# A PADRE in MUFTI
# (A Multi User Free Text retrieval Intermediary)

D. Hawking P. Bailey D. Campbell P. Thistlewaite
A. Tridgell
Cooperative Research Centre For Advanced Computational Systems
Department of Computer Science
The Australian National University
Canberra ACT 0200 AUSTRALIA

E-mail: dave@cs.anu.edu.au

**Abstract**

The Parallel Document Retrieval Engine (PADRE) has hitherto lacked the ability to support multiple time-sharing searchers, a deficiency which detracted from its cost effectiveness. Extensions, adjuncts and performance improvements are now proposed to permit multiple use and to ensure reasonable response times. This paper describes the proposed multi-user architecture, reports query-processing speed-ups, outlines a number of alternative types of user-interaction client including an interface to network browsers in common use. Implementation progress is reported and potential load handling capacity is discussed.

KEYWORDS Text retrieval, information retrieval, parallel computing.

## 1 Introduction

The Parallel Document Retrieval Engine (PADRE) has previously demonstrated that full text scanning methods supported by parallel hardware permit powerful query constructors and rapid response to changing document collections [3, 5, 6]. The addition of parallel-disk-resident inverted file indexes and dictionaries has potentially extended data handling capacity to the terabyte level, with some loss of flexibility [7].

Parallel machines significantly extend the range of what is possible in document retrieval but the cost of the parallel hardware is excessive in most foreseeable circumstances unless multiple "simultaneous" users can be supported. Up to the present time, PADRE has been a single-user system but a time-sharing, multi-user architecture is now under development.

Work to achieve a satisfactory result has taken four directions:

1. Devolving responsibility for user interaction away from PADRE to remote clients;
2. Designing suitable interaction clients and an interface to commonly used information navigation clients;
3. Designing a suitable timesharing manager for remote interactions;
4. Tuning search engine performance to maximise the number of timesharing clients which may be supported.

# 2 Search Engine Performance Improvements

Retrieval performance which is adequate when dedicated to a single user becomes unacceptable when shared among a large number. Consequently, speed of the basic search engine is critical to a multi-user PADRE. Accordingly, considerable work has been done both to speed up query processing based on full text scanning and to improve the capabilities of indexed searches while endeavouring to preserve the already high performance using this method.

Research topics such as those in the 1995 Text Retrieval Conference task [2] may be considered typical of specifications likely to be generated by subject experts lacking experience with automated retrieval systems. An example of such a topic is: *Where are all the nuclear power stations in the United States and what has been their rate of production?* If the query generated from this topic searches only for `nuclear power station`, `United States` and `rate of production` occurring in proximity with each other, precision/recall performance is likely to be poor. A query which is expanded to include alternative expressions to the terms appearing in the topic description is likely to perform much better. Manually generated PADRE queries in this year's TREC task often included dozens, sometimes hundreds, of alternatives.

Unfortunately, expanded queries tend to lead to proportionately increased processing time. In TREC-3, a list of $k$ alternates were located using $k$ complete Boyer-Moore-Gosper (BMG) passes through the raw text. Use of index-based searches would have speeded things up considerably but many PADRE search modes used in the queries were not supported using the indexed method.

## 2.1 Single-Pass Scanning For Multiple Alternate Patterns

Tridgell and Hawking [10] have developed a BMG-derived single-pass method (referred to as *bmg2* for locating all occurrences of all of a set of alternate literal patterns. Figure 1 shows the very significant speedup achieved with multiple patterns. The results shown are somewhat flattering to the bmg2 method due to the use of constant pattern length. The performance of BMG-derived algorithms improves with pattern length. In the bmg2 case, performance is related to the length of the shortest pattern. If one alternative is much shorter than the others, the advantage of the single-pass method over multi-pass is significantly reduced (but still worthwhile).

