance factor (right). The middle part of the screen contains the text of the selected high level and low level requirement pair. The controls at the bottom of the screen allow the analyst to supply feedback about the current candidate link under consideration. Upon analyst request, the feedback information is sent to the feedback processing module implemented in C++ and the candidate list is refreshed according to the new result.

5. EXPERIMENTAL SETUP

We have tested four methods (tf-idf, tf-idf+thesaurus, LSI, LSI+thesaurus) on two datasets (CM-1, MODIS). In each test, we have simulated perfect analyst behavior for eight steps. Four types of behavior, Top1, Top2, Top3 and Top4 were simulated: Top*i* means that on each step and for each high-level requirement, we verify the first *i* of its unverified candidate links. For each run

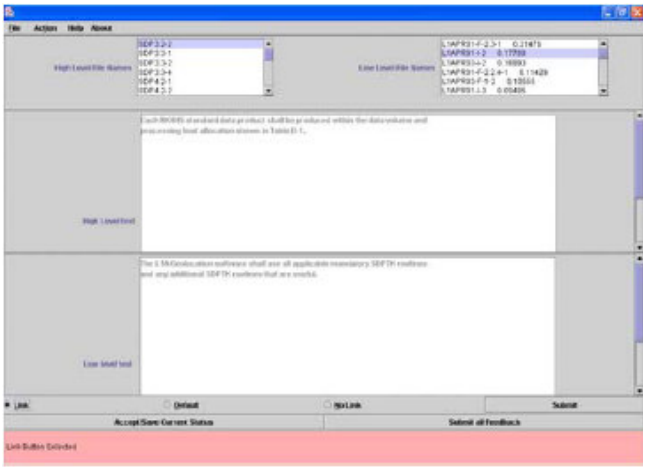

Figure 3. A screenshot of RETRO.

and iteration, we collected the resulting candidate trace. We determined the values of primary and secondary measures for each iteration for the initial candidate trace and for *filtered* candidate traces. These were obtained by removing (from the initial trace) the links with relevance lower than a threshold α. We used values of α=0.05, 0.1, 0.15, 0.2, 0.25.

6. EXPERIMENTAL RESULTS

Table 1 shows the results of running all four methods on both datasets with no feedback and no filtering. We show two runs of LSI methods, with 10 and 19 dimensions for MODIS and with 100 and 200 dimensions for CM-1. We note that all methods gave excellent recall for the CM-1 dataset, and good-to-excellent recall for MODIS. At the same time, precision and selectivity were quite low for both datasets, especially for CM-1.

Table 1. Results of IR methods on CM-1 and MODIS datasets, no feedback, no filtering.

	MODIS			CM-1		
	Prec.	**Recall**	**Sel.**	**Prec.**	**Recall**	**Sel.**
tf-idf	7.9%	75.6%	41.9%	1.5%	97.7%	42.8%
tf-idf+TH	**10.1%**	**100.0%**	43.1%	1.5%	97.7%	42.8%
LSI (10/100)	6.3%	92.6%	64.1%	0.9%	98.6%	71.5%
LSI+TH (10/100)	6.5%	95.1%	63.7%	0.9%	98.6%	71.5%
LSI (19/200)	4.2%	63.4%	65.2%	0.9%	98.8%	73.9%
LSI+TH (29/200)	5.4%	80.4%	65.8%	0.9%	98.8%	73.9%

Table 2 shows the effects of filtering on the results of IR methods with no feedback. For brevity, all cells of the table have the format *(Precision, Recall, Selectivity)*. We report results for LSI on 10/100 dimensions for MODIS/CM-1 respectively. The numbers for tf-idf + thesaurus and LSI+thesaurus for CM-1

Table 2. Effects of filtering on the results of IR methods for MODIS and CM-1 datasets, no feedback.

