Debugging Overview

Justin M. Turney

May 18, 2017

Debuggers

These debuggers are useful for debugging C++:

- GDB
  GNU Debugger
  Generally found on Linux/Unix boxes
- LLDB
  LLVM Debugger
  Default on modern Mac OS boxes

This is useful for debugging memory issues in your C++ code:

- Valgrind
  For Macs support generally lags behind a full release. Current support is for Mac OS <= 10.11.

Debugging Psi4

- Starting the debugger with Psi4:
  - **GDB**

    ```
    % gdb --args python /path/to/psi4/bin/psi4
    ```

  - **LLDB**

    ```
    % lldb -- python /path/to/psi4/bin/psi4
    ```

- This only sets up the debugger; it doesn't start executing Psi4.

Debugging Psi4

- Stopping the code when an exception is thrown:
  - **GDB**

    ```
    (gdb) catch throw
    ```
  - **LLDB**

    ```
    (lldb) break set -E C++
    ```

Debugging Psi4

- Setting breakpoints in the code:
  - **GDB**

```
(gdb) break main
(gdb) break test.c:12
```

  - **LLDB**

```
(lldb) breakpoint set --name main
(lldb) br s -n main
(lldb) b main

(lldb) breakpoint set --file test.c --line 12
(lldb) br s -f test.c -l 12
(lldb) b test.c:12
```

Debugging Psi4

- Running Psi4 in the debugger:
  - **GDB**

    ```
    (gdb) run
    (gdb) r
    ```

  - **LLDB**

    ```
    (lldb) run
    (lldb) r
    ```

Debugging Psi4

- Attach to existing process with ID 123:
  - **GDB**

    ```
    (gdb) attach 123
    ```

  - **LLDB**

    ```
    (lldb) process attach --pid 123
    (lldb) attach -p 123
    ```

Debugging Psi4

- Show a backtrace:
  - **GDB**

    ```
    (gdb) bt
    ```
  - **LLDB**

    ```
    (lldb) thread backtrace
    (lldb) bt
    ```

Debugging Psi4

- Stepping through the code:
  - **GDB**

```
(gdb) step      % step into function calls
(gdb) s
(gdb) next      % step over function calls
(gdb) n
```

  - **LLDB**

```
(lldb) step     % step into function calls
(lldb) s
(lldb) next     % step over function calls
(lldb) n
```

Debugging Psi4

- Stepping through the code:
  - **GDB**

```
(gdb) step      % step into function calls
(gdb) s
(gdb) next      % step over function calls
(gdb) n
```

  - **LLDB**

```
(lldb) step     % step into function calls
(lldb) s
(lldb) next     % step over function calls
(lldb) n
```

Debugging Psi4

- Continue execution:
  - **GDB**

    ```
    (gdb) continue
    (gdb) c
    ```

  - **LLDB**

    ```
    (lldb) continue
    (lldb) c
    ```

Debugging Psi4

- Printing variables in the current frame (local):
  - **GDB**

    ```
    (gdb) p bar
    ```

  - **LLDB**

    ```
    (lldb) frame variable bar
    (lldb) fr v bar
    (lldb) p bar
    ```

Debugging Psi4

- Show the variables in the current frame (local):
  - **GDB**

    ```
    (gdb) info locals
    ```
  - **LLDB**

    ```
    (lldb) frame variable --no-args
    (lldb) fr v -a
    ```

External Resource

- An excellent resource for additional commands can be found here: https://lldb.llvm.org/lldb-gdb.html