## 2.2 Improvements To Index-Based Methods

Indexed searches may now be performed in case-insensitive mode. It is also now possible to search for word prefixes rather than whole words. In addition, cost-effective, index-based methods for locating phrases have been implemented which are able to cope with differences in whitespace or punctuation and with variations in parts of speech of the constituent words.

Further extensions to allow indexed location of arbitrary word parts such as suffixes and regular expressions including a literal part are being contemplated.

## 2.3 Decreased Collection Load Times

In applications where multiple large collections are used or when the collections cannot remain resident in memory for long periods, the speed of loading is vital to a practical query service.

PADRE has benefited considerably from the dramatic decrease in time to load document collections made possible by the HiDIOS filesystem on the cell-connected option disks. [9] Document collections are represented as parallel files. If disk-resident indexes and other structures are used, they are represented in a similar manner.

Using 32 small (0.5 gigabyte) option disks, on a 128 cell AP1000, loading rates in excess of 50 Mbytes/sec. are typical, compared with about 1 Mbyte/sec. for uncompressed loads via the host.

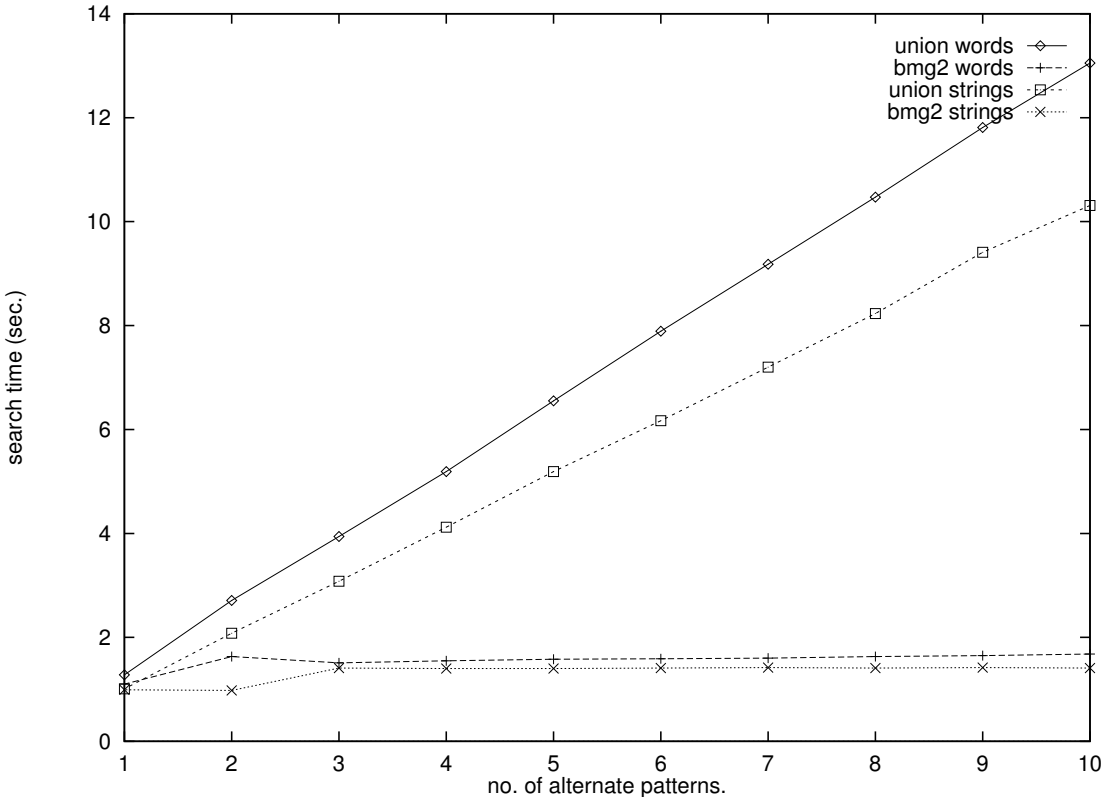


Figure 1: Search times to locate $n$ alternative patterns, where $n$ varies from 1 to 10, for the old multi-pass (union) method and the new bmg2 single-pass method. The dataset (CD2 of the TREC collection) comprised over a gigabyte of data. All patterns were 7 characters long. In one case the patterns are nonsense strings which do not occur anywhere in the text and in the other they are 7-letter words occurring between 7,000 and 23,000 times in the text. Times reported are elapsed times as measured on the host. A 128 cell AP1000 was used.

