

# Sary: Reusable Components and Tools for Searching Large Corpora

Satoru Takabayashi

Graduate School of Information Science,  
Nara Institute of Science and Technology

Email: satoru-t@is.aist-nara.ac.jp

Tel: +81-743-72-5248, Fax: +81-743-72-5249

Sary Web: <http://sary.namazu.org/>

## 1 Introduction

Since corpus-based natural language processing has to deal with large corpora, efficient searching of the large corpora is inevitably necessary. For example, one might want to examine how a word or a phrase is used in the large corpora or to collect frequencies of all terms in the large corpora. Our system *Sary* solves these problems by providing fast full-text search facilities for a single large text on the order of 100 MB using a data structure called *suffix array*[2]. *Sary* provides not only useful tools for searching large corpora, but also provides well-implemented libraries as reusable components.

## 2 System Overview

Our system *Sary* consists of three parts: `mksary` command, `sary` command, and libraries. One can construct the suffix array for a certain text file with `mksary` command and can search the text file with the constructed suffix array with `sary` command. `sary` provides facilities similar to `grep` command including case insensitive searches. *Sary* has the following characteristics:

- **Fast** Since searching of the text file with the suffix array is performed in a binary fashion, searching time grows logarithmically to the length of the text. Thus, fast full-text searching can be achieved for large text files. Additionally, *Sary* is written in C for maximizing the performance. Performance evaluation is discussed in Section 4.
- **Reusable** Not only does *Sary* provides C library, but also libraries for higher level programming languages Perl<sup>1</sup> and Ruby<sup>2</sup>. One can use the libraries for developing his/her own software. *Sary* supports UTF-8 encoding for handling various languages.
- **Free** *Sary* is a free software under the terms of GNU Lesser General Public License<sup>3</sup> so that everyone can

<sup>1</sup><http://www.perl.com/>

<sup>2</sup><http://www.ruby-lang.org/>

<sup>3</sup><http://www.gnu.org/copyleft/lesser.html>

use, copy, distribute, and/or modify it.

Another suffix array library *SUFARY*[3] has been available since 1997. The difference between *Sary* and *SUFARY* is the design philosophy. *Sary* is written in C but in object-oriented fashion. *Sary* aims for maintainability, extensibility, robustness, and reusability. In fact, the library of *Sary* does not use global or static variables at all except the version number for making all functions reentrant so that the library can be used in multi-threaded programming.

## 3 Suffix Array

A suffix array is a data structure designed for efficient searching of a large text. The data structure consists of an array containing the indices to the text suffixes sorted in lexicographical order. Each suffix is a string starting at a certain position in the text and ending at the end of the text. Searching the text can be performed by binary search using the suffix array because the suffixes can be seen as the sorted suffixes in the lexicographical order.

Suppose that we have a sample text “abracadabra” and wish to construct a suffix array for the text. First, we assign index points to the sample text. Index points specify positions where search can be performed. In our example, index points are assigned to every character. It is possible to assign index points every word or every line.

Text	a	b	r	a	c	a	d	a	b	r	a
Index	0	1	2	3	4	5	6	7	8	9	10

Second, we should sort the index points according to their corresponding suffixes. Suffixes are sorted in lexicographical order and index points are reordered according to their corresponding suffixes. The following figure shows the result of sorting.

