GROMACS - Task #656

Implement selection input from a file in the new framework

01/09/2011 08:25 PM - Teemu Murtola

Status:	Closed
Priority:	Normal
Assignee:	Teemu Murtola
Category:	selections
Target version:	5.0
Difficulty:	uncategorized

Description

The new mechanism for providing selections is a lot more flexible than the old one, and makes it possible to have several different command-line options for providing selections for analysis. Interactive input is also possible, with the user being prompted for all selections that are required and were not provided on the command line. However, file input is more complex; if there are more than one option that can take an unspecified number of selections, reading them from a single file is tricky. So, there are two options:

1. Provide a mechanism for providing a separate file for each option, and parse the selections from there.
2. Provide a mechanism for specifying the option to which a selection belongs in a file.

The first one is probably both more intuitive and easier to implement: changes should be limited to source:src/gromacs/selection/selectionoptionstorage.cpp, and the parser does not need to be changed at all.

Related issues:

Blocked by GROMACS - Task #985: Context information for exceptions	**Closed**	**08/02/2012**

Associated revisions

Revision 762095c2 - 04/18/2012 06:16 PM - Teemu Murtola

Split FileNameOption into a separate file.

IssueID #642 (also helps for some alternatives in IssueID #656)

Change-Id: I61b236fcd59963c8ba5f7a49aba2b5197552d316

Revision 672c3b1b - 05/03/2012 06:32 AM - Teemu Murtola

Initial implementation of selection file input.

- Add SelectionFileOption and related classes to implement an option
that can be used to provide selections from a file.
- Add SelectionCollection::parseRequestedFromFile() to do the actual
work.
- Add special handling for the option in command-line help to print it
together with other selection options.
- Add tests.
- Removed support for specifying a selection option multiple times on
the command line, since it is not clear how it should work together
with the new option.
- Temporary exception safety fix for selection file input.
- Updated valgrind suppression rules for MacOS for some
exception-handling stuff.

Initial implementation for IssueID #656.

Change-Id: Id4ddf545ec13986fa57dac42a8b1dc4075a42840

Revision 896cf4e4 - 05/18/2012 11:46 AM - Teemu Murtola

Split option code away from SelectionCollection.

Move code required for selection option implementation from
SelectionCollection to a separate SelectionOptionManager class.
Simplified the selection request handling code in the process, but
otherwise this commit mainly moves code around without changing any
functionality (except for small changes required to use the new class
in code using the selections).

This makes the responsibilities in the code clearer, since SelectionCollection is now completely independent of any options implementation. It also simplifies the involved classes, since the options handling (which will become even more involved with #656) no longer complicates the core of the selection module.

Related to #656.

Change-Id: Iac24a2f9392c8cc2421edef3f212ddb462e864b3

Revision 3db13af1 - 05/18/2012 08:59 PM - Teemu Murtola

Simpler required selection input from a file.

Now a selection file option (-sf) assigns the contents of the file to required selection option(s) that haven't yet been assigned if it has no other context. Main benefit is that a plain '-sf file.dat' (without any other selection options specified) assigns the contents of file.dat to required selections, similarly to how it worked in 4.5.

Resolves the main open issue for #656, error message and help improvements remain.

Change-Id: I6311b548ab21ecf0f489f2743e2750411f86b3ff

Revision 45d74f45 - 06/13/2012 11:52 PM - Teemu Murtola

Updates to selection online help.

Some of the help topics were outdated after changes introduced by the linked Redmine issues (some of them even before that). Improved the situation a bit and also did some misc. other documentation fixes.

Related to #656 and #666.

Change-Id: I7f18ca4ad8ce2aa5cd808281bab044253ca77f56

Revision 8d575f9e - 09/07/2012 11:34 AM - Teemu Murtola

Better error messages for SelectionFileOption.

Use the added exception context information to provide the file name in the error message when there is a problem with a selection provided with -sf. Also improve the error messages for cases where the number of selections in the file does not match what is expected, or where the assignment of selections to options is ambiguous.

Closes #656.

Change-Id: I6158960a970e6a22fb28f3872538f1de16883887

History

#1 - 04/13/2012 05:41 PM - Teemu Murtola

- Target version set to 5.0

#2 - 04/17/2012 06:32 PM - Teemu Murtola

Some alternative implementation possibilities include:

1. Have a common prefix that is used to designate a file: e.g. -sel file:abc.dat would read selections from abc.dat for the -sel option. Easy to implement, but perhaps not the most intuitive.
2. Have -sel abc.dat read selections from abc.dat. This would be very intuitive, but the issue is that how can be figure out whether the string is a file name or a selection? The simplest check is that if the string specifies a file name that exists, then this file is read, otherwise the string is parsed as a selection. Can lead to unobvious error messages if there is a typo in the file name.
3. Alter the selection parser to treat a lone string as a file name if it doesn't match anything else, or some other variant of modifying the selection syntax such that file input is possible.
4. Have a syntax something like this: -sel -sf abc.dat. Also allowed would be -sel -sf abc.dat -sel2 -sf bcd.dat, as well as -sel -sel2 -sf abc.dat if it was unambiguous.

