TREC-II Routing Experiments with the TRW/Paracel Fast Data Finder

Matt Mettler
TRW Systems Development Division
Redondo Beach, CA

Fritz Nordby
Paracel Inc.
Pasadena, CA

1.0 Introduction

For TREC-II, we were interested in experimenting with improved methods of constructing queries for the Fast Data Finder (FDF) text search coprocessor. We learned from TREC-I that while the pattern matching ability of the FDF can sometimes be put to significant advantage (we had the high score on 8 of the 50 routing topics in TREC-I), this wasn't sufficient overall to overcome the weaknesses traditionally associated with the boolean approach to text retrieval. Many of the TREC topics are too abstract and ambiguous to respond well to a boolean query formulation.

Our goal for this year therefore, was to apply the FDF hardware to a more statistical or soft boolean retrieval approach while not giving up on our ability to make use of specific features or patterns in the text when they are obviously important.

We experimented with two different schemes. In the first scheme, we utilized subquery proximity to rank hit documents. We developed the subqueries manually, then determined the optimum proximity values by test runs on the training data. The most effective values were then used in the official routing queries. The second scheme was an FDF adaptation of the traditional Information Retrieval (IR) term weighting approach. In addition to single word terms, we also included two and three word phrases, and FDF subqueries designed to detect special features in the text.

While in the terminology of TREC both are examples of manual query formulation with feedback, we believe these techniques can be evolved to create queries automatically from samples of relevant text and to also incorporate user knowledge of specific text features of interest when it exists. We also continue to believe that the utilization of a hardware accelerator such as the Fast Data Finder, enables the implementation of high performance routing or dissemination applications at a far lower cost than can be achieved with conventional general purpose processors.

2.0 The FDF Text Retrieval Approach

The Fast Data Finder is a hardware device that performs high-speed pattern matching on a stream of 8-bit data. It consists of an array of identical programmable text processing cells connected in series to form a pipeline processor. The cells are implemented using a custom VLSI chip designed and patented by TRW. In the latest implementation, each chip contains 24 processor cells and a typical system will have 3,600 cells. Each cell can match a single character of query or perform all or part of a logical operation. The processors are interconnected with an 8-bit data path and approximately 20-bit control path. To perform a search, a microcode program is first downloaded into the pipeline to direct each processor. The database is then streamed through the pipeline. The data bytes clock through each processor in turn until the whole database has passed through all processors. As the data is clocking through, the processors alter the state of the control lines depending on their program and the data stream values.

When the pipeline's processor cells detect that a series of database characters match the desired pattern, a hit is indicated and passed by external circuitry back to the memory of the host processor and to the user. The FDF pipeline runs at a constant speed as it performs character comparisons and logical operations, regardless of query complexity.

The queries or patterns are specified in the FDF's Pattern Specification Language (PSL). The hardware directly supports all the features in the PSL query language without the need for software post-processing. The processors in the pipeline may all be used to evaluate a single large query or may be assigned to evaluate numerous smaller queries. The number of pipeline cells a query needs is proportional to the size of the query. PSL provides numerous search functions, which may be nested in any combination, including:

- Boolean logic including negative conditions
- Proximity on any arbitrary pattern
- Wildcards and “don't cares” anywhere in the word
- Character alternation
- Term counting, thresholds, and sets
- Error tolerance (fuzzy matching)
- Term weighting
- Numeric ranges

The Fast Data Finder was originally designed and developed at TRW. In 1992, TRW licensed the FDF technology to Paracel Inc., which now sells a commercial product called the FDF-3.

3.0 Proximity Query Generation

Our first set of experiments revolved around the use of subquery proximity to rank hit documents. We began with a simple observation: topics are often a conjunction of ideas or concepts. For example, Topic 51, Airbus Subsidies, is a conjunction of the idea “Airbus” (a particular aircraft manufacturer and European consortium) and the idea “subsidy” (in particular, subsidies from the nations belonging to that consortium). Other articles about Airbus Industrie (new planes, fly-by-wire in the A320, accident reports, financial health reports, etc.) are not relevant to this topic; nor are articles which describe subsidies not directed toward Airbus.

Traditional IR term weighting techniques do not give any explicit benefit to articles which conjoin ideas. Articles which include terms relevant to each of the component sub-topics will receive high scores; but so will articles which include many terms relevant to only one sub-topic. Recent efforts implicitly include conjunctions through the use of phrases as terms in otherwise traditional statistical methods.

