

Performance Improvement

The material for this lecture is drawn, in part, from The Practice of Programming (Kernighan & Pike) Chapter 7

Goals of this Lecture



- · Help you learn about:
 - Techniques for improving program performance
 - How to make your programs run faster and/or use less memory
 - The GPROF execution profiler
- · Why?
 - In a large program, typically a small fragment of the code consumes most of the CPU time and/or memory
 - A power programmer knows how to identify such code fragments
 - A power programmer knows techniques for improving the performance of such code fragments

2

Performance Improvement Pros

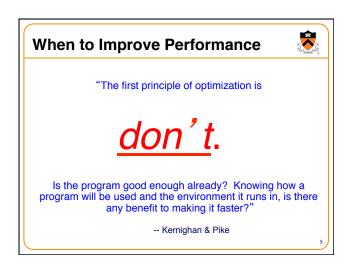


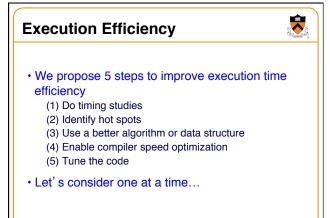
- Techniques described in this lecture can yield answers to questions such as:
 - How slow is my program?
 - Where is my program slow?
 - Why is my program slow?
 - How can I make my program run faster?
 - How can I make my program use less memory?

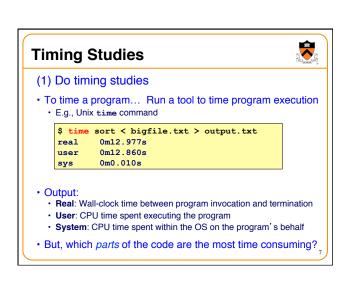
Performance Improvement Cons

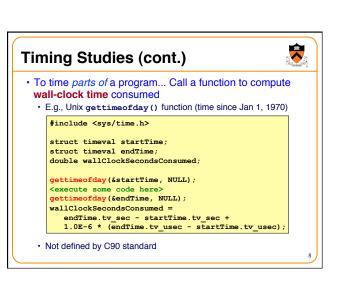


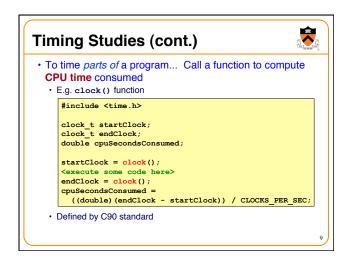
- Techniques described in this lecture can yield code that:
 - · Is less clear/maintainable
 - · Might confuse debuggers
 - · Might contain bugs
 - Requires regression testing
- So...



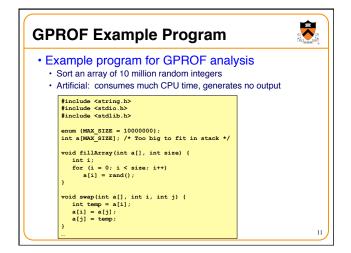


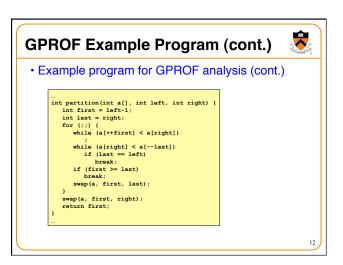


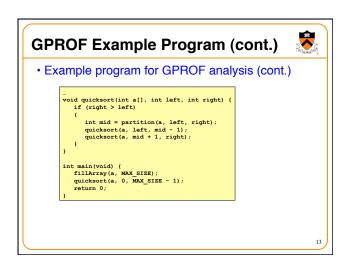


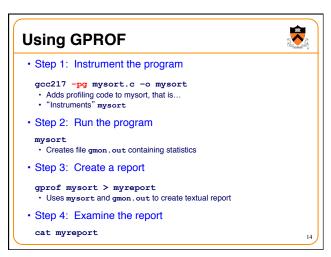


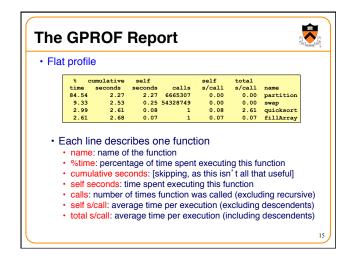


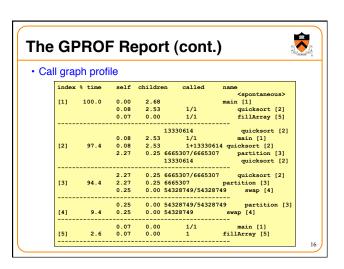












The GPROF Report (cont.)



- · Call graph profile (cont.)
 - Each section describes one function
 - Which functions called it, and how much time was consumed?
 - · Which functions it calls, how many times, and for how long?
 - · Usually overkill; we won't look at this output in any detail

17

GPROF Report Analysis



- Observations
 - swap () is called very many times; each call consumes little time; swap () consumes only 9% of the time overall
 - partition() is called many times; each call consumes little time; but partition() consumes 85% of the time overall
- Conclusions
- To improve performance, try to make partition () faster
- Don't even think about trying to make fillArray() or quicksort() faster

18

GPROF Design



- Incidentally...
- How does GPROF work?
 - Good question
 - Essentially, by randomly sampling the code as it runs
 - ... and seeing what line is running, & what function it's in

Execution Efficiency Summary



- Steps to improve execution (time) efficiency:
- (1) Do timing studies
 - (2) Identify hot spots
 - (3) Use a better algorithm or data structure
 - (4) Enable compiler speed optimization
 - (5) Tune the code

20

