Kolokacje 1.0 User’s Manual

Aleksander Buczyński
Translated by Radosław Moszczyński

Contents

1 Abstract 1

2 Definitions 1

3 Structure overview 1

4 Crawler 2
  4.1 Short overview ............................................. 2
  4.2 Usage ...................................................... 3
  4.3 Initial URL and download queue ......................... 4
  4.4 Accepted file name extensions ......................... 4
  4.5 Long text files ........................................... 5
  4.6 Ignored web pages ....................................... 5
  4.7 Ignored element tags .................................... 5

5 IndexBuilder 5
  5.1 Short overview ............................................. 5
  5.2 Usage ...................................................... 6
  5.3 Ignoring selected words .................................. 6
  5.4 Building indices of large corpora ....................... 6

6 Archive queries 6
  6.1 Word query ................................................ 6
  6.2 Collocation query ......................................... 7
  6.3 Context query ............................................. 8
  6.4 Relationship between queries ............................ 8
  6.5 General and asterisk queries ............................. 8
  6.6 No results? ............................................... 9

7 Single user interface – SAManager 9
  7.1 Overview .................................................. 9
  7.2 Usage ...................................................... 9
    7.2.1 Plain text ............................................. 9
    7.2.2 CSV .................................................. 10

1
1 Abstract

Kolokacje is an Internet crawler and a collocation search engine. It is made up of several modules which may be executed via command-line or a graphical interface.

Usage of the program and individual modules:

```
java kolokacje.standalone.SAMain [dir]
java kolokacje.crawler.Crawler dir [var1=value1 var2=value2...]
java kolokacje.index.IndexBuilder dir [var1=value1 var2=value2...]
java kolokacje.server.PrettyPrinter dir [var1=value1 var2=value2...]
java kolokacje.server.QueryServer dir [var1=value1 var2=value2...]
java kolokacje.standalone.SAManager dir [var1=value1 var2=value2...]
```
2 Definitions

In the current manual, the term *archive* indicates a user selected working directory in which the program stores downloaded texts, indices it has created, summaries and temporary files. You can place additional configuration files in the directory to adjust the behavior of individual modules.

When reference is made to a specific file in an archive, please bear in mind the fact that you can modify the configuration and rename the file in question. However, this is rather uncommon, so the manual refers to default file names, e.g. *addresses.acc*, instead of *file whose name is defined by the addressFile variable*.

3 Structure overview

The program is made up of three modules which reflect processing stages which a downloaded text undergoes — Crawler, IndexBuilder and Archive. The last one has many versions which correspond to various designs, or design components, of the user interface.

Each of the three modules may be executed via command line or a graphical interface provided by SAMain, which is itself a separate module.

Crawler searches the Internet, selects pages, picks out individual elements of a page, converts and saves them in consistent format on a local disk.

IndexBuilder reads previously downloaded and converted text files from the archive. It then updates the directory with collocation index files, which make it possible to quickly access statistical data and search for specific words or collocations.

The next stage consists of accessing data stored in the archive, computing statistical scores and generating human-readable reports. This may be done in two ways, one oriented towards posting the results on the Internet (PrettyPrinter and QueryServer) and the other towards single users (SAManager).

SAManager provides a Java-based (AWT / Swing) graphical interface for accessing the archive, oriented towards single users working upon a collection of texts. Owing to its design, you can access the basic functions of the archive module with just a couple of mouse clicks.

The WWW interface is combined of PrettyPrinter and QueryServer modules. PrettyPrinter generates web pages containing basic summaries and reports (in plain HTML) which feature links to dynamic PHP pages. These allow you to issue more specific queries which are then handled by the QueryServer. E.g., clicking on a collocation in a statically generated score summary, or a comparison of such summaries, directs you to a dynamically generated list of sample contexts of the collocation. Clicking on a word in a statically generated frequency dictionary directs you to a dynamically generated score summary of the word’s collocations.

The division into two “network” submodules is a consequence of the fact that
4 Crawler

4.1 Short overview

Crawler copies the text content of selected web pages to a selected archive, during the process of which the content is converted into uniform encoding and divided into elements which roughly reflect the HTML / XML structure. The elements may undergo a selection process, short or repeating ones may be rejected. The element division is reflected in line division.

