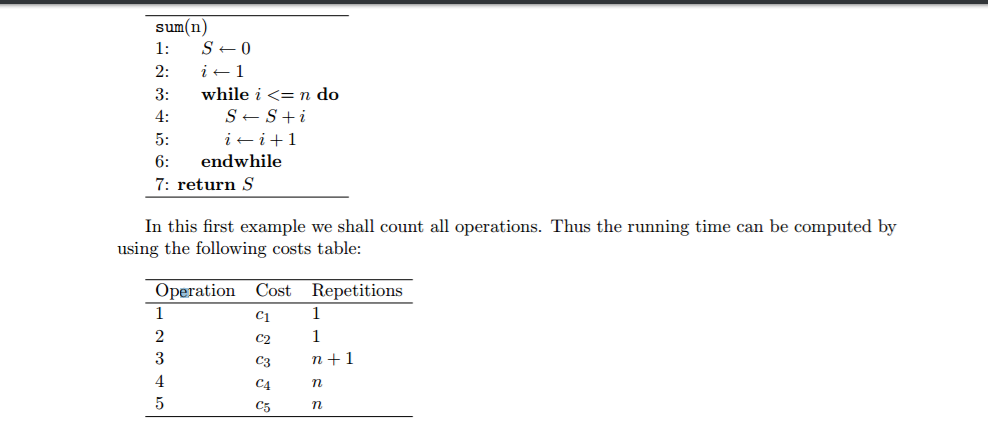
The analysis of time efficiency is based on estimating the time needed to execute the algorithm. Since the main aim is to establish how depends the running time on the problem’s size the running time is not really the physical time corresponding to the execution of the algorithm on a computer but an estimate of the number of basic operations executed by the algorithm. Let us denote by T(n) the running time when the input’s size is n. To estimate T(n) we have to choose a computing model and a measuring unit. We shall consider a computing model (also called random access machine) having the following characteristics:

• All processing steps are sequentially executed (there are not operations simultaneously executed).

• The time of executing the basic operations does not depend on the values of the operands.

• The access time to data does not depend on their address (there are no differences between processing the first element of an array and processing the last element). Choosing a measuring unit means to establish which are the basic operations and considering as measuring unit the time needed to execute them. Thus the running time will be expressed as a number of operations. As basic operations are considered, usually, the assignment, arithmetical operations (addition, subtraction, multiplