Spejd – readme

26 czerwca 2012

Bartosz Zaborowski

Institute of Computer Science Polish Academy of Sciences

About

Spejd is a shallow parser, which allows for simultaneous syntactic parsing and morphological disambiguation, developed at the Institute of Computer Science, Polish Academy of Sciences, Warsaw.

Spejd homepage:

- official: http://zil.ipipan.waw.pl/Spejd/
- sourceforge (with main bugtracker): http://sourceforge.net/projects/spejd/

Author: Bartosz Zaborowski [bartosz.zaborowski@ipipan.waw.pl]

The current implementation is based on ideas and work of:

- Aleksander Buczyński
- Aleksander Wawer
- Adam Przepiórkowski
- Bartosz Zaborowski
- Aleksander Zabłocki

License: GPL v.3

1 Requirements

Spejd's requirements depend on version:

- **Windows version**: Windows XP SP3 or newer, 25MB of hard drive space
- **Linux static binaries**: GNU/Linux 2.6.9 or newer (64bit for 64bit version), 15MB of hard drive space
- **source code**: POSIX operating system (or POSIX libraries for Windows/MinGW), either GNU make or some other building environment (in that case you have to set it up manually), C and C++ compiler, dependencies meet (see bellow)

1.1 Dependencies

Libraries required for building Spejd (binary versions come with their dependencies). In parentheses versions that were tested.

- POSIX system or Windows with libraries implementing POSIX (at least posix-threads and iconv needed)
- zlib compression library (1.2.5)
- ICU unicode library (4.6.1, 4.8)
- boost library (at least regex and iostreams) $\geq$ 1.40 (1.46) (1.42.0 probably has a bug in gzip compressor, don't use it)
- rt library (for realtime clock)
- libxmlpp library (2.32.0)
- morfeusz morphological analyzer library (0.7/20110416),
  (you can get it from http://sgjp.pl/morfeusz/dopobrania.html)

Optional dependencies:

- google perftools (tcmalloc library) (1.6)
- pantera tagger (you can get it from http://code.google.com/p/pantera-tagger/)

2 Standards for representation of linguistic information

2.1 Tagset

Tagset used in Spejd can be defined for each grammar separately. It is set in the configuration file and defined in a separate file. See the 'doc/manual.pdf' for details on syntax of this file and 'examples/sample-morfeusz.cfg' for a sample tagset file.

2.2 Input and output formats

Spejd currently can read plain text and XCES and TEI P5 corpus xml formats (as in IPIPAN Corpus and National Corpus of Polish respectively). There is a configuration option for setting input format, however for most cases they can be automatically recognized.

Spejd can write output in two formats: XCES and TEI. The output format is set in the configuration file.

3 Installation

3.1 From binaries

Windows binary:
The Windows binary version comes with installer. You have to execute it and follow the instructions.

Linux binary:
Just unpack the package somewhere, it is ready for use.

3.2 From source code

Installation from source on *NIX systems is similar to many other packages. In the simplest case the shell commands `./configure; make; make install` should configure, build, and install Spejd (the last command may require superuser privileges). For standard installation options see INSTALL file.

The Spejd's configure script accepts the following nonstandard options:

- `--enable-static-spejd`
  This option causes the Spejd executable to be linked statically. It probably will not be portable if the Spejd is compiled with pantera support (since it installs some other data, not only the library). Disabled by default.
- `--with[out]-pantera`
  With this option you can disable or enable the Pantera tagger support. By default configure will check the system for existence of this package.
- `--with[out]-tcmalloc`
  This option allows to disable custom allocation methods from google perftools. It is optional, but recommended. It can speedup the Spejd by approx. 10% and it gives more reliable memory management/limits. The default is to build with tcmalloc.
- `--with-libicu-prefix=DIR`
  Causes to search for the ICU library in DIR. Precisely: icu-config will be searched in `$PATH` and also DIR/bin. It will be executed with –prefix=DIR parameter. By default icu-config is searched for in the `$PATH` environment and it will look for library in its default localizations. This option is useful probably only if you have installed ICU in nonstandard localization, like `$HOME`.

Compilation from source on Windows: Under Unixes the configure script generates config.h file. Under Windows the building environment must be manually set and special config.h file must be used - it is in the package under the name config.h.win. Just remember to rename it to config.h.

When compiling using MinGW remember to add the '-mthreads' option to get thread-safe versions of stl.

4 Basic usage

Spejd is a command line tool. It doesn't have any graphical interface.

The basic invocation command is following:

```
spejd [-c <config file name>] <paths to input files or dirs>
```

The `-c <config file name>` argument is optional, if not specified Spejd will look for a config.ini file in the current directory.

The config.ini file from examples directory in Spejd package can be used as basis for specific configuration. For parsing with attached example grammar it can be used without modifications.

If the path to input files is a directory, Spejd will search for input files recursively in this directory (it can filter searched file names, see example configuration). Output files will be placed along the input files according to naming rules set in configuration file.

The command to parse the example file using example grammar supplied with Spejd is:

```
spejd corpus
```

assuming the working directory is 'examples' from the Spejd package. With Linux binary packages it may be necessary to invoke Spejd with its path, e.g.:

```
/home/username/spejdpackage/spejd corpus
```

To parse few files in one invocation list them as parameters, e.g.:

```
spejd file1.txt file2.txt an_xces_corpus/morph.xml a_tei_corpus_directory
```

Note for Windows 7 users: Windows 7 sets the Spejd's installation directory to be read only. Hence, to experiment on 'examples' directory you have to copy it somewhere where you are able to write, e.g. to the 'my documents' directory.

5 Additional documentation

The main documentation is included in the package in 'doc/manual.pdf' file. Additional documentation can be found in example files (examples directory) in comments. They describe configuration, tagset definition, dictionaries and rules syntax.

6 Examples

See the 'examples' directory in the package (or in installation directory under Windows) for an example of config.ini (self explained in comments) and other files including example input.