



URSA: The Unicode Retrieval System Architecture

*Multilingual and Translingual Information
Retrieval using Unicode*

Mark Davis
Computing Research Lab, New Mexico State University
<http://crl.nmsu.edu/Research/Projects/tipster/ursa>

URSA, the Unicode Retrieval System Architecture, is a text retrieval architecture that supports multiple information retrieval, language instruction and corpus analysis applications at New Mexico State University's Computing Research Lab. The system has been developed under contract with the US Government. It supports tokenization, indexing and multiple retrieval techniques, including BOOLEAN, Natural Language, Contextual and special statistics retrieval techniques. We have implemented several large-scale demonstration systems using URSA, and tested performance across a range of text encodings and collection sizes.

In this presentation, I will

- Describe the history of URSA and its role in CRL research and development.
- Give an overview of the architecture describing its major components.
- Describe the problems that arise in information retrieval across different languages, with special attention to the problems of morphology and segmentation in Asian languages and solutions in URSA.
- Examine the costs of Unicode support in large-scale IR engines like URSA.
- Overview different retrieval modalities supported by the system.

I will conclude with demonstrations and discussions of applications built on the URSA engine.

Definitions

- **Multilingual Retrieval:** retrieval in languages beyond English
- **Translingual Retrieval:** retrieval where query and document languages differ
 - also: Crosslingual Text Retrieval
 - also: Cross-Language Information Retrieval (CLIR)
- **Corpus Analysis:** Statistical and linguistic analysis of text across large document collections?
 - What words characterize hate speech?
 - What words co-occur with this word depending on meaning?
 - Is this word rare or common?

URSA was developed to support research and development along several avenues. Several DARPA efforts have been focussed on text processing technology (TIPSTER I, II and III) and CRL has participated in each of them. The areas of special interest have been

- Developing multilingual text processing technology.
- Developing multilingual text retrieval technology.
- Developing multilingual text display and editing technology.
- Researching how people can use multilingual text technology effectively.
- Researching machine translation techniques.
- Developing interactive language learning technologies.
- Developing machine translation systems.
- Developing “light-weight” translation technologies.
- Researching translingual retrieval technologies.

URSA was an outgrowth of these various, disparate--but tightly coupled--needs, based on the growing consensus that using Unicode has the potential to radically simplify the treatment of multiple languages.

History

- 1992: First English-Japanese translingual experiments at NMSU
- 1994: Text processing goes mainstream
- 1995: SIGIR hosts CLIR special session
- 1996: TREC hosts CLIR track
- 1997: URSA effort begins
- 1997: Numerous URSA applications and experiments
- 1997: Commercial cross-language capabilities begin to emerge

In 1992, CRL won a small exploratory grant to research using novel methods to retrieve Japanese documents given English queries. The techniques involved learning from “parallel” English-Japanese document collections. Shortly thereafter, we received several grants to develop a translator’s workstation that operated on Spanish and English, but which was quickly expanded to additional languages. CRL’s Unix/MOTIF text display and editing widget collection matured simultaneously through this period. The explosive growth of the World Wide Web around 1994/95 suddenly thrust several rather obscure topics--Text Retrieval and Machine Translation--into the spotlight. Shortly thereafter, worldwide interest in translingual text retrieval began to grow, with special sessions at SIGIR (Special Interest Group of the ACM in Information Retrieval) and AAAI (American Association for Artificial Intelligence) on what was then called Crosslingual Information Retrieval (CLIR).

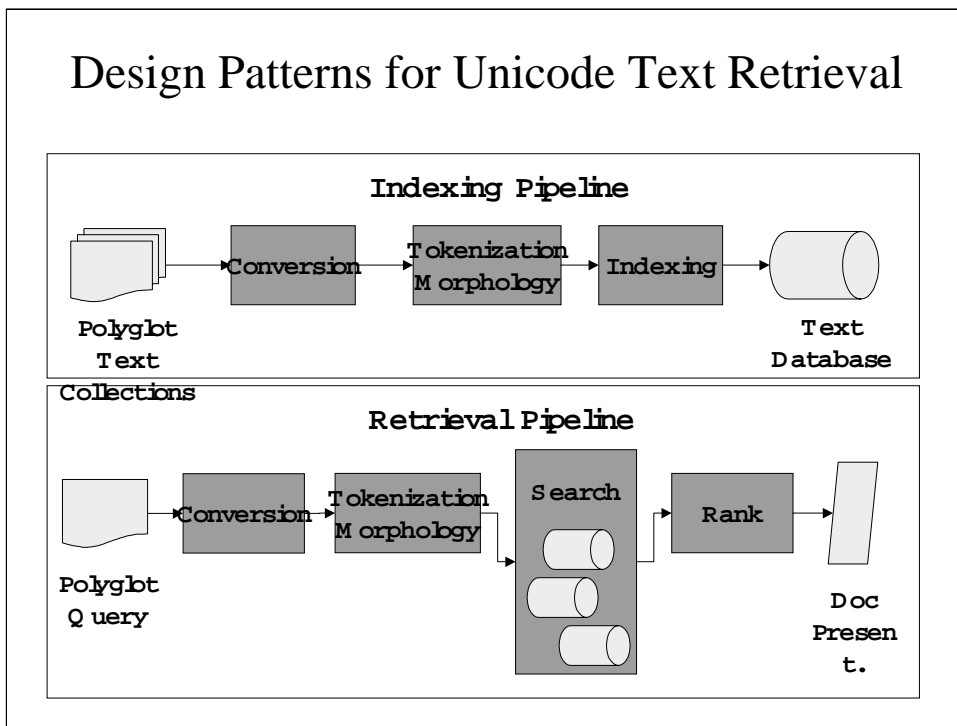
The URSA effort began in 1997, with primary development occurring in late 1997. The core indexing and retrieval engine was functional by May 1997 and robust, large-scale applications were operational in the summer of 1998. Around the same time, commercial vendors began selling some multilingual (typically localized or monolingual non-English) and even limited translingual technologies.