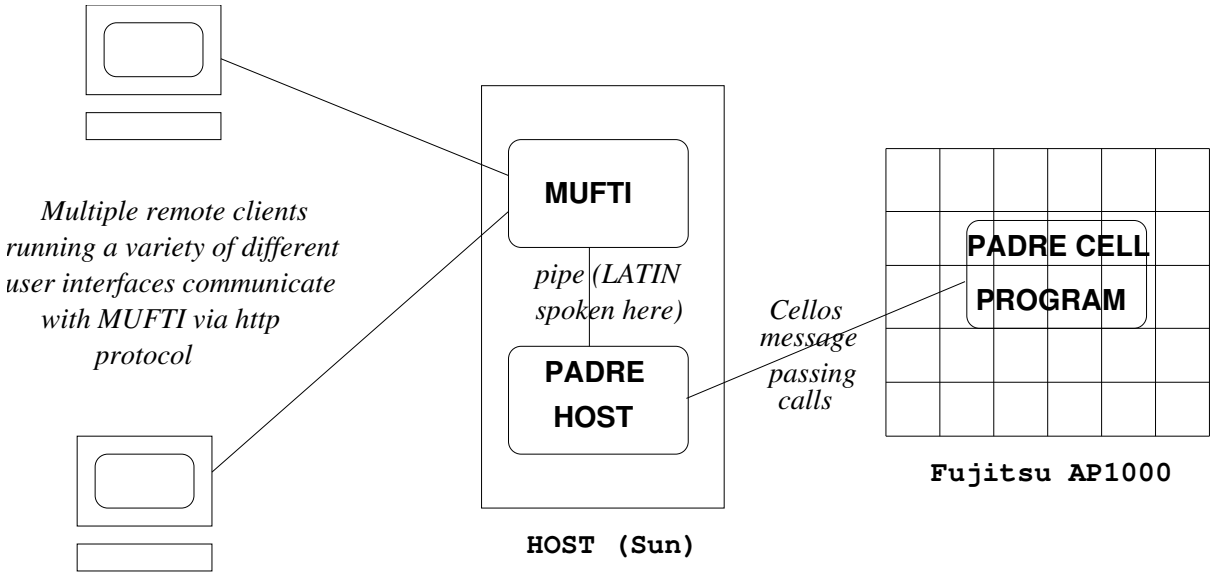


Figure 2: Components of PADRE/MUFTI multi-user architecture.

# 3 PADRE Possessed By A Daemon?

One of us (Campbell) has completed a preliminary implementation of a multi-user, timesharing document retrieval server architecture based on a daemon (MUFTI) running on the AP1000 front-end, as shown in figure 2. MUFTI listens on a TCP/IP port for one or more connections from remote (or local) user-interface clients. Communication over these sockets conforms to the *http* (HyperText Transfer) protocol [8] and conveys PADRE queries in one direction and search results in the other. MUFTI is responsible for starting up PADRE and if it is not already running, loading the appropriate text base if it is not already loaded and for terminating PADRE if no queries have been received for a specified timeout period.

Because the PADRE search engine can process only a single request at a time, there is no need for MUFTI to be multi-threaded either. MUFTI will accept connections from multiple clients but causes them to wait until the currently active request is processed.

