
Algorithms

Part 2: Measuring Time

Notes for CSC 100 - The Beauty and Joy of Computing
The University of North Carolina at Greensboro

Reminders

For Friday:

Prepare for Lab 5 - do Pre-Lab Reading!

For Next Week:

Blown to Bits - Chapter 3 (RR by next Tuesday)

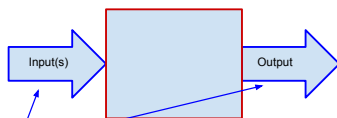
Homework:

Work on Homework 2!

Last Time We Saw...

Problems are defined by input/output relation, with no reference to how they are solved (*focus is on what*)

Algorithms are well-defined computational procedures that solve problems (*focus is on how*)



Problem specifier worries about input and outputs

Implementer / algorithm designer worries about the computational process

In BYOB

Problem Focus

GCD of 15 and 6

With a well-chosen name, that may define the problem well enough for the user!

Algorithm Focus

```
GCD of px and py
script variables aCounter
set aCounter to px
if py < px
  set aCounter to py
repeat until aCounter divides evenly into px and py
  change aCounter by 1
report aCounter
```

This is an over-simplification: Sometimes the user wants to know some properties of the block implementation.

Question: What kinds of properties?

Algorithm Characteristics

- Does the algorithm work correctly (does it solve the problem)?
- Is the answer provided precise?
- How confident are you in the correctness of the algorithm and implementation (simpler algorithms are easier to verify)?
- How much memory does the algorithm require?
- How fast is the algorithm?

Algorithm Characteristics

- Does the algorithm work correctly (does it solve the problem)?
 - Is the answer provided precise?
 - How confident are you in the correctness of the algorithm and implementation (simpler algorithms are easier to verify)?
 - How much memory does the algorithm require?
 - How fast is the algorithm?
- Assume no problems with correctness or precision for now.
- Memory is a problem for some algorithms, but not as common a limiting factor as...

Time is usually the most interesting and limiting characteristic, whether talking about running a big computation for a week, or calculating a new graphics frame in 1/30 of a second.

What is "time" for an Algorithm?

Time is time, right?



But...

- Does time depend on things other than the algorithm?
- If run many times (on the same input), is time always the same?
- If QuickSort runs in 20 seconds on my old IBM PC, and SelectionSort runs in 0.5 seconds on my current computer, is SelectionSort a faster algorithm?
- Can we give clock time without implementing the algorithm?

Correcting for vagueness of timing

Wall-clock times depend on:

- Speed of computer that it's run on
- What else is happening on the computer
- ... and a few other things we'll address later

But... these are not differences in algorithms!

Solution: Algorithms are sequences of steps, so count steps!

Question: What's a step?

BYOB blocks and "steps"

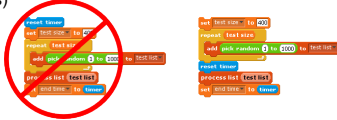
Which of these should not be treated as "one step"?

- a) `set variable to 15`
- b) `sum + value`
- c) `add 15 to list`
- d) `list contains 15`
- e) `set inf 10`

Experimenting with timing BYOB scripts

Timer is available to help test things out

- Reset timer to start it at zero
- Save current timer value into a variable for "lap timer"
- Watch variable shows limited precision - for more use "say"
- Tip: surround only what you're interested in timing with reset/set blocks (not initializations)



Summary

Time is one of the most important algorithm characteristics

An "algorithm" should be independent of what runs it
→ So measure time in steps, not seconds

But - when you want time in seconds for a specific implementation, BYOB gives you tools to measure that.