Research and Engineering Questions

- What role can Unicode play in indexing, retrieval and presentation?
- How does Unicode interact with language-specific issues in multilingual/translingual information retrieval?
- What performance costs and benefits come from Unicode in IR?
- Is machine translation needed for translating queries and documents?
- Under what circumstances can we use gloss or word-for-word translations?
- What IR methodologies can turn dross translations into retrieval gold?

Our work on URSA has been driven by several fundamental research and development issues. Some of these we felt we knew the answers to before we began R&D, but even then some of the answers surprised us! The questions range from rather broad philosophical questions like “How do we design retrieval systems that make use of potentially bad translations to retrieve good multilingual documents?” to more focussed engineering questions concerning the appropriate role of Unicode in passing Chinese segmentation information on to a text indexing engine.

Not all of these questions are fully answered, although we are beginning to understand many of the questions by active experimentation and ongoing refinement of the URSA engine. Perhaps most exciting is that we have a stable, robust retrieval engine that supports several very different retrieval needs, from interactive translingual retrieval to multilingual statistical corpus analysis.



Text retrieval engines generally--but not always--share certain broad thematic similarities. First, they operate on the principal of “inversion”; where a text document representation on a file system can be thought of as mapping from document to the characters, words and document structure, a retrieval index can be thought of as having “inverted” that relationship, so that we now can go from words to their documents. Further, conflicting application demands determine differing needs for determination of what constitutes a word, under which circumstances word forms match, and how documents that contain differing distributions of words are to be ranked as to success in fulfilling a query.

The URSA system follows the general model of inversion for creating an index that can supply token to document mappings. A second stage handles retrieval from multiple indexes and ranking of retrieval results. Critical components include the tokenization/morphology stage that is applicable to both the indexing and retrieval pipelines. Tokenization is the determination of “meaningful tokens” for indexing. Morphology is analysis of surface forms to determine their root forms (and sometimes part-of-speech). These are very complex and somewhat controversial topics across the World’s languages, and topics that are highly application dependent: retrieval may need to operate on surface forms and not consider root forms, or error rates for tokenization may be so high that it can’t be considered reliable enough for retrieval purposes.

Tokenization, Segmentation and Morphology

- What is a good token for text retrieval purposes?
 - Example: “Clinton’s” should match “Clinton” if we are interested in finding articles about Clinton, but should only match occurrences of “Clinton’s” if we are interested in building a language workbench for studying how words are used in different contexts.
 - Example: “semicontinuous”
 - should it match “continuous”?
 - should it match “continuum”?
 - should it match “continue”?
 - Continues? Continuing?
- What is a good token in Chinese where “words” are much more difficult to define?
 - therestands “the rest and s...” or “there stands ...”

Tokenization, segmentation and morphology reference the complexity of determining the best items for indexing. These properties vary across languages and applications. Tokenization in English, for example, is hampered by the fact that abbreviations sometimes contain unexpected period sequences. Proximity retrieval bounded to the same sentence can be incorrect if the abbreviation is misperceived as a sentence bound. Further, we expect that “truffles” and “truffle” will both be retrieved using a query like “Get me documents about truffles” but we may also expect “continuous” to be retrieved when “semicontinuous” is the query term. English tokenization and morphology is difficult enough, but other languages exponentially compound the problem. In Arabic, for example, the root token is often far removed from the surface form, as in Korean. German compounds nouns together into long words, making easy matches difficult. Chinese, furthermore, doesn’t even divide up “words”. Ideograms in Chinese have abstract meanings, but they also can be used as phonetic devices for foreign words and phrases, making sequences of them very “word-like”.