An alternative is the use of boolean operators. This has the desired effect -- an AND of terms forces a conjunction -- but the use of booleans in IR has been viewed with some skepticism and disfavor. Boolean operators often find a conjunction of terms where none truly exists (for example, Airbus and subsidies might be mentioned in two separate and unrelated portions of an article); or, if made sufficiently restrictive to eliminate spurious matches, boolean-based searches often miss relevant articles.

We have followed an approach which incorporates both ideas. Rather than focus on specific phrases, we search for terms in proximity to one another. The terms in the query are chosen to represent each of the constituent sub-topics, just as in a boolean search. The specificity of the query is adjusted by varying the required proximity of the terms. Thus, for Airbus subsidies we might search for terms representing “Airbus” in a range of proximities to terms representing “subsidies”.

This approach allows conjunctions to be graded. A small proximity restriction (say, 3 words) yields results similar to a keyphrase search, indicating that the two concepts are indeed associated in the article and that the article is relevant to the topic. A large proximity restriction (1 article) is analogous to a simple boolean keyword search and retreives articles in which the concept terms may be only loosely associated. Intermediate proximities (1 sentence, 1 paragraph, etc.) indicate intermediate degrees of association and intermediate recall/precision trade-offs.

It is also possible to use multiple proximities in a single query with this method, or to use proximities and occurrence frequencies together, to form multi-dimensional arrays of query parameters. For example, for Topic 62, Military Coups D'etat, the number of conjunctions was traded off against the proximity of the conjunction to form a two-dimensional query set.

For the initial experiment, lists of synonyms representative of each idea in a topic were manually built, and one- or two-dimensional query sets were built from these lists. These queries were then run against the training database, and after some feedback, the query sets were finalized. Each finalized query set was run against the training database to determine a ranking of the queries based solely on selectivity.

Table I shows a sample proximity query and Table III shows our TREC-II results. The number of relevant documents retrieved by the proximity method queries are labeled TRW1.

4.0 Statistical Query Generation

Our second set of experiments revolved around the use of term weighting. Our basic approach was to follow the well researched path of generating term weights proportional to the occurance of words in a sample of relevant text and inversely proportional to the occurance of words in the database as a whole. Since we were doing the routing topics, we gathered statistics on Volume II and hoped that they would be valid for Volume III. For sample relevant documents we used the NIST TREC-I relevant documents from Volume II. We did not use the topic narratives or descriptions. In addition to single word terms, we also considered two and three word phrases. We used the FDF itself to scan the training corpus and determine the phrase frequencies of interest.

To adapt this standard approach to the FDF, we needed to make three algorithmic modifications. First, we needed to adapt to the limitations imposed by the FDF hardware. While extremely effective for pattern matching, the FDF is not a general purpose computer. While the FDF processor cells can perform basic addition/subtraction, the datapath available to accumulate an aggregate score for a document is limited to 8 or 9 bits. Thus we had to restrict the term weights (and the range of their sums) to integer values between 0 and 255 or 0 and 511. This had the effect of truncating most topic's query terms at 10-20 terms (words, phrases, or special features). We also excluded terms from our queries that did not appear in at least 30% of the relevant sample documents.

Second, we were striving to not give up the strengths of the FDF's pattern matching capabilities to pinpoint special features in the text which have a large impact on document relevance. We manually reviewed the topics and prepared special feature subqueries in an attempt to increase the precision for particular topics. For Topic 59, Weather Related Fatalities, we manually prepared a special feature subquery to detect phrases detailing a numeric value of people killed. We determined the frequency of each special feature, both in the sample relevant documents and in the training database as a whole, and just added these into the word list as if they were regular single word terms. In some instances our manually prepared subqueries jumped to the top of the list of statistically relevant terms for a topic; in others they didn't.

Third, we observed that some topics had particular words, phrases, or special features that were present in almost all relevant documents. We converted terms that occurred in > 90% of the relevant documents to boolean ANDs in our queries. This was intended to improve precision for topics like 62, Military Coups D'etats. The topic narrative specifically stated that the country involved must be named. One of our special feature subqueries was a list of known foreign country names. While of no statistical significance as a term, this subquery did hit on almost every sample document. We thus ANDed it into the query as a required boolean term.

