Demitri Gentile

Professor Alnaji

CS M135

Module 16 : Assignment Complexity

* **Algorithm Complexity/Efficiency**  
  Algorithm Complexity/Efficiency is the amount of computer time/computations it takes to run an algorithm.  
  (When it comes time to put an algorithm to work or choose between competing algorithms, we need a way to measure and compare algorithms. There are many different things we could measure about an algorithm: the number of lines of code to express, how much time it takes to program and debug, the amount of memory used while running, and time taken to run are all things we might care about. But in general, the most important metric is usually "how much work does it require for a problem of size n"?)  
  <https://www.oercommons.org/courses/elementary-data-structures/view> (6.1 Algorithm Efficiency)
* **Big-O Analysis**  
  The exact way in which we count units of work in an algorithm is not as important as the degree to which the algorithm depends on the size of its input. An algorithm that always involves the same amount of work is more efficient than one where the work grows as a function of the input size (at least once we pass a certain problem size).  
  <https://www.oercommons.org/courses/elementary-data-structures/view> (6.2 Big-O Analysis)
* **Space-Time/Time-Memory Tradeoff**

The Space-Time/Time-Memory tradeoff is when an algorithm/program uses more space (memory) but decreases time, or vice versa.  
(A space-time or time-memory tradeoff in computer science is a case where an algorithm or program trades increased space usage with decreased time. Here, space refers to the data torage consumed in performing a given task (RAM, HDD, etc), and time refers to the time consumed in performing a given task (computation time or response time).)  
<https://www.oercommons.org/courses/elementary-data-structures/view> (6.5 Space-Time Tradeoff)  
  
  
**Some other sources used:**

<https://www.oercommons.org/courses/algorithmic-efficiency>

<https://www.oercommons.org/courses/cs-principles-2019-2020-1-11-18-algorithms-detour-shortest-path/view>

<https://www.oercommons.org/courses/think-complexity>