Unicode simplifies the handling of tokenization by providing merged assignment of punctuation across differing languages, for example, but in URSA we have had to extend the role of certain codepoints to support segmentation and tokenization.

Asian Language Indexing

- Multiple models:
 - N-Gram+ indexing of Chinese and Japanese
 - XYGFSDH = XY, YG, FS, SH, HD
 - DGUjdkjHDKJD = DG, GU, jdkj, HD, DK KJ, JD
- Segmentation markup using zero-width space (0x200b) on UCS2 files
- Some language markup for preserving identities of Chinese Han, Japanese Kanji and Korean Hanja.

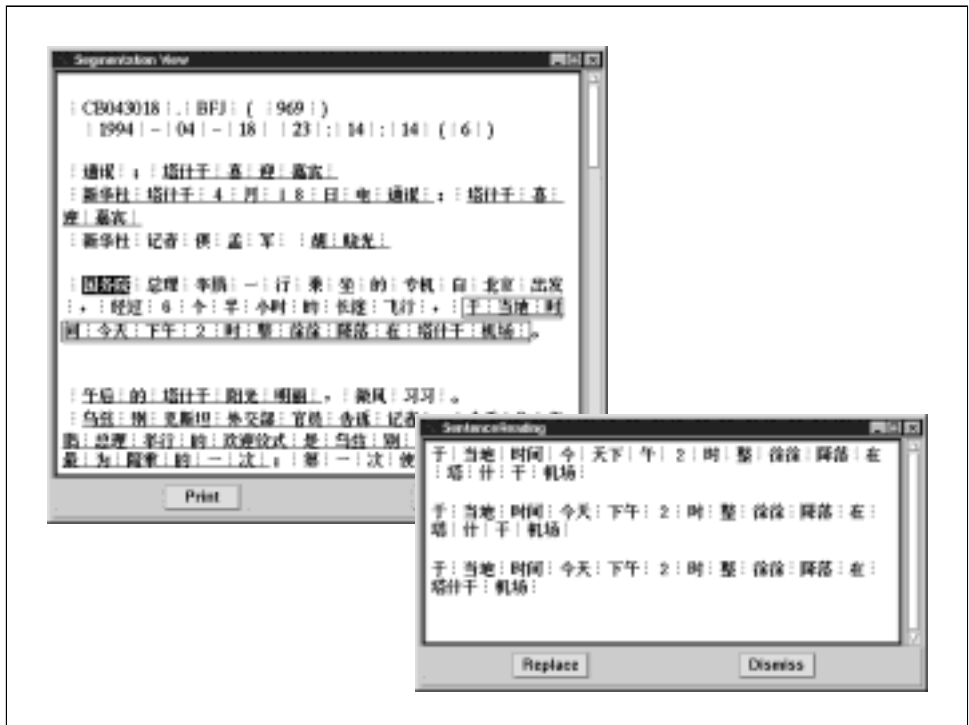
The problems of tokenization, segmentation and morphology are acutely illustrated by the Asian languages, especially Japanese and Chinese. URSA's tokenization process for Asian languages can operate in several modes. We have implemented this flexibility because it is currently unknown (empirically) what the best approach is for handling retrieval in these languages. We support, for example, indexing all possible sequences of 1 or more ideograms in Chinese and Japanese.

We also support treating sequences of Hiragana and Katakana in Japanese as separate tokens. We further support indexing Chinese and Japanese using special markup in the UCS2 stream based on zero-width spaces (0x200b) where a zero-width space serves to signal the start of the next token. Various methods can be used to segment Chinese and Japanese and insert zero-width spaces in the UCS2 stream.

We also support the preservation of language identity for ideograms by three codepoints from the private-use area of Unicode. By prefacing tokens with these codes we insure that unified Chinese, Korean and Japanese only match queries in their specific languages. Again, this is an implementation-driven decision, however, and it is possible to imagine applications that would want tokens without this language markup on the tokens.



This slide shows a Unix-based text editor from CRL's Oleada project for language instruction and translation support. The Chinese document has been segmented in the next slide.



The segmentation window shows the same document segmented. For display and editing purposes the segmentations are shown with vertical separator bars. The third text paragraph with the dark highlighting is shown in again with multiple “readings” in the lower right hand window. Automatic segmentation processes are often not 100% certain (indeed, Chinese linguists appear to only agree around 80% of the time). Multiple readings are useful in interactive settings. For indexing, however, we choose the most probable reading. This means that there is the very real possibility that the query tokens may not match the indexing tokens because the segmentations were different! This remains an active area of research.

The segmentation is represented by a zero-width space in the UCS2 data stream and each segment is prefaced by a one of three codepoints from the private use area to anchor it by language.

