

Contents

Faster builds

Fast machine, plenty of memory

Non-optimised build

Parallel make

ccache

distcc

Improve the build system

Making a release

Template arguments naming convention

Macros to handle with care

Using `gdb` or `valgrind` on TAF-Kit

Faster builds

Vaucanson takes a long time to build, but the time can be reduced dramatically with a few simple measures.

Fast machine, plenty of memory

Use a fast machine with plenty of memory. 2GB seems to be a minimum for an optimised build; any less causes severe swapping.

Non-optimised build

Use:

```
./configure ... CCFLAGS="-g -ggdb -Wall" CXXFLAGS="-g -ggdb -Wall"
```

N.B. The variables go at the end of the line.

Parallel make

Use:

```
export MAKEFLAGS="-j $(grep ^processor /proc/cpuinfo | wc -l)"
```

This allows make to run up to one process per CPU core.

FIXME: It would be polite to use `-l` to stop make from starting new processes if the machine load goes above a certain level, e.g. `2.0 * no_of_CPUs`.

ccache

Use `ccache` with a cache size of at least 4Gb (at the time of writing on a 32-bit Debian system `vaucanson` uses about 3.5Gb):

```
ccache -M 4G
```

distcc

The following instructions are for LRDE users, but may be adapted to other places.

1. Wake up as much machines as you want with `lrde-wakeonlan`. Use `lrde-wakeonlan .` to wake up all hosts.
2. Configure with:

```
./configure CC=gcc-4.2 CXX=g++-4.2
```

You need to specify the GCC version number to make sure all machines use the same compiler.

3. Update the `.distcc/hosts` files with the list of build hosts available. It should look something like:

```
berville-en-caux.lrde.epita.fr/2,lzo
marvejols.lrde.epita.fr/2,lzo
whiteagonycreek.lrde.epita.fr/2,lzo
--randomize
```

The script `~adl/usr/bin/update-distcc-hosts` can create this file automatically for you.

4. Run `make -jN CC='distcc gcc-4.2' CXX='distcc g++-4.2'` where `N` is the number of available hosts. Beware that preprocessing and linking are still done locally, so you may not want to use more than `-j8` on a single core CPU.

Improve the build system

FIXME: Remove this section once the build system can't easily be improved further!

The build system has recently (at the time of writing) been improved to build the libraries much faster, but there's further room for speed-ups. For example: make the tests use `-DVCSN_USE_LIB`.

Making a release

Don't do these steps from memory.

- Make sure the last run of the autobuilder was successful.
- Check `trac` to make sure there are no important pending tickets.
- Update `doc/share/` with `cd doc; make share-up`.
- Make sure `doc/NEWS.txt` is up-to-date. (Mention important known bugs!)
- Make sure `doc/README.txt` is up-to-date.
- Make sure `doc/FAQ.txt` is up-to-date.
- Make sure `AUTHORS` is up-to-date.

Make sure your system has up-to-date tools (Autotools, Swig, Doxygen, ...) before continuing.

- Bump the version number in `configure.ac`.
- Run `bootstrap`.
- Write the `ChangeLog` entry for all the above changes (But don't commit it before `distcheck`.)
- Run `make distcheck`.

- Commit all changes on success. Commit suicide otherwise.
- Tag the repository for the release.
- Append a `a` to the version number in `configure.ac` and commit this new change so that the next run of the autobuilder won't create a release.
- Copy the files created by `distcheck` to `/lrde/dload/vaucanson/` don't forget to `chmod a+rX` all files and directories, and to update the `latest` link.
- Create the release page on the LRDE wiki.
- Update the Vaucanson page to point to it.
- Update the Vaucanson download page to point to the release.
- Send an announcement to vaucanson@lrde.epita.fr. The text of the announcement should explain what Vaucanson is (so we can forward the mail to another mailing list) and should include the list of major improvements since the last version (i.e., the top of `NEWS`). Do not assume that people will follow links to get details.
- If the release is a beta release, or an intermediate release before a major release, make it clear in the announcement and on the wiki.
- Install any new major release on `vcsn.enst.fr`.
- Complete and detail this list with what was missing (whatever will help the next guy doing the release).

Template arguments naming convention

Template arguments:

- `A` : Automaton structure.
- `AI` : Automaton implementation.
- `S` : Series.
- `SI` : Series implementation.
- `W` : a Word.

In the case where multiple possibilities could be used, suffix the template argument with the appropriate numbering. For example, to enable the use of two different automaton implementations for each argument of an algorithm:

```
template <typename A, typename AI1, typename AI2>
Element<A, AI1>
algorithm(const Element<A, AI1>& a1, const Element<A, AI2>& a2);
```

Macros to handle with care

The `VCSN_GRAPH_IMPL` macro must only appear in three locations:

- `include/vaucanson/context`
- `include/vaucanson/automata/generic_contexts`
- `include/vaucanson/misc/usual_macros.hh` (be careful when defining new macros using it)

Any other use is irrelevant and may be very harmful. Moreover this macro must never be used in a file with guards.

Using gdb or valgrind on TAF-Kit

The executables that are built in `taf-kit/src/` are `libtool` scripts that call the true executables (usually hidden under `taf-kit/src/.libs/`). You cannot run `gdb` directly on these scripts, you should always ask `libtool` to do it for you:

```
% cd taf-kit/src
% export VCSN_DATA_PATH=$PWD/../../data
% libtool --mode=execute gdb ./vcsn-int-b
(gdb) run determinize x.xml
...
```

It's often more convenient to run the scripts from `taf-kit/tests` because they export `VCSN_DATA_PATH` and run the corresponding executable from `taf-kit/src` for you. In that case you have to use the `PREVCSN` environment variable to specify these `libtool` options:

```
% cd taf-kit/tests
% PREVCSN='libtool --mode=execute gdb' ./vcsn-int-b
(gdb) run determinize x.xml
...
```

The same commands can of course be used to run other tools like Valgrind. Here is how to run TAF-Kit under Valgrind and attach a debugger on the first error:

```
% cd taf-kit/tests
% PREVCSN='libtool --mode=execute valgrind --db-attach' ./vcsn-int-b
```

A note for Darwin users: because your system comes with another tool called `libtool`, GNU `libtool` is usually installed as `glibtool`. Alternatively, you may want to use the copy of `libtool` output by `configure` at the root of Vaucanson's build tree.