When a Web browser connects to the MUFTI URL (Universal Resource Locator), a HTML page is returned which provides some explanatory information and a form including a scrolling text window into which the search query can be typed. When the query is complete, the browser uses its own standard methods to convert the text into the form of an expanded URL which is transmitted to the server. The server performs minor editing to unpack the PADRE query, submits it to PADRE itself and then filters the output into a HTML generated list of "relevant" documents. The searcher may select one of these for retrieval and display.

This architecture is not tied to any particular Web browser and allows for the possibility of multiple styles of query generation client to operate concurrently. Some possibilities are described in the next section.

Document retrieval queries generated by retrieval experts (or by novices with access to query term expansion technology such as thesauri and relevance feedback) are typically quite complex. They are also likely to be subject to a series of interactive modifications as the searcher homes in on the documents they seek. These two characteristics, coupled with the requirement to maximise PADRE throughput, suggest that:

1. Query optimisations performed by the query generation client would be highly desirable. These may

include:

(a) Elimination of redundant operations,

(b) Naming intermediate result sets for later reference,

(c) Re-casting or re-ordering the query to restrict the set of documents which must be considered in the later part of the query, and

(d) Aborting query processing when the number of potentially interesting documents has stabilised at a manageable level.

2. The addition of session management and context saving capabilities to PADRE would be highly desirable. In this model, each separate client connection would be allocated a unique session-id used in all requests. Results of each query would be tagged with the relevant session-id, saved to disk and restored as necessary to reduce the cost of subsequent query modifications.

It may eventually be necessary to introduce micro-scheduling at the sub-query level to improve fairness between multiple simultaneous searchers, even though overall throughput would decrease considerably due to context-switching overheads.

In the initial implementation, there is no microscheduling and no session management.

The basic quanta of work processed by PADRE/MUFTI are of two types:

1. A query which is processed in its entirety and which cannot rely on PADRE to remember any previous state. After processing, results in the form of a list of relevant documents, are sent back to the client and PADRE is reset.

2. A request to return a uniquely specified document (presumably one of the documents listed in response to a query).

# 4 Alternative Query Interfaces

## 4.1 PADRE Character Based Interface

The native PADRE query language LATIN (Local ASCII Text INterface) is not well suited to the needs of human researchers. As a person-machine interface, it is verbose, syntactically exacting and allows many gratuitous opportunities for human error. It will be retained, however, because it is convenient to use when the query generation process is machine assisted. Furthermore, direct, batch-style input is very useful for debugging PADRE modifications and for qualifying new versions.

The more user-friendly query interfaces described below must generate queries in LATIN and pass them to PADRE via MUFTI.

In the future, LATIN is likely to be extended to incorporate predicates and conditionals which may be used to modify the form of a query based on initial results, either to increase efficiency or improve precision or recall.

At the time of writing, the only access to the multi-user capabilities of MUFTI is by typing LATIN in the search query field of the interactive form.

## 4.2 PRELATE / RETRIEVE

Hawking and Bailey [4] described a graphical user interface (RETRIEVE), coded in Tcl/Tk, to a predecessor of PADRE. However, RETRIEVE could only generate queries which were restricted to Boolean combinations of literal terms, whereas, more generally, PADRE queries can be regarded as tree-structures in which literals, regular expressions, numeric ranges or lists of alterantive literals form the leaves and set operators

or proximity operators form the other nodes. Any node may be tagged with a relevance weighting and an action to be taken after completing the operation.

A prototype of a Tcl/Tk interface capable of generating unrestricted queries has recently been developed. It presently operates in off-line mode, generating queries in LATIN for later processing by PADRE. This prototype, known as PRELATE (Padre Retrieval LAnguage Topic Editor) was used in generating the manual queries for this year's TREC task and proved very useful in eliminating certain categories of error and in allowing the user to easily visualise the structure of the evolving query. However, some further improvements to the interface are planned.

It is also planned to make PRELATE work on-line to MUFTI/PADRE by configuring it as an *applet* under a TCL/Tk Web browser such as SurfIt! [1]. We hope that the required modifications will be small.

