ALGORITHMIC PROBLEM SOLVING

1. Algorithmic Problem Solving Fundamentals

There are three components to using Algorithms to solve a problem you face. You must choose/create an algorithm, ensure the algorithm produces the desired output, and analyze the algorithm for feasibility.

1.1. **Picking an Existing Algorithm.** Questions to answer when considering appropriate, existing algorithms;

- Which algorithm takes the least amount of time?
- Which one takes the least amount of memory?
- Which is the easiest to program?

Always choose an algorithm that best suits the circumstances.

1.2. **Creating Your Own Algorithm.** The process of creating and evaluating your own algorithm expressed algorithmically is:

(1) Understand the Problem.
(2) Decide on:
    - computational means,
    - exact vs. approximate solving,
    - data structure(s),
    - algorithm design technique
(3) Design an Algorithm
(4) Check Correctness
    - If not correct go to step 2 or 3
(5) Analyze the Algorithm
    - If not good enough go to step 2 or 3
(6) Code the Algorithm

1.3. **Algorithm Correctness.** Once an algorithm has been specified, you have to prove its correctness.

To accomplish this, you must prove that the algorithm yields the required result for *every* legitimate input in a finite amount of time. Tracing the algorithm's operation for a few specific inputs does not prove its correctness.

With some algorithms proving correctness is simple. For others it is complicated. As an example of a non-complicated proof take a look at the end of this document for two proofs of Euclid's gcd algorithm.

A common method for proving correctness is mathematical induction.

Date: September 27, 2018.

1.4. **Algorithm Analysis.** No perfect algorithm exists. To be able to choose between algorithms that achieve the same end you must balance these data points:

- Time efficiency — how fast each algorithm runs.
- Space efficiency — how much memory each algorithm needs.
- Simplicity
  - like Beauty, often in the eye of the beholder.
  - Which is simplest of the three *gcd* algorithms?
- Generality
  - of the problem each algorithm solves
  - of the range of inputs each algorithm accepts

2. Common Types of Problems Amenable to Algorithmic Problem Sovling

2.1. **Sorting.**

- With all the courses I need/should take, which one should I take next?

2.2. **Searching.**

- Where did I leave my wallet???

2.3. **String Processing.**

- Auto-correcting spelling and grammar along with word suggestion are done with string processing algorithms.

2.4. **Graph Problems.** This is the oldest area of application of algorithms since graphs can be used to model many kinds of everyday problems.

- What is an efficient way to get from the STC to the Snow building? {transportation and communications networks problems}
- In what order should I work on my homework to get done the soonest? {project and production scheduling}
- How do I get my character from point A to point B without dying? {game problems}
- What is the maximum number of links required to reach a web server from a mobile device by the most direct route between them.{maxima, minima network/routing problems}

2.4.1. *Easy Graph Problems.*

- Shortest Path
- Minimum Spanning Trees

2.4.2. *Hard Graph Problems.*

- Traveling Salesperson
- Map-Coloring

2.5. **Problems that ask to find a combinatorial object.** These are the most intransigent problems in computing. The size of the solution grows extremely fast when the initial data set size only grows moderately. Also, there are no known algorithms for most of these types of problems that can be to compute a solution in a reasonable amount of time (before the sun dies). Worst of all, most computer scientists believe they are unknown because they can not exist.

- permutation
- combination (subset)
- power set (set of all subsets)

2.6. **Geometric Problems.**

- Which major city is closest to Rexburg? {closest pair of points out of a set of points problem}
- How can I draw a 3-D map of Idaho? {polygon finding problem}

2.7. **Numerical Problems.** These problems involve mathematical objects of a *continuous* nature such as :

- If a train leaves Chicago at 3:00 PM heading east... {solving equations}
- Finding the optimal path of a product through a factory ignoring physical constraints. {solving systems of equations}
- If the economy grows at a rate expressed by what appears to be a non-describable sinusoidal function $f(t)$ for the next 10 years, how much bigger will it be then compared to today? {computing definite integrals}
- How efficient is my code? {evaluating functions}

3. How can the correctness of Euclid's GCD algorithm be proven?

GCD Definition. Given two positive integers m and n, find their greatest common divisor, i.e., the largest positive integer which evenly divides both m and n.

3.1. **The Naïve Way.** The central claim of Euclid's algorithm is $gcd(m, n) = gcd(n, m \pmod n)$.

Recall the definition of *division*: an integer a (not zero) *divides* an integer b (or equivalently, b is *divisible* by a, or a is a *divisor* of b. All signified by $a|b$) if there is an integer x such that $b = ax$.

Here's a Lemma (a *helper* to the main proof):

For any pair of positive integers m and n, if d divides both m and n then d also divides both n and r, where $r = m \pmod n$.

Proof of Lemma

Another way to write r is $m - qn$ where q is the *quotient* ($q = \lfloor m/n \rfloor$). Now ask:

1. Does d divide m? (Yes, it's assumed in the premise of the lemma.) 2. Does d divide qn? (Must prove that since d divides n (also assumed) that d divides a multiple of n.) 3. Does d divide $m - qn$? (If it divides both /m/ and /qn/, that is?)

To prove the Lemma you must prove the answer to 2 and 3 is yes.

Show that d divides qn given d divides n as follows:

(1) $n = dx$(Definition of division)

(2) $qn = q(dx)$(Multiply both sides by /q/)

(3) $qn = d(qx)$(Associative law of multiplication)

Show that if d divides u, and d divides v, then d also divides their difference $u - v$ as follows:

(4) $u = ds$(Definition of division)

(5) $v = dt$(Definition of division)

(6) $u - v = ds - dt$(Equals subtracted from equals are equal)

(7) $u - v = d(s - t)/$(Distributive law of multiplication)

(8)

Therefore d divides $u - v$ by definition of division (since $s - t$ is an integer).

Statements 2 and 3 have both been proved, therefore the Lemma is proved.

Proof of Euclid's GCD Since d was arbitrarily chosen (it could be any divisor), the above shows that m, n and n, $m \pmod n$ have the same (non-empty) set of common divisors, which necessarily includes the largest element in the set, which is the GCD.

QED

3.2. **The Non-naïve Way.** E1. [Find remainder.] Divide m by n and let r be the remainder. (We will have $0 \leq r < n$.)

E2. [Is it zero?] If $r = 0$, the algorithm terminates; n is the answer.

E3. [Interchange] Set $m \leftarrow n, n \leftarrow r$, and go back to step E1.

After step E1, we have $m = qn + r$ for some integer q. If $r = 0$, then m is a multiple of n, and clearly in such a case n is the greatest common divisor of m and n. If $r \neq 0$, note that any number which divides both m and n must divide $m - qn = r$, and any number which divides both n and r must divide $qn + r = m$; so the set of divisors of m, n is the same as the set of divisors of n, r and, in particular, the *greatest* common divisor of m, n is the same as the greatest common divisor of n, r. Therefore step E3 does not change the answer to the original problem.

QED