	MODIS				CM-1	
	tf-idf	tf-idf+TH	LSI	LSI+TH	tf-idf	LSI
Filter	Precision, Recall, Selectivity					
0	7.9%, 79.6, 41.9%	10.1%, 100%, 43.1%	6.3%, 92.6%, 64.1%	6.5%, 95.1%, 63.6%	1.5%, 97.8%, 42.8%	0.9%, 98.6%, 71.5%
0.05	7.7%, 48.7, 27.6%	12.1%, 78%, 28.3%	8%, 58.5%, 32.2%	9.3%, 68.3%, 32.2%	**4.3%, 92.2%, 14.7%**	**3.9%, 91.4%, 16.3%**
0.1	11.7%, 29.2%, 10.9%	22.3%, 65.8%, 13%	9.3%, 29.2%, 13.7%	14.9%, 46.3%, 13.6%	**10.8%, 76.4%, 4.92%**	**9.3%, 77%, 5.7%**
0.15	17.2%, 24.4%, 6.2%	25.6%, 46.3%, 7.9%	15.2%, 26.8%, 7.7%	19.2%, 39%, 8.9%	19.1%, 53.7%, 1.9%	16.8%, 56.5%, 2.3%
0.2	21.6%, 19.5%, 3.9%	33.3%, 39%, 5.1%	20.4%, 21.9%, 4.7%	22.4%, 31.7%, 6.2%	27.1%, 32.6%, 0.8%	24.8%, 39%, 1%
0.25	32%, 19.5%, 2.7%	36.3%, 29.2%, 3.5%	25%, 19.5%, 3.4%	27.9%, 29.2%, 4.6%	34.8%, 21.9%, 0.4%	31.6%, 24.3%, 0.5%

Table 3. Effects of relevance feedback (Top 2) on the results of the IR methods, MODIS and CM-1 datasets, no filtering.

	MODIS				CM-1	
I	tf-idf	Tf-idf+TH	LSI	LSI+TH	tf-idf	LSI
	Recall Precision Selectivity					
0	75.6% 7.9% 41.9%	100.0% 10.2% 43.2%	92.7% 6.4% 64.1%	95.1% 6.6% 63.7%	97.8% 1.6% 42.8%	98.6% 1.0% 71.5%
1	78.0% 8.2% 42.1%	100.0% 10.0% 43.9%	87.8% 6.3% 61.7%	95.1% 7.1% 58.6%	97.8% 1.6% 43.6%	98.9% 1.0% 71.1%
2	87.8% 9.4% 41.4%	100.0% 9.8% 44.8%	85.4% 6.4% 58.9%	100.0% 7.4% 59.6%	97.8% 1.6% 43.8%	98.6% 1.0% 70.4%
3	85.4% 9.1% 41.1%	100.0% 9.9% 44.4%	92.7% 7.3% 56.0%	95.1% 7.6% 55.2%	97.8% 1.6% 43.5%	98.6% 1.0% 69.7%
4	87.8% 9.9% 38.9%	100.0% 10.6% 41.4%	92.7% 7.7% 53.3%	95.1% 8.4% 50.1%	97.8% 1.6% 43.3%	98.6% 1.0% 69.1%
5	85.4% 10.5% 35.7%	100.0% 11.5% 38.3%	87.8% 8.2% 47.3%	95.1% 8.8% 47.8%	97.8% 1.6% 43.0%	98.6% 1.0% 68.5%
6	85.4% 11.6% 32.3%	100.0% 12.9% 34.2%	87.8% 8.7% 44.3%	95.1% 9.1% 46.2%	98.1% 1.6% 42.7%	98.6% 1.0% 67.9%
7	82.9% 13.3% 27.4%	100.0% 14.4% 30.5%	90.2% 9.6% 41.2%	95.1% 9.9% 42.4%	98.3% 1.6% 42.5%	98.6% 1.0% 67.2%
8	82.9% 18.0% 20.3%	100.0% 18.8% 23.4%	87.8% 10.9% 35.3%	92.7% 10.2% 39.8%	98.3% 1.6% 42.2%	98.3% 1.0% 66.5%

dataset are the same as the numbers for tf-idf and LSI respectively (no change). As can be seen from the table, filtering (for the most part) drastically reduces recall, while improving precision to the level of 20-35% and improving selectivity roughly tenfold. Filtering with small (0.05, 0.1) filters produces more stable results for the CM-1 dataset than for MODIS.

Table 3 documents the effects of feedback on the results of IR methods with no filtering. The contents of each cell have the format *(Recall, Precision, Selectivity).* In this table, we list the results of Top 2 feedback strategy. For the MODIS datset, tf-idf shows slight improvement in recall and more than twofold improvement in precision. Tf-idf + thesaurus shows improvement in precision without the loss of total recall. For both datasets, feedback for LSI results in slight loss of recall, while tf-idf for CM-1 shows no change.

