Other Valgrind Tools

Memcheck is not the only tool in Valgrind—although it is one of the most commonly used ones, and is what many people think of when they hear "Valgrind." As you become a more advanced programmer, you may find it useful to put some of the other tools to use. We will not delve into them too deeply, but we will note a couple that exist, so that you can know to explore them further as you need to.

There are a variety of issues that arise in multi-threaded programming, such as deadlocks due to improper ordering of acquiring locks, data races, or improper use of synchronization primitives. You should also know that these can be difficult to find due to the non-deterministic behavior of multi-threaded executions—a bug may manifest one time, but then not show up in the next hundred times you run the program. The Valgrind tool **Helgrind**is designed to check for a variety of errors related to multi-threaded programming. See http://valgrind.org/docs/manual/hg-manual.html for more details on Helgrind.

Valgrind's tools are not limited to helping your find correctness bugs in your program, but also can be helpful in understanding performance memory usage issues. For example, the **Callgrind** tool gives information about the performance characteristics of a program based on Valgrind's simulation of hardware resources as it executes the program. Another tool is**Massif**, which profiles the dynamic memory allocations in the heap, and gives information about how much memory is allocated at any given time and where in the code the memory was allocated.

See http://valgrind.org/info/tools.html for an overview of the tools. Also note that many of these other tools have header files that you can include with macros that allow your program to interact directly with Valgrind (similarly to the functionality in **memcheck.h**).