Homework 10
Profiling

Assigned: Friday, March 18

Due: Friday, March 25, 11:00AM (Hard Deadline)

Submission Instructions

Submit this assignment on Gradescope. You must submit every page of this PDF. We recommend using the free online tool PDFescape to edit and fill out this PDF. You may also print, handwrite, and scan this assignment.

There may multiple answers for each question. If you are unsure, state your assumptions and your reasoning for why you think your answer makes sense.

You must select the right page for each question on Gradescope.
If you do not select the right pages, we will not grade the question.

Optional Reading

Developer Survey 2016, from Stack Overflow.

`https://stackoverflow.com/research/developer-survey-2016`

Some students have asked for some insights into CS in industry. Every year, Stack Overflow does a large survey of developers. Their intro does a nice job of covering its caveats, but it remains an interesting look into what it means to have a career in CS and what CS in the real world looks like.

CAVEATS

`gprof` is **not** a debugging tool. Your code **must** work correctly before trying to run gprof. Your code **must** exit normally (no segfault, exception, etc) or gprof will not do anything!

`gprof` **does not work** on macs. On mac, the best tool to use is Instruments, however, for this homework please use your VM and gprof.

1 Reasoning about performance

Clone a copy of https://gitlab.eecs.umich.edu/ppannuto/c4cs-w16-profiling.

Look through `main.c` to understand what this program is doing (not much, a "nop" is "no-operation", that is, sit idle for a cycle). The call graph of this program is about:

```
\-- main
    \-- parent
            \-- child1
            \-- child2
    \-- no_children
```

The `time` utility reports how long a program takes to run.

Run `make baseline`.

How long did it take for the baseline to run?
Given how long baseline took, how long do you expect each of the following to take from the start of each function until it finishes?

There's a lot of variability for these answers depending on the computer. Baseline 9-10 sec seconds

child1 1.5-2.6 seconds

child2 26-36 nanoseconds ← note the units here

parent 2-3 seconds ← think carefully about this one! What's weird about the for loops?

no_children .3 seconds

2 `gprof` overhead

Now run `make overhead`.

What is the difference between `main_no_profiling` and `main_with_profiling`? `main_no_profiling` runs the code only tracking overrall time, and `main_with_profiling` runs both the code and profiler.

Which ran faster, `main_no_profiling` or `main_with_profiling`? Why? `main_no_profiling`, because profiling takes time.

Between child1 or child2, which should you remove such that `main_no_profiling` and `main_with_profiling` will have nearly the same runtime? Why remove that child? child2, because the for loop takes a long time, which causes the profiler to take an even longer time.

3 Understanding `gprof`

Finally, run `make profile.txt`. Take a look at the `profile.txt` file.

The gprof output has many different times it reports, such as "cumulative seconds" and "self seconds". Which timing report matches what Question 1 asked for? total (s/call)

How well did your estimates from Question 1 line up with the time reported by gprof?
What explains the difference between your estimate and the time reported by gprof? Lots of accepted answers. Example: Question 1 doesn't take into account transfer times between functions

What generated the file `gmon.out`? Compiling with the `-pg` flag.

Document Revision History

March 21, 5PM: Revise the first question in section 3.

~~Did Question 1 ask for "cumulative seconds" or "self seconds"? What is the difference?~~ → The gprof output has many different times it reports, such as "cumulative seconds" and "self seconds". Which timing report matches what Question 1 asked for?