MATH 282 Analysis of Algorithm’s Complexity

**Algorithm:** Matrix multiplication

**Factor to be analyzed:** Time (number of steps, speed)

**Situation to be analyzed:** Square matrix (*n* rows by *n* columns)

**Explanation of situation:** Number of rows is same as number of columns (*n* by *n*)

Simplifies analysis

**Key step (reflects work done):** Accessing the matrix (getting or setting an element)

**Parameter for analysis:** *n* is the number of rows or columns in the square matrix

*(what determines how the algorithm’s efficiency changes as the size of data increases?)*

**Questions/Process:**

* Are there any steps which are not simple steps (comparable to the key step)?
  + If so, what is the complexity of those steps (relative to the key step)? Factor into the analysis.
* If desired, count the number of times each step is carried out (or just the key step).
* Identify each loop and determine how many times the loop is carried out (in relation to *n*).
* How are the loops related?
  + If nested, multiply the steps.
  + If separate, add the steps.
* Eliminate any constants and any lower-level terms.

To create the *n* by *n* product matrix involves *n*2 steps

To multiply the matrices, there are three loops

First (outermost) loop is done *n* times (once for every row)

Second loop is done *n* times (once for every column)

Third (innermost) loop is done *n* times (once for every column in left op, or row in right op)

The loops are nested, so multiply steps: *n* \* *n* \* *n* = *n*3

[Detailed analysis: 2*n*3 steps to get elements, *n*2 steps to set elements]

Total steps are *n*2 + *n*3, so order is *n*3

**Result:** Algorithm is O( *n*3 )