Table II shows a sample statistical query. DocCount is the number of documents in the sample that included the term, phrase, or special feature. DbCount is the number of documents in our training sample that included the term. Weight is DocCount divided by DbCount. PslWeight is the integer coefficient based on the Weight. The relevant documents retrieved by the statistically generated queries are labeled TRW2 in Table III.

TABLE I – Sample Proximity Query

```
Query Template for Topic 75, Automation

/* Concept 1: automation */
define x_auto 'automat[e|ion|ed|es|ing]' end

/* Concept 2: change in economics */
define x_up_dn  '[increas|decreas][e|es|ed|ing]'
      |'rais[e|es|ed|ing]'          |'lower[|s|ed|ing]'
      |'ris[e|es|ing]'              |'fall[s|ing]'
      |'drop[|s|ped]'               |'hike[|s|d]'
      |'dip[|s|ping]'               |'cut[|s|ting]'
      |'expan[d|ds|ded|sion[|s]]'   |'reduc[e|es|ed|tion[|s]]' end

define x_econ 'cost[|s]' | 'payroll*'
      |{ 3 words -> 'work' and 'force') } end

/*
 * Query template used for Topic 75
 */
{ (prox1) ->
      @x_auto and
      { (prox2) -> @x_up_dn and @x_econ } }
```

TABLE II – Sample Statistical Query

Term weighting table for Topic 93, NRA Political Backing

Doc sample size is 150

Term	DocCount	DbCount	Weight	PslWeight
nra	87.0	130.0	0.66923077	62
national_rifle_a	145.0	255.0	0.56862745	53
assault_weapons	51.0	161.0	0.31677019	30
semiautomatic	68.0	332.0	0.20481928	19
gun_control	73.0	387.0	0.18863049	18
handguns	51.0	319.0	0.15987461	15
rifle	146.0	1336.0	0.10928144	10
handgun	52.0	550.0	0.09454545	9
firearms	63.0	811.0	0.07768187	7
rifles	46.0	1318.0	0.03490137	3
gun	126.0	3866.0	0.03259183	3
guns	96.0	3020.0	0.03178808	3
law_enforcement	51.0	2509.0	0.02032682	2
assault	67.0	3491.0	0.01919221	2
ban	91.0	5510.0	0.01651543	2
enforcement	54.0	4825.0	0.01119171	1
weapons	99.0	8993.0	0.01100856	1
association	148.0	16780.0	0.00882002	1
legislation	64.0	7984.0	0.00801603	1
laws	45.0	7662.0	0.00587314	1

TABLE III - TRW/Paracel TREC-II Routing Results

Qry	NIST Rel.	Relevant Retr @100 Best	Median	Worst	TRW1 Rel	TRW2 Rel	Description
51	11	11	10	0	10	10	Airbus Subsidies
52	454	100	97	31	99	99	S. Africa Sanctions
53	154	84	50	1	52	55	Leveraged Buyouts
54	124	65	57	0	21	47	Sat. Launch Contracts
55	320	100	87	0	54	96	Insider Trading
56	395	96	88	3	89	96	Prime Rate Moves
57	319	91	73	0	61	91	MCI
58	76	62	39	4	49	28	Rail Strikes
59	574	97	80	1	95	95	Weather Fatalities
60	18	9	4	0	2	3	Merit-Pay vs. Seniority
61	67	60	24	2	7	24	Israel & Iran-Contra
62	426	92	75	9	82	81	Military Coups D'etat
63	74	40	24	0	24	29	Machine Translation
64	282	68	53	1	29	57	Hostage-Taking
65	214	55	29	0	2	46	Info Retrieval Systems
66	86	40	19	0	20	23	NLP
67	365	64	52	0	10	61	Civil Disturbances
68	76	61	38	0	58	55	Health Hazards
69	1	1	0	0	0	0	Reviving SALT II Treaty
70	34	32	28	0	31	32	Surrogate Motherhood
71	300	73	35	1	21	32	Border Incursions
72	91	43	31	0	17	30	Demographic Shifts/U.S.
73	355	74	49	0	23	74	Demographic Shifts/World
74	323	60	32	5	6	35	Conflicting Policies
75	372	59	28	1	59	31	Automation
76	163	34	24	0	11	34	Original Intent
77	85	49	36	0	36	40	Poaching
78	83	66	58	0	40	57	Greenpeace
79	341	85	75	0	81	77	FRG Party Positions
80	143	54	8	0	54	8	Candidate Platforms
81	4	4	2	0	1	2	PTL Fallout
82	203	89	65	0	30	87	Genetic Engineering
83	235	64	31	2	54	25	Protect the Atmosphere
84	101	27	16	0	22	14	Alternative Energy
85	670	88	69	38	59	84	Official Corruption
86	40	27	16	0	19	12	Bank Failures
87	151	75	42	2	42	36	S&L Prosecutions
88	32	29	21	0	11	26	Crude Oil Price Trends
89	17	8	2	0	3	7	OPEC Investments
90	75	39	10	0	39	9	Oil & Gas Reserves
91	9	6	3	0	5	5	Acq of Advanced Weapons
92	27	17	6	0	7	9	Military Equip Sales
93	94	65	61	9	61	62	NRA
94	300	76	47	0	57	62	Computer Crime
95	359	77	45	1	63	40	Computer Crime Detection
96	310	86	42	2	23	49	Computer Medical Diag
97	319	59	21	0	20	41	Fiber Optics Appl
98	722	96	67	0	83	84	Fiber Optics Manuf
99	291	99	92	10	91	89	Iran-Contra
100	204	94	84	0	52	89	Controlling High Tech

