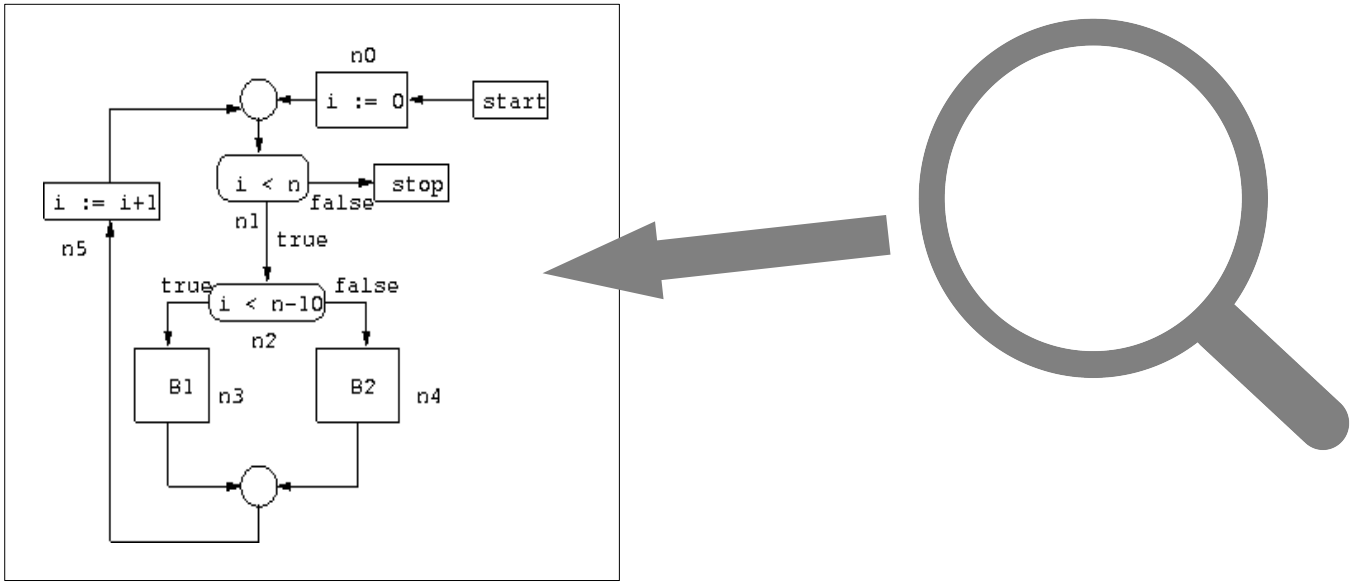
Dynamic Program Analysis
Lecture 10: Valgrind



Valgrind

A framework for building dynamic analysis tools for C/C++ programs

Valgrind will run the binary on a CPU emulator, with the appropriate instrumentation for the selected analysis

The run produces a list of messages from the execution: typically, a report of suspected errors in the program

Free software, anyone can download and use without restrictions

Website: `www.valgrind.org`

Valgrind Execution Environment

Runs on unix systems (like linux), **not** under Windows

Is run from the command line:

```
> valgrind [valgrind-options] your-prog [your-prog-options]
```

Example: running memcheck on program `prog`

```
> valgrind --tool=memcheck prog
```

Memcheck is the default tool, so the same is accomplished by:

```
> valgrind prog
```

Valgrind Output - the Commentary

A stream of text, detailing error reports and other significant events

Typically written to `stderr`, but can be redirected to files or sockets

Example of output in the commentary (from memcheck):

```
==25832== Invalid read of size 4
==25832==    at 0x8048724: BandMatrix::ReSize(int, int, int) (bogon.cpp:45)
==25832==    by 0x80487AF: main (bogon.cpp:66)
==25832== Address 0xBFFFF74C is not stack'd, malloc'd or free'd
```

First number is process ID, can mostly be ignored

Then, memcheck reports on a possible memory-related bug

Valgrind Workflow

1. Compile and link your program
2. For each test vector in your test vector set:
   - Run your program with Valgrind, using the appropriate Valgrind tool
3. Analyse the collected outputs
4. Fix the reported errors, if any
5. If errors were fixed, go to 1