The module may be stopped at any point and then reactivated. It is also possible to download several versions of the same page into the archive (e.g. a daily news service).

The module creates a local directory containing uniformly encoded files (UTF-8 by default), which represent the text content of the processed pages. The filenames are of xxxx.txt type, where XXXXX is a natural number (supplemented by zeros in the front if necessary, so that it always consists of five digits): e.g. 00001.txt, 00002.txt, 00003.txt, etc.

Additionally, the directory also contains the following, also in the form of text files:

- a list of pages represented by the created text files - indexed.url;
- a list of ignored URLs, containing the causes of rejection, which makes it possible to diagnose unexpected range of crawling (being a result of e.g. a mistake in the regular expression describing a web page, inability of the program to recognize the hyperlinks utilized by the author or obscure filename extensions) - ignored.url;
- a dictionary of the elements’ initial bits, if the duplicate filter is enabled (which is recommended) - indexed.elm;
- a queue of unexplored links, if the program is externally terminated or the specified limit reached, which makes it possible to resume crawling later on - queue.url.
4.2 Usage

Synopsis:

java kolokacje.crawler.Crawler dir [var1-value1 var2-value2...]

Examples:

java kolokacje.crawler.Crawler emacs \
start=http://jameshornton.com/emacs/node/

java kolokacje.crawler.Crawler md \
start=C:\Mojedo1 inputEncoding=windows-1250

dir – the archive directory, defined in 2. If the supplied name points to a non-
existent directory, the module will try to create it.

var – one of the configuration variables defined in config.ini.

value – new variable value (which overrides the value declared in config.ini
for the time of current execution).

You can take advantage of the possibility to supply configuration variables
via a command line parameter in the case of a single execution or one generated
by an external program, but it is nevertheless recommended to create a local
configuration file.

Significance of individual variables and their default values are described
in 11.

The variable which is used especially frequently while executing the module is
start, which supplies the address of a web site or the path of a file or a directory
in which the program is to start crawling. If start points to a local directory,
the module will treat it as if it were a web page containing links to all the files
and directories contained within it.

4.3 Initial URL and download queue

The web pages which are to be downloaded are specified in addresses.acc
which is located in the archive directory. There are three ways to define a down-
load queue.

If the regexpAddressesTester option is enabled, the program will assume
each line of addresses.acc to be a regular expression describing the web pages
to be downloaded. Should a found URL fit any of the expressions, it will be
added to the download queue.

If regexpAddressesTester is disabled, the program will assume each line of
addresses.acc to be a prefix describing the web pages to be downloaded. Should a found URL begin with any of the prefixes, it will be added to the down-
load queue.

If addresses.acc is not defined by the user, the only accepted URLs will
be the ones whose prefix equals start.
If `addresses.acc` is defined by the user, the only accepted URLs will be the ones which satisfy one of the rules located in the file.
If the initial URL does not contain any specific file name, but rather the name of a server (and perhaps some directory), it should and with a slash (/).

### 4.4 Accepted file name extensions

Because taking file name extensions into account within the regular expression which describes a web page would be complicated and introduce obscurity, a separate configuration file, `extension.acc`, was created to store the list of accepted extensions.

By default the program downloads all the files that have one of the following extensions: `.htm`, `.html`, `.xhtml`, `.xml`, `.txt`, `.php`, `.asp`. In order to make the program accept URLs ending with a slash (/), the latter is also included in the default list.

The current version does not change its behavior depending on a file’s kind, so e.g. it looks for the same tags while recognizing the language, no matter if it is HTML or XHTML.

### 4.5 Long text files

Because the maximal size of an element is limited and the program is focused on processing HTML files, the current version might have problems with analyzing text files longer than 32768 characters and lacking HTML/XML tags.

It can be dealt with by supplying a long text file with “dummy” tags, e.g. `<chapter n>` in the beginning of every chapter, should the file contain a book.

### 4.6 Ignored web pages

The URLs of web pages that were not qualified for download are recorded in `ignored.url` and tagged, which makes it possible to determine the rejection reasons.