Table 4 shows what happens when filtering and feedback are combined. We include the recall and precision results of tf-idf and tf-idf+thesaurus runs for the MODIS dataset and tf-idf run for

Table 4. Effects of relevance feedback and filtering on the results of IR methods.

(a) MODIS dataset, tf-idf

Filter	0.05		0.1		0.15		0.2	
It.	Rec (%)	Pr. (%)	Rec (%)	Pr. (%)	Rec (%)	Pr. (%)	Rec (%)	Pr. (%)
0	48.8	7.8	29.3	11.8	24.4	17.2	19.5	21.6
1	48.8	8.1	29.3	12.6	24.4	20.0	22.0	36.0
2	48.8	8.9	31.7	17.6	24.4	31.3	24.4	47.6
3	53.7	11.1	31.7	21.7	31.7	38.2	31.7	61.9
4	65.9	14.9	46.3	33.9	36.6	51.7	31.7	65.0
5	68.3	19.7	53.7	51.2	46.3	70.4	41.5	77.3
6	**70.7**	**33.7**	**65.9**	**64.3**	48.8	74.1	48.8	80.0
7	**75.6**	**50.0**	**68.3**	**70.0**	63.4	78.8	51.2	80.8
8	**80.5**	**58.9**	**70.7**	**74.4**	**68.3**	**82.4**	**63.4**	**86.7**

(b) MODIS dataset, tf-idf+thesaurus

Filter	0.05		0.1		0.15		0.2	
It.	Rec (%)	Pr. (%)	Rec (%)	Pr. (%)	Rec (%)	Pr. (%)	Rec (%)	Pr. (%)
0	78.0	12.1	65.9	22.3	46.3	25.7	39.0	33.3
1	78.0	12.1	61.0	21.6	51.2	33.3	41.5	44.7
2	78.0	12.7	61.0	25.0	51.2	42.9	46.3	61.3
3	78.0	14.1	63.4	29.5	56.1	50.0	53.7	68.8
4	**90.2**	**18.9**	**75.6**	**37.3**	56.1	52.3	56.1	69.7
5	**95.1**	**22.4**	**75.6**	**46.3**	58.5	57.1	56.1	67.6
6	**97.6**	**30.3**	**78.0**	**52.5**	**65.9**	**69.2**	58.5	70.6
7	**97.6**	**39.2**	**92.7**	**60.3**	**78.0**	**74.4**	**75.6**	**77.5**
8	**97.6**	**43.0**	**92.7**	**65.5**	**90.2**	**77.1**	**78.0**	**82.1**

(c) CM-1 dataset, tf-idf

Filter	0.05		0.1		0.15		0.2	
l	Rec (%)	Pr. (%)	Rec (%)	Pr. (%)	Rec (%)	Pr. (%)	Rec (%)	Pr. (%)
0	92.2	4.4	***76.5***	***10.8***	53.7	19.1	32.7	27.1
1	91.7	4.3	***77.0***	***10.9***	55.4	19.8	38.0	31.6
2	91.4	4.3	***76.2***	***10.8***	59.0	20.8	42.9	34.4
3	91.7	4.4	***77.6***	***10.9***	63.2	22.0	45.4	34.8
4	92.0	4.4	***78.9***	***11.1***	66.5	23.1	48.8	35.6
5	92.0	4.4	***81.4***	***11.5***	67.6	23.6	52.6	37.6
6	92.2	4.4	***82.8***	***11.7***	**70.1**	**24.3**	55.4	39.1
7	92.2	4.4	***84.2***	***11.9***	**70.9**	**24.5**	57.6	39.6
8	92.2	4.5	***84.8***	***12.0***	**74.0**	**24.8**	**61.2**	**40.9**

CM-1. For MODIS, both recall and precision show stable improvement through the iterations. Filtering at 0.05 and 0.1 levels produces excellent recall and excellent precision. At higher filtering values, precision improves even more, but at the price of decreased recall. For CM-1, filtering at 0.1 produces very good recall, with precision around 10-12% (a significant improvement over the results shown in Table 1). Filtering at higher thresholds leads to significant improvement in precision, but still keeps recall relatively high: above 70% for 0.15 and above 60% for 0.2.