5.0 Analysis of Results

The results from our two TREC runs (Table III) are summarized below. The proximity queries (TRW1) scored at or above the median on 28 topics (including three topics which achieved the best score) and below median for 22 topics. While not bad, our proximity queries did not perform as well as we'd hoped. Where the proximity queries did poorly, we attribute this primarily to poor term selection. One such case was Topic 65, Information Retrieval Systems. We made two errors: first, due to an oversight, one of the manually entered query terms was overly broad; second, the query author considered database systems to be "information retrieval systems". We feel that this is a fault of the query formulation, not of the assessments. If we had checked the training assessments, we would have eliminated the terms from our query. Any system which relies solely on the topic statement will run afoul of this problem. Systems which make use of user-supplied relevance information will achieve better performance. Our results again demonstrate the inherent problems with basically boolean query formulations. Our efforts at using proximity to soften the boolean were not sufficient to overcome this weakness.

Run	High	Above Med	Median	Below Med	Low	11-pt avg
TRW1	3	19	6	22	0	0.2525
TRW2	5	27	5	13	0	0.3459

The statistical queries (TRW2) did much better, scoring at or above the median on 37 of the topics. The 11-pt average and total relevant documents retrieved figures were excellent and are close to the best academic groups. The adaptations to run the statistical queries on the FDF hardware evidently did not hurt performance. We again observed that the dominant factor in achieving good performance is proper term selection. The details of the term weight calculations didn't seem to make much difference except to influence which terms were selected. We tried a number of different schemes for generating term weights including using various statistical parameters, log weighted coefficients, and converting terms present in all sample documents to boolean ANDs in the query. For a given set of terms, we did not find much difference in performance between these schemes. The scheme used for TRW2 was one of the simpler ones we tried.

We were also interested in evaluating the use of phrases and special features as additional terms in our statistical queries. They seemed to help, but not dramatically. This was a disappointment. Looking at the results topic by topic however, we observed a lot of variation. For some topics, the addition of a key phrase or special feature helped a great deal. This indicates that use of phrases and special features has promise for improving performance, but that we just have not learned how and when to employ them. For example, our term weighting scheme this year didn't account for term interdependence. Particularly when we start mixing single word terms with phrases and special features that contain those same terms, it would seem the algorithm could be improved by explicitly accounting for this redundancy.

6.0 Conclusions and Future Plans

Overall, our results for TREC-2 are encouraging. We were successfull in adapting the FDF hardware to running a soft statistical information retrieval algorithm. Not only did the FDF run all the final searches with no significant post-processing, we also found it a useful tool for experimenting with different proximity windows and deriving database frequencies for phrases and special features. We plan to continue our work and to participate in TREC-III next year. The lessons we learned from TREC-I have proven very valuable, and so too should the lessons of TREC-II. For TREC-III, we will continue to examine the range of available query formulation, execution and evaluation models, with an eye to adapting those models for use with the FDF search hardware. We are looking at methods to use the raw horsepower of the FDF to expand existing techniques in ways which had previously been considered too computationally expensive.

We are also struck by the observation that different query generation techniques are effective on different topics. A system that could execute a range of methods might be able to match the query generation approach to the topic type. We plan to consider possible schemes for characterizing the topic in advance so that the system might be able to use the best method.