<table>
<thead>
<tr>
<th>Tag</th>
<th>Rejection reason</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>malformed</code></td>
<td>the URL is malformed, e.g. because of an undefined protocol</td>
</tr>
<tr>
<td><code>syntax</code></td>
<td>the URL is illegal, e.g. because of some characters which should not normally occur in an URL</td>
</tr>
<tr>
<td><code>extension</code></td>
<td>the file’s extension is not listed in <code>extension.acc</code></td>
</tr>
<tr>
<td><code>address</code></td>
<td>the URL does not fit any of the patterns specified in <code>addresses.acc</code> or <code>start</code></td>
</tr>
<tr>
<td><code>IO</code></td>
<td>connection problems or inability to read the file</td>
</tr>
<tr>
<td><code>encoding</code></td>
<td>obscure encoding</td>
</tr>
</tbody>
</table>
4.7 Ignored element tags

Some tags may be declared to have only a formatting function (by default it is e.g. font). Such tags will be transparent for the download module and will not be taken into account during segmentation.

5 IndexBuilder

5.1 Short overview

IndexBuilder loads the text files contained within a specified directory, then creates a collocation index and an indexed suffix array.

The collocation index contains information about pairs of words occurring near each other in a text.

Occurrence lists of a word are created by means of summing up all the occurrences of word pairs containing the word in question. Owing to the lists, the collocation index also serves the purpose of an occurrence index.

The suffix array contains all the words found in the corpus (save for duplicates). The index, which is “parallel” to the array, enables you to efficiently look up words containing a selected string of characters.

5.2 Usage

Synopsis:

```java
java kolokacje.index.IndexBuilder dir [var1-value1 var2-value2...]
```

Examples:

```java
java kolokacje.index.IndexBuilder emacs
java kolokacje.index.IndexBuilder emacs minWordLength=4
```

dir, var, value have the same significance as in Crawler.

5.3 Ignoring selected words

Some words may be declared to be transparent, which means they will be omitted during indexing. In order to do that, you need to create a file named ignore.wrd in the archive directory before building the index, containing a list of the words, each in a separate line. An alternative file may be declared via the whiteWordsFile variable.

You can also make the program ignore words that are shorter than a specified length. This is controlled via the minWordLength variable.
5.4 Building indices of large corpora

When building the index of a large corpus (500K words or more), the program might report an OutOfMemoryError. If this happens, you should remove the default limit of stack size and memory available to the Java virtual machine. This may be done by executing Java (not the program itself) with the -Xmx<size> parameter, e.g.:

```
java -Xmx228m kolokacje.index.indexBuilder emacs
```

228m in the example means that the virtual machine will be able to use up to 228 megabytes of memory.

When setting a new memory limit, it is not recommended to exceed the amount of physically available RAM, as it may considerably slow down the program.

6 Archive queries

6.1 Word query

You can issue either general word queries, about all the words present in the text (frequency dictionary) or specific ones, about a specific word or word family (e.g. *word* will query about words containing the "word" root, whereas *a - words ending with an "a").

The most prominent parameter of a word query is a white-space separated list of word families you are interested in. Each of the list members may start or end with an asterisk, which signifies zero or more characters. When the parameter is empty, the query is assumed to be general.

Additional query parameters:

- minimal number of occurrences – words which occur less frequently than a declared value are ignored in the results
- exclude numbers – defines whether numbers are to be ignored. Every word starting with a digit is regarded to be a number (including e.g. 3D, 121a)
- word order – the implemented orders are alphabetical, frequency based, and a tergo

6.2 Collocation query

You can issue either general collocation queries, about all the collocations present in a text, or specific ones, about collocations formed by a specific word or word group. When the parameter is empty, the query is assumed to be general.

Additional query parameters:

- tests, whose values are to be computed and displayed in the results
the test according to whose value the results are to be sorted. If the **all rankings separately** option is chosen, a score summary comparison will be created (see below)

- minimal number of occurrences – collocations which occur less frequently than a declared value are ignored in the results
- collocation direction (only left, only right, both left and right) – in the case of specific queries it defines the group of collocations which is to be included in the results
- exclude proper names – whether proper names are to be excluded from the results. Proper names are defined as collocations whose all occurrences in a corpus start with a capital letter
- only proper names – whether the results are to exclude collocations which do not meet the proper name criteria

Depending on the selected sorting order, the result will be:

- a collocation score summary, sorted according to the result of the selected test. The columns represent: the collocating word pair, the first word's number of occurrences, the second word's number of occurrences, the results of individual tests;
- a comparison of score summaries according to various tests. The columns represent score summaries (with word pairs sorted on the basis of their scores, in decreasing order) according to results of individual tests.