In Table 5, we briefly summarize the best results achieved during other runs of the experiment. We note that, as expected, the results of Top1 behavior are somewhat worse, while the results of Top3 behavior are somewhat better than the results of Top2 behavior. Our choice to highlight the results of Top2 behavior is explained by our desire to balance the quality of the candidate trace and the amount of work (in terms of analyst feedback) needed at each iteration.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we discussed a variety of IR methods applied to the requirements tracing problem. We focused on our results with open source datasets that have been posted at http://promise.site.uottawa.ca/SERepository/. We have established that already-existing IR methods can be used to automate candidate link generation with minimal modification. We evaluated the methods and found that we can achieve recall and precision in line with and even better than existing tools [4]. We also demonstrated that feedback information from the analyst can significantly improve requirements tracing.

Our future work is in several directions. First, we want to apply more methods and evaluate their performance. Second, we want to investigate the impact of the analyst on the tracing process. That includes evaluating how the quality of candidate link lists affects analyst decisions. A small pilot study is underway. Finally, we want to perform usability analysis of RETRO. We anticipate that improvements can be made.

Table 5. Results of experiments: best of the rest.

Dataset	Method	Fback	Filter	It.	Pr.	Rec.
MODIS	LSI	Top2	0.05	8	**39.4**	**68.3**
MODIS	LSI	Top2	0.1	7	**56.4**	**53.6**
MODIS	LSI	Top2	0.2	6	**66.6**	**53.6**
CM-1	LSI	Top2	0.15	8	**24.1**	**73.6**
CM-1	LSI	Top2	0.2	8	**37.7**	**60.1**
MODIS	tf-idf	Top1	0.1	8	**68.3**	**68.3**
MODIS	tf-idf	Top1	0.15	8	**75.8**	**53.5**
MODIS	tf-idf	Top3	0	8	**25**	**85.3**
MODIS	tf-idf	Top3	0.1	7	**62.2**	**80.4**
MODIS	tf-idf	Top3	0.2	8	**86.1**	**75.6**
MODIS	tf-idf	Top3	0.35	8	**93.1**	**68.8**
MODIS	Tfidf-th	Top1	0.05	8	**31.5**	**100**
MODIS	Tidf-th	Top1	0.1	8	**51.6**	**78**
MODIS	Tfidf-th	Top3	0.05	5	**26.7**	**100**
MODIS	Tfidf-th	Top3	0.1	8	**63**	**100**
MODIS	Tfidf-th	Top3	0.2	7	**78.7**	**90.2**
CM-1	tf-idf	Top3	0.15	8	**25.3**	**77**
CM-1	tf-idf	Top3	0.2	8	**40.7**	**62**

8. ACKNOWLEDGEMENTS

Our work is sponsored by NASA under grant NAG5-11732. We thank Stephanie Ferguson, Ken McGill, and Tim Menzies. Also, thanks to the Metrics Data Program for the use of CM-1 and to the MODIS program for making their data publicly available. We also thank Sarah Howard and James Osborne, who worked on earlier versions of RETRO.

9. REFERENCES

[1] Baeza-Yates, R. and Ribeiro-Neto, B. Modern Information Retrieval, Addison-Wesley, 1999.

[2] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., and Harshman, R. "Indexing by Latent Semantic Analysis," 1990, Journal of the American Society of Information Science.

[3] Haritsa, J., Carey, M., and Livny, M. On Being Optimistic about Real-Time Constraints, *Proc. of ACM SIGACT-SIGMODSIGART* Symposium on Principals of Database Systems, April 1990, pp. 331-340.

[4] Hayes, J.H., Dekhtyar, A., and James Osborne, Improving Requirements Tracing via Information Retrieval, in Proceedings, 11th International Requirements Engineering Conference (RE 2003), pp. 151-161, September 2003, Monterey Bay, CA.

[5] Hayes, J.H., Dekhtyar, A., Sundaram, S.K., and Sarah Howard, Helping Analysts Trace Requirements: An Objective Look (2004), in Proceedings, 12th International Requirements Engineering Conference (RE 2004), pp. 249-261, September 2004, Kyoto, Japan.

[6] Hayes, J.H., Dekhtyar, A., Sundaram, S.K., Measuring the effectiveness of Retrieval Techniques in Software Engineering, October 2004, (TR422-04).

[7] Requirements Specification, SDST-0591, GSFC SBRS, September 11, 1997.

[8] MDP Website, CM-1 Project, http://mdp.ivv.nasa.gov/mdp_glossary.html#CM1.

[9] MODIS Science Data Processing Software Requirements Specification Version 2, SDST-089, GSFC SBRS, November 10, 1997.