### 4.3 Automatic Query Generation From Prose Specification

In some institutions, librarians or research assistants take prose descriptions of research topics given to them by researchers and turn them into queries in the language of the information retrieval system in use. The nature of these descriptions is copied by the TREC topics (for example, the one given earlier in this paper). For researchers expert in their subject matter but not in the intricacies of computerised information sources or query languages, this style of relevance definition is very convenient and effective.

Methods have been developed by one of us (Thistlewaite) for automatically converting topics in this form into PADRE queries in LATIN.

The methods are principally based on Natural Language Processing techniques, and do not depend on the frequency of occurrence of words in the topic specification. Topics are syntactically normalised, and are then morphologically and structurally analysed, to identify the key and auxillary components of the topic. An online thesaurus is then used to semantically clarify the topic descriptors, and these are then mapped onto the LATIN syntax. If information is available about the database to be searched - in particular, the frequency and distribution of terms and phrases from the topic - it can be used to generate a partial order of more strict or less strict search expressions, and these can then be automatically applied to the search depending on whether the search objective is to maximise recall or precision.

Automatic query generation from prose is easily adaptable to the multi-user, interactive situation. Relatively small extensions to the current Web interface are required to allow the prose description to be filtered into LATIN before transmission to PADRE.

An interface to PADRE is currently being developed which allows for the automatic dynamic construction and execution of a query. The final query is determined dynamically, in response to both (i) the NLP analysis of the original topic description, and (ii) probing of the actual database for data on the frequency and distribution of expressions. Consequently, the automatic querying process can firstly effect controlled maximisation of recall, and then attempt to maximise precision within the recalled set.

### 4.4 WAIS/Z39.50

Addition of a WAIS/Z39.50 interface to PADRE/MUFTI has been under contemplation for some time. If WAIS clients were the standard tools for querying textbases across the network, such an interface would obviously make PADRE very much more accessible and useful.

However, it is not clear that WAIS will be the dominant standard. Indeed NETSCAPE, possibly the most widely used Web browser at present, does not include a WAIS interface. Furthermore the largest-scale servers of text information currently on the Internet, such as LYCOS do not use WAIS/Z39.50 for external connections either.

The question is further complicated by differences between the querying capabilities of different WAIS clients and by the recent revisions to the Z39.50 standard. The likely degree of acceptance of the new standard or its ramifications for a PADRE interface are unknown at this stage.

At the moment, it seems that the http protocol, reinforced if necessary by CGI (Common Gateway Interface) scripts, is the best way to communicate queries to the server and return document titles and contents to the user.

To date, there has been no progress on the development of a WAIS/Z39.50 interface to PADRE but it is clear that a different daemon from MUFTI will be required to support the different protocol. Changes to PADRE itself will also be required to support the query style supported by current WAIS clients.

Changes to support boolean queries are not likely to be significant as recent changes to PADRE (a choice of different relevance estimation functions including scoring 1.0 if the term is present and the ability to exclude as irrelevant, documents which score less than a threshold value) allow this style of query to be emulated.

Changes to support query functions relating to document attributes such as date, authorname etc. will be more significant. PADRE *component*s are presently defined in a more restricted way and need generalisation. Successful use of document attributes imposes requirements on the markup of the documents themselves.

Finally, many WAIS clients provide the ability to expand queries using relevance feedback. Although PADRE does not include explicit support for this technique, relevance feedback is essentially a client-side function which is unlikely to benefit greatly from parallelisation. If built into a particular WAIS client, the use of relevance feedback should not place any additional demands (other than processing time) on PADRE. The relevance feedback style of interaction, in which the searcher instructs the retrieval client to derive a new query using retrieved documents as models is something which could be useful in other, non-WAIS, PADRE clients.