### 6.3 Context query

The main parameter of a context query may be either a word or a collocation (a white-space separated word pair). You are allowed to use an asterisk in both cases, to indicate an unknown part of the word, but you should use this possibility sensibly, as computing the answer for a context query like *a* *b* might take a very long time to complete. The result is displayed in the form of text fragments containing the word or the collocation.

Additional query parameters:

- the length of the left/right context, defined by number of characters.
  The context may actually be a little shorter, as it is cut at the nearest white space in order not to display incomplete words

### 6.4 Relationship between queries

Both the individual and network interfaces try to facilitate issuing queries related to the results of the previous one. Therefore:

- in word query results, clicking on specific buttons or links (**samples** or **collocations**) issues a context or collocation query for the selected word
in collocation query results, clicking on a collocation issues a context query
in context query results, clicking on the file name opens the original file.
In the network version, there is also a link to the web page on the basis
of which the local file was prepared.

6.5 General and asterisk queries

Even though suffix arrays greatly speed up looking for words containing a spec-
ified root, it is not recommended to overuse the asterisk, as the number of words
that fit a specified pattern is usually much larger than you expect.
General queries are faster than specific ones, therefore it is not recommended,
although possible, to use the ** pattern to define any word.
Examples:

- it is better to issue a general query about all the words than a specific
  query about words that fit the ** pattern
- it is better to issue a general query about all the collocations than a specific
  query about collocations of **
- it is better to issue a query about X's contexts than about collocation
  contexts for X **
- it is better not to issue queries about collocation contexts for ** ** at all

6.6 No results?

If you issue a word or collocation query and get no results (or much less results
than you expected), it may be a result of:

- a misspelled word in the query or in the processed text;
- overrestrictive result filters; it is especially recommended to examine the min-
  imal number of occurrences, because in the case of small corpora the default
  value (3) filters out most word pairs.

7 Single user interface – SAManager

7.1 Overview

SAManager provides a graphical interface for querying the archive and browsing
the results.

You have to type the query into the text field at the window’s bottom. Then, click on one of the buttons: Find words, Find collocations or Find samples. In the case of word and collocation queries, it is possible to leave the text field empty – the issued query will be then a general one.
Significance and parameters of particular queries are explained in 6.
7.2 Usage

Synopsis:

java kolokacje.standalone.SAManager [var1=value1 var2=value2...]

Examples:

java kolokacje.standalone.SAManager emacs
java kolokacje.standalone.SAManager gnupl forceWindowsLocation=1

The results window allows you to save the results on the local disk as either plain text (Save TXT) or Comma Separated Values (Save CVS).

7.2.1 Plain text

If your choice is plain text, the program’s behavior will depend on the type of results:

- in the case of collocation score summaries and their comparisons, the whole list of collocations will be saved, with every word pair in a separate line, sorted according to decreasing prominence
- in the case of words, only the words will be saved, each in a separate line, without the number of their occurrences
- in the case of context samples, the whole contexts will be saved (three columns forming a sample)

7.2.2 CSV

If you choose Comma Separated Values, all the columns of the result table are saved. Each row is represented as a separate line, the content of every cell is placed in inverted commas and separated from the others by commas. Inverted commas within cell content are replaced by double quotes.

Data saved in this way may be imported into a spreadsheet (e.g. OpenOffice Calc) for further processing.

8 Single user interface - SAMain

8.1 Usage

Synopsis:

java kolokacje.standalone.SAMain
java kolokacje.standalone.SAMain dir
8.2 Archive selection

If you do not supply the name of the archive you wish to work upon as a command line parameter, you should choose it right after the module is executed. It is also possible to create a new archive by supplying a name of a non-existent directory.