The Costs of Unicode for IR

- UCS2 storage of lexicon (up to 2X the size of the native lexicon)
- Lexicon sizes:
 - For 200 Mb ISO8859-1 Spanish newswire:
 - 200,000 unique tokens
 - average of 5 bytes per token
 - 1 Mb ISO8859-1
 - 2 Mb UCS2
- Index sizes:
 - For 498.2 Gb UCS2 text
 - Index is 91.66 Mb

Unicode provides a unified framework for tokenization, but also incurs some costs. Perhaps the most pressing issue is how is Unicode IR going to explode index sizes. Indexes for IR systems are already large, requiring between 20% and 200% of the size of the original texts, depending on the needs of the application. The increasing availability of inexpensive mass storage has decreased the importance of index size somewhat, but at the same time the desire to index gigantic information resources has grown as well (I.e., the WWW).

In URSA, we have the flexibility of indexing to the among the smallest indexes currently reported in the literature. The additional space requirements for indexing UCS2 versus ISO8859-1 or some other character encoding is therefore limited to the costs of UCS2 representations of the unique tokens. For ISO8859-1, that means double the size of the lexicon. But what does this really mean for large text collections? Well, an examination of 200 Mb of Spanish yields around 200,000 unique tokens. In Spanish, there are approximately 5 bytes per word, so the lexicon can occupy as little as 1 Mb of disk. In practice, the disk occupancy may be slightly larger than that due to btree overhead or other counts info, but it serves as a useful estimate. The UCS2 version of the same strings is therefore 2 Mb. Now, URSA will generate maximally-compressed indexes that allow nearly instantaneous natural language queries for this text in around 90 Mb.

The Costs of Unicode

- The space cost of Unicode is:
 - $2/92$ versus $1/92 = 1.1\%$ larger!
- Time costs are minimal and are swamped by the costs of converting complex scripts from native encodings.

There is no significant penalty for using
Unicode in an IR system in terms of
processing time or space!

Using the preceding estimates, then, we can see that the costs of Unicode indexing amount to around 1.1% difference from non-Unicode indexing with the same engine. The performance figures scale very well, too, with index sizes of 310 Mb for 1.53 Gb (averages of 20% across evaluations). Further the number of unique tokens per unit text drops off by Zipf's law as we index more related text in the same language, decreasing the load of the lexicon on the index as a whole.

Finally, the speed of URSA is 635 Mb/Hour on a dual-processor Sun Enterprise server under moderate load. This is quite comparable to modern high-performance retrieval engines, and is inherently parallelizable given a scheme for merging results.

Our general conclusion is that, neglecting the space required of the original text in UCS2 form (the preservation of this text is often unnecessary), the costs of a fully Unicode retrieval engine in terms of time or speed are very low.

URSA Retrieval

- Natural language queries
- Ranked retrieval using 75,600 ranking variants
- Boolean search
- Context Search operators (A occurs within X words of B)
- Index statistics
 - How many words/tokens were indexed?
 - How many unique words/tokens were indexed?
 - How many documents contain this word/token?
 - What is the average number of times a word occurs in a document?
 - How does this word compare to that average?

The retrieval pipeline in URSA has great flexibility for differing retrieval needs. WWW-style retrieval with simple keyword or short-phrase retrieval is supported with 75,600 variant retrieval parameterizations described by an octet of parameters that describe how to weight term occurrences in documents, how to normalize term occurrences by document lengths, how to weight term occurrences in queries, etc. This is the meat of information retrieval research for the last 5 years and has had a significant impact on the performance of IR systems for WWW retrieval.

The system also supports BOOLEAN and CONTEXT searches in a limited manner at moment, but expanding all the time.

Finally, the system supports extensive statistics reporting for corpus analysis, with API calls and tools to retrieve the counts and relative counts of almost any aspect of the system (tokens, surface forms of tokens, documents, sentences, etc.).

Demonstration Systems

- Xcount: Word statistics application
- IKWIC: Concordance search of large collections and parallel texts
- J24: WWW document retrieval with thumbnail visualization
 - Some empirical support that thumbnails allow faster judgments of document relevance than
- Arctos: Interactive translingual text retrieval with WWW MT support
 - Some empirical evidence that Arctos can be used to retrieve and understand foreign language documents

These various demonstration system all use URSA at some level. XCOUNT and IKWIC have been delivered to Government sponsors and will be used in language learning environments.

J24 is being used in ongoing interactive text retrieval research at NMSU focussed on trying to improve user interfaces for retrieval. The system currently operates on English texts and produces thumbnail views of document contents with highlighted query terms.

Arctos is being used to develop and expand interactive translingual retrieval. Users enter queries in English and retrieve documents in German, Italian and French. The can use online resources to improve their translingual queries by looking-up translations of phrasal terminology and expanding or collapsing parts of the query translation. The J24 interface then presents the retrieved multilingual documents for browsing and used online resources to machine translate the resulting documents.