# 5 Estimating Load Handling Capability

It has not yet been possible to collect experimental data on how many interactive clients can be supported in everyday use by a given PADRE. We are not aware of any standard benchmark of multi-user document retrieval performance, though in our Internet-dominated world, server benchmarking would be quite broadly useful.

It seems very likely that the number of clients which can be simultaneously handled by PADRE will depend upon the following factors:

1. Whether or not PADRE uses full-text scanning, memory-resident index or super dictionary methods.
2. Using full-text scanning method, the size of the collection.
3. Using super dictionary method, the number of subcollections.
4. The complexity distribution for queries. If queries with many terms or queries containing computationally expensive terms such as regular expressions are encountered more than infrequently, long and variable response times may be expected and micro- scheduling would be indicated.
5. The pattern of interaction by researchers. How does the time taken to construct a query and digest its results compare with the PADRE time to process it? How frequently are queries modified in minor ways (taking very little human time) and resubmitted? It is expected that interaction patterns will vary considerably with the type of query interface.

In a bureau service, it may be necessary to implement timeouts on searches or to restrict the complexity of queries available to ordinary users in order to ensure reasonable response. Users may be unimpressed by having to wait long periods while one of their number searches ten gigabytes of documents for occurrences of anagrams of Jim Morrison!

# 6 Conclusions

Substantial progress has been made toward rendering PADRE capable of operating as a multi-user server on the Internet or other TCP/IP network. Prototype implementations of various key elements of a successful multi-user system exist and their performance is encouraging. A range of user interface styles seems likely to improve the accessibility of PADRE and to improve the productivity of researchers using it. As yet, there has been no opportunity to study and characterise multi-user performance.

In the future we hope to benchmark (and further improve) multi-user performance of PADRE/MUFTI and to complete the implementation of effective and industry-standard query interfaces.

## ACKNOWLEDGEMENT

The support of the ACSys Co-operative Research Centre and the ANU-Fujitsu CAP project is gratefully acknowledged. Our thanks to NIST in the USA and various copyright holders for access to the TIPSTER data collection.

# References

[1] S. Ball, S. Uhler and J. Levy `http://pastime.anu.edu.au/SurfIt`.

[2] D.K. Harman *Proceedings Of The Third Text REtrieval Conference (TREC-3)*, US Department of Commerce, NIST Special Publication 500-225, (Apr 1995).

[3] D.A. Hawking 'High Speed Search of Large Text Bases On the Fujitsu Cellular Array Processor,' *Proceedings of the Fourth Australian Supercomputing Conference* pp. 83-90. Gold Coast, Australia, (Dec 1991).

[4] D.A. Hawking and P.R. Bailey "Towards a Practical Information Retrieval System For The Fujitsu AP1000)," in *Proceedings of the Second Fujitsu Parallel Computing Workshop* paper P1-S, Kawasaki, Japan, (Nov 1993).

[5] D.A. Hawking and P.B. Thistlewaite 'Searching For Meaning With The Help Of A PADRE', *Proceedings Of The Third Text REtrieval Conference (TREC-3)*, US Department of Commerce, NIST Special Publication 500-225, (Apr 1995)

[6] D.A. Hawking 'PADRE — A Parallel Document Retrieval Engine', in *Proceedings of the Third Fujitsu Parallel Computing Workshop* paper P2-C, Kawasaki, Japan, (Nov 1994).

[7] D.A. Hawking and P.R. Bailey 'Communing With The Infinite — Will A Terabyte Bring The PADRE To His Knees?', submitted to the 19th Australasian Computer Science Conference (ACSC'96)

[8] World Wide Web Consortium `http://www.w3.org/`.

[9] A. Tridgell and D. Walsh 'The HiDIOS Filesystem' in *Proceedings of the Fourth Parallel Computing Workshop* paper ??, Imperial College, London, (Sep 1995).

[10] A. Tridgell and D.A. Hawking 'Fast String Search For Multiple Patterns' manuscript in preparation