8.3 Main menu

The main menu appears after you choose the archive. The buttons allow you to:

<table>
<thead>
<tr>
<th>Button</th>
<th>Purpose</th>
<th>Active</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start / resume</td>
<td>executes <strong>Crawler</strong>, assuming that <strong>start</strong> is the URL typed into the</td>
<td>always</td>
</tr>
<tr>
<td>crawling</td>
<td>window above button</td>
<td></td>
</tr>
<tr>
<td>Choose file or directory</td>
<td>opens the file selection dialog; after the choice is made, it executes</td>
<td>always</td>
</tr>
<tr>
<td></td>
<td><strong>Crawler</strong> with the <strong>start</strong> parameter pointing to the chosen file or</td>
<td></td>
</tr>
<tr>
<td></td>
<td>directory</td>
<td></td>
</tr>
<tr>
<td>Edit list</td>
<td>opens the list of words ignored by <strong>IndexBuilder</strong></td>
<td>always</td>
</tr>
<tr>
<td>(Re)build index</td>
<td>executes <strong>IndexBuilder</strong> and builds a collocation index, taking into</td>
<td>if the archive</td>
</tr>
<tr>
<td></td>
<td>account all the parameters set in the panel; then it starts <strong>SAManager</strong></td>
<td>contains at</td>
</tr>
<tr>
<td></td>
<td>(see 7) in a separate window, which allows you to query the new index</td>
<td>least one file /</td>
</tr>
<tr>
<td></td>
<td></td>
<td>web page</td>
</tr>
<tr>
<td>(Re)build and</td>
<td>see above; additionally, the index is saved in the archive</td>
<td>see above</td>
</tr>
<tr>
<td>save index</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Query saved index</td>
<td>opens a previously saved index and starts <strong>SAManager</strong> in a separate</td>
<td>if the archive</td>
</tr>
<tr>
<td></td>
<td>window (see 7), which allows you to query the index</td>
<td>contains a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>collocation index</td>
</tr>
<tr>
<td>Custom config.ini</td>
<td>opens a local configuration file</td>
<td>always</td>
</tr>
<tr>
<td>Archived pages</td>
<td>displays a list of URLs pointing to the pages downloaded to the archive</td>
<td>if the archive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>contains at</td>
</tr>
<tr>
<td></td>
<td></td>
<td>least one file /</td>
</tr>
<tr>
<td></td>
<td></td>
<td>web page</td>
</tr>
<tr>
<td>Change archive</td>
<td>choose a different archive</td>
<td>always</td>
</tr>
<tr>
<td>About</td>
<td>displays basic information about the program</td>
<td>always</td>
</tr>
<tr>
<td>Exit</td>
<td>terminates the program</td>
<td>always</td>
</tr>
</tbody>
</table>
8.4 External editor

An external editor, defined via the editorCommand variable, is used for browsing and editing some of the files, such as the local configuration file, the list of ignored words or the archive files. The program can supply the editor with the following parameters: \{0\} - file name, \{1\} - encoding and \{2\} - character number within the file. If the setEnvironment variable is enabled, the following environment variables are also declared: FILENAME,ENCODING and POSITION respectively.

8.5 Windows location in FVWM

When arranging windows, FVWM does not take into account the so-called window decorations, which makes program windows partially invisible if default configuration is used. In order to avoid this, you have to set the value of forceWindowLocation (see 11) to yes. By modifying the values of locationX and locationY you can influence the initial position of windows.

9 Network interface – PrettyPrinter

9.1 Short overview

First, the module loads a collocation index. Then, on the basis of templates, it creates and saves XHTML files in the archive, which contain basic summaries: the list of downloaded pages, a dictionary of most frequently occurring words, the highest scoring collocations according to all the score summaries and a comparison of the summaries. It is also possible to generate collocation score summaries for especially important words or word groups.

9.2 Usage

Synopsis:

```
java kolokacje.server.PrettyPrinter \ndir [var1-value1 var2-value2...]
```

Examples:

```
java kolokacje.server.PrettyPrinter emacs
java kolokacje.server.PrettyPrinter emacs serverPort-7762
java kolokacje.server.PrettyPrinter emacs wordsMinCount-20
java kolokacje.server.PrettyPrinter gnupl \nmaxRows-100 rowsPerPage-100
```

dir, var, value have the same significance as in Crawler.
9.3 File structure

In order to work properly, PrettyPrinter requires web page templates and a stylesheet. It is most recommended to create archive directories on the same level and in the same directory where the following are located: \texttt{kolokacje} (which contains the program code), \texttt{styles} (which contains the style sheets) and \texttt{php} (which contains the php code that makes it possible to query the QueryServer module and display the results).