The last one seems most attractive. Opinions are welcome.

#3 - 04/19/2012 08:44 PM - Teemu Murtola

Thought a bit more of the issue, and the fourth option above is probably the best, even if it requires some implementation work. There are some

questions of how to handle corner cases with required selections, though: if a program accepts selection options -sel, -sel2, -optsel and -optsel2, with first two required, then what should happen with the following arguments, in particular with the required selections?

- -sf abc.dat
- -sel -sf abc.dat
- -sf abc.dat -sel
- -optsel -sf abc.dat
- -optsel -sf abc.dat -sf bcd.dat
- -optsel -sf abc.dat -optsel2 -sf bcd.dat

For reference, the current implementation (without the file option) works as follows:

- If an option is provided on the command line, the values given for it are parsed as the selection.
- If an option is provided on the command line, but no value is given, then that selection is always prompted interactively.
- If a required option is not provided on the command line, it is also prompted interactively.

Some behaviors are probably easier to implement than others, but I would welcome comments on what would feel most natural, and it might be best to not know too much of the implementation details to comment on that. :)

#4 - 04/28/2012 03:55 PM - Teemu Murtola

- Status changed from New to In Progress

- Assignee set to Teemu Murtola

Will try to implement at least a simple version of the separate command-line option for file input to help the discussion.

#5 - 04/30/2012 09:13 AM - Teemu Murtola

https://gerrit.gromacs.org/#/c/879/ now implements the basics of the fourth option. Of the alternatives in comment 3, the second now assigns the contents of abc.dat into -sel, the fourth into -optsel, and the last puts abc.dat into -optsel and bcd.dat into -optsel2. All others give errors. Additionally,

- -sel -optsel -sf abc.dat

puts the first selections in abc.dat into -sel and the rest in -optsel (requires that -sel takes a predetermined number of selections).

Error messages still need to be improved, but some questions from #666 would be nice to address first (so that the error message could refer to relevant help text). It could also be nice to be able to provide values for all required selections with simply -sf abc.dat without mentioning any other selection options on the command line (this is how it works in 4.5, but there is always just a single -select option in that version, so it's simpler there).

#6 - 05/19/2012 05:49 AM - Teemu Murtola

https://gerrit.gromacs.org/#/c/977/ adds support for -sf abc.dat setting the required options if no other context is present. So now the first alternative in comment 3 puts the contents of abc.dat into -sel and -sel2. The fifth puts abc.dat into -optsel and bcd.dat into -sel and -sel2. The third still gives an error. Additionally,

- -sel "resname RA RB" -sf abc.dat

puts the contents of abc.dat into -sel2 (since -sel is already set).

#7 - 05/21/2012 10:42 PM - Roland Schulz

This looks very nice and user-friendly!

One minor issue. Why did you choose to make selections from a file not automatically end on a newline as is the behavior for interactive selections? Thus
atomnr 1
atomnr 2
is 2 selections interactive but one invalid selection in a file (fixed by adding a semicolon).

#8 - 05/22/2012 06:05 AM - Teemu Murtola

Roland Schulz wrote:

> One minor issue. Why did you choose to make selections from a file not automatically end on a newline as is the behavior for interactive selections? Thus
> atomnr 1
> atomnr 2
> is 2 selections interactive but one invalid selection in a file (fixed by adding a semicolon).

I originally wrote the parser to only accept semicolons as separators, to avoid the need to specify line continuation explicitly. Semicolon after the last selection has always been optional, I think, to make command-line input work intuitively. I then realized that for interactive input, this is not very intuitive, so I made that accept newlines in addition to semicolons, but require backslash for line continuation. My reasoning for keeping the file input as it was originally was that complex selections are typically given from a file, and multiline input felt easier with the old behavior (no explicit line

continuation).

In 4.5, if selections are given from a pipe/redirected stdin, it uses the same logic as for file input, but I haven't yet ported that to the C++ code (it uses isatty(), and haven't yet considered whether that should be wrapped and if, then how).

#9 - 09/11/2012 06:04 AM - Teemu Murtola

- Status changed from In Progress to Closed

#10 - 02/14/2014 08:10 PM - Teemu Murtola

- Project changed from Next-generation analysis tools to GROMACS

- Category set to selections