If this is impossible or inadvisable, you have to:

- correct the links to the stylesheet in utilized templates;
- correct the link to the stylesheet in \texttt{php/coll.php};
- correct the \texttt{phpRef} variable (which points to the location of \texttt{coll.php} relative to the archive directory) in the configuration file.

The resulting HTML files are always saved in the archive directory.

10 Network interface – QueryServer

10.1 Overview

The module loads a collocation index and starts a query server which listens for incoming queries (submitted through e.g. a PHP interface) on an external port. Each query is served in a separate thread, therefore the server may answer several queries simultaneously. Should an error in handling of a query cause the thread to terminate prematurely, it will not interrupt the work of the server.

10.2 Usage

Synopsis:

\texttt{java kolokacje.server.QueryServer dir \textbar \textbar var=value1 var2=value2...}

Examples:

\texttt{java kolokacje.server.QueryServer emacs}
\texttt{java kolokacje.server.QueryServer emacs serverPort=7762}
\texttt{java kolokacje.server.QueryServer gnupl maxAsterixAmbiguity=100}

\texttt{dir, var, value} have the same significance as in Crawler.
10.3 QueryServer and PrettyPrinter

The query server may be located on a different computer than the HTML files generated by PrettyPrinter. This can be utilized e.g. to prohibit direct access to the query server (i.e. not via queries generated by the WWW interface).

It is important to make sure that:

- the HTML and PHP files are located on the same machine;
- `php/queryServer.php` contains the address of the server (see below);
- the query server is visible to the WWW server;
- the collocation index and the suffix array files are located on the same machine where the query server is;
- `QueryServer` and `PrettyPrinter` are executed with the same value of the `serverPort` variable (it is best to declare it in the local configuration file);
- the whole set of archive text files is available on both computers (this actually is not required, but lack of these files limits the functionality of the program).

10.4 Configuring PHP code

In order to make the PHP pages “see” the query server, you have to supply them with its address. You need to change the value of the `queryServer` variable located in `php/queryServer.php` so that it reflects the address of the computer which runs the `QueryServer` module.

The port number is transmitted dynamically by the URL in the form of the `port` parameter, so you should not include it in the PHP code.

11 Program configuration

11.1 Configuration variables

The configuration variable values may be declared via:

- the graphical interface of `SAMain` (only selected variables)
- a command line parameter (variable-value) of `Crawler`, `IndexBuilder`, `SAManager`, `PrettyPrinter` or `QueryServer`
- a local configuration file (`config.ini` located in the archive directory)
- a general configuration file (`config.ini` located in the `kolokacje` directory, which defines default values)
The priority is reflected in the above order, i.e. values of the local configuration file override default values of the general file, whereas values declared via command line parameters or the graphical interface of SAWin override both configuration files.

Assigning a value to a variable always has the following syntax:

```
variable=value
```

The configuration files allow a white space on one or both sides of the - character, the command line parameters do not.

Within the configuration files, each value declaration must be located in a separate line. Comments may be marked with # or ;. Empty lines are ignored.

In the case of binary (logical) variables, you can use any of the following values: true, yes, on and 1. Analogously, false - no - off - 0.

### 11.2 Sample local configuration file

MS Windows users who are creating a corpus on the basis of local text files may place the following `config.ini` in the archive:

```plaintext
defaultInputEncoding - windows-1250
archiveEncoding - windows-1250
directory - notepad {0}
setEnvironment - no
```

### 11.3 Universal variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Default value</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>archiveEncoding</td>
<td>UTF-8</td>
<td>encoding of archive files and the suffix array</td>
</tr>
<tr>
<td>collocationIndexFile</td>
<td>colls.idx</td>
<td>name of the collocation index file</td>
</tr>
<tr>
<td>indexSummaryFile</td>
<td>colls.ini</td>
<td>name of the collocation index summary file</td>
</tr>
<tr>
<td>suffixBaseFile</td>
<td>suffix.txt</td>
<td>name of the suffix array file</td>
</tr>
<tr>
<td>suffixIndexFile</td>
<td>suffix.idx</td>
<td>name of the suffix array index file</td>
</tr>
<tr>
<td>urlListFile</td>
<td>indexed.url</td>
<td>name of the file which contains the list of downloaded WWW pages</td>
</tr>
</tbody>
</table>
### 11.4 Crawler

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>considerOnlyEndTags</td>
<td>off</td>
<td>whether the program should only take closing tags into account during segmentation</td>
</tr>
<tr>
<td>defaultInputEncoding</td>
<td>iso-8859-2</td>
<td>encoding of input files</td>
</tr>
<tr>
<td>duplicateFilter</td>
<td>on</td>
<td>whether duplicate elements should be omitted during download</td>
</tr>
<tr>
<td>duplicateFilterLength</td>
<td>40</td>
<td>the amount of characters compared while looking for duplicates</td>
</tr>
<tr>
<td>ignoreTagFile</td>
<td>ignore.tag</td>
<td>file that contains the tags which are to be ignored when segmenting the input page</td>
</tr>
<tr>
<td>indexedElementsFile</td>
<td>indexed.eln</td>
<td>file which contains information about already downloaded elements, used when the duplicateFilter is enabled</td>
</tr>
<tr>
<td>languageFilter</td>
<td>off</td>
<td>whether the language filter is enabled (behavior depends on languageFilterStrict, the language is defined by requestedLanguage)</td>
</tr>
<tr>
<td>languageFilterStrict</td>
<td>1</td>
<td>if the language filter is enabled, the value of this variable defines whether the module will reject pages that contain markup pointing to a language different than the requested one (0) or all the ones that do not contain the requested language (1)</td>
</tr>
<tr>
<td>maxFilesPerCrawl</td>
<td>800</td>
<td>maximum number of pages which may be downloaded during a single crawl</td>
</tr>
<tr>
<td>minElementLength</td>
<td>1</td>
<td>minimal element length which qualifies it for download</td>
</tr>
<tr>
<td>overrideByDefault</td>
<td>no</td>
<td>whether the program should delete the existing content of an archive prior to downloading new pages</td>
</tr>
<tr>
<td>regexpAddressTester</td>
<td>yes</td>
<td>if the variable is enabled, the lines of addresses.acc are assumed to be regular expressions describing the accepted URLs; if not, they are assumed to be accepted URL prefixes</td>
</tr>
<tr>
<td>requestedLanguage</td>
<td>pl</td>
<td>if the language filter is enabled, the variable defines the requested language (by means of a two-character ISO code)</td>
</tr>
<tr>
<td>resumeInterrupted</td>
<td>yes</td>
<td>whether the program should resume interrupted activity</td>
</tr>
<tr>
<td>simplifyText</td>
<td>yes</td>
<td>whether the content of a page should be simplified by means of normalizing inverted commas, double quotes and spaces</td>
</tr>
<tr>
<td>urlQueueFile</td>
<td>queue.url</td>
<td>file which contains the list of URLs queued for downloading</td>
</tr>
<tr>
<td>urlIgnoredFile</td>
<td>ignored.url</td>
<td>file which is to store the ignored URLs (not qualified for download)</td>
</tr>
</tbody>
</table>
11.5 IndexBuilder

<table>
<thead>
<tr>
<th>minWordLength</th>
<th>2</th>
<th>minimal word length which makes it opaque (i.e. taken into account when building the index)</th>
</tr>
</thead>
<tbody>
<tr>
<td>segmentSeparators</td>
<td><code>.,@ () [] {} !?\/^&lt;</code></td>
<td>segment border markers (in order for a pair of words to be included in the index, they cannot be separated by any of them)</td>
</tr>
<tr>
<td>whiteWordsFile</td>
<td>ignore.wrd</td>
<td>file that contains words which are to be ignored during index building</td>
</tr>
<tr>
<td>wordSeparators</td>
<td><code>+-* #&amp; \$</code></td>
<td>word border markers (supplementing <code>segmentSeparators</code>); please bear in mind that white space is always considered to be a word border</td>
</tr>
</tbody>
</table>

11.6 SAManager / SAMain

<table>
<thead>
<tr>
<th>defaultRowsPerPage</th>
<th>20</th>
<th>default number of rows per page when displaying results in the stand-alone version</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultTestSelection</td>
<td>Freq, z22, SCP, FSCP</td>
<td>default collocation test selection, represented as a list of comma-separated abbreviations</td>
</tr>
<tr>
<td>editorCommand</td>
<td>emacs {0}</td>
<td>command issued for executing the external editor – see 8.4</td>
</tr>
<tr>
<td>forceWindowsLocation</td>
<td>no</td>
<td>whether new windows should be forcibly located in <code>locationX, locationY</code> position; useful e.g. in the case of FVWM, should the windows be located too high</td>
</tr>
<tr>
<td>locationX</td>
<td>40</td>
<td>the X coordinate of the upper left corner of new windows if <code>forceWindowsLocation</code> is enabled</td>
</tr>
<tr>
<td>locationY</td>
<td>40</td>
<td>the Y coordinate of the upper left corner of new windows if <code>forceWindowsLocation</code> is enabled</td>
</tr>
<tr>
<td>setEnvironment</td>
<td>yes</td>
<td>whether environment variables should be declared when executing the external editor</td>
</tr>
</tbody>
</table>

18
11.7 Network interface – general

<table>
<thead>
<tr>
<th>serverPort</th>
<th>7801</th>
<th>port on which the server listens for incoming queries</th>
</tr>
</thead>
</table>

11.8 PrettyPrinter

<table>
<thead>
<tr>
<th>params</th>
<th>values</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>colMinCount</td>
<td>3</td>
<td>minimal number of a collocation’s occurrences which qualifies it to be included in score summaries</td>
</tr>
<tr>
<td>excludeNumbers</td>
<td>yes</td>
<td>whether numbers should be omitted in word and collocation lists (collocations containing numbers)</td>
</tr>
<tr>
<td>excludeProperNames</td>
<td>yes</td>
<td>whether proper names should be omitted in collocation lists</td>
</tr>
<tr>
<td>maxRows</td>
<td>100</td>
<td>maximal number of rows in a collocation score summary which is to be saved</td>
</tr>
<tr>
<td>phpRef</td>
<td>../php/coll.php</td>
<td>location of coll.php relative to the archive directory</td>
</tr>
<tr>
<td>printAllWords</td>
<td>yes</td>
<td>whether word lists should be saved</td>
</tr>
<tr>
<td>printAllColIs</td>
<td>yes</td>
<td>whether collocation score summaries should be saved</td>
</tr>
<tr>
<td>printAllNames</td>
<td>yes</td>
<td>whether a list of proper names should be saved</td>
</tr>
<tr>
<td>printAllPages</td>
<td>yes</td>
<td>whether a list of downloaded pages should be saved</td>
</tr>
<tr>
<td>printerPrompt</td>
<td>yes</td>
<td>whether to offer the user additional collocation score summaries for individual words or word groups</td>
</tr>
<tr>
<td>rowsPerPage</td>
<td>50</td>
<td>number of rows per page</td>
</tr>
<tr>
<td>templatesEncoding</td>
<td>UTF-*</td>
<td>encoding of page templates</td>
</tr>
<tr>
<td>wordsMinCount</td>
<td>100</td>
<td>minimal number of occurrences which qualifies a word for being included in word lists</td>
</tr>
</tbody>
</table>
### 11.9 QueryServer

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fieldSeparator</td>
<td>&lt;&gt;</td>
<td>record field separator in the answer</td>
</tr>
<tr>
<td>maxAsteriskAmbiguity</td>
<td>20</td>
<td>maximal asterisk ambiguity</td>
</tr>
<tr>
<td>recordSeparator</td>
<td>&lt;&lt;&lt;</td>
<td>record separator in the answer</td>
</tr>
<tr>
<td>serverLogFile</td>
<td></td>
<td>log file (stderr, if it’s not specified)</td>
</tr>
<tr>
<td>serverTestSelection</td>
<td>Freq, DF, LLR, Mxi, z22, Dice, SCP, FSCP, RIDF</td>
<td>collocation tests utilized by the server and the order in which they are displayed (a comma-separated list)</td>
</tr>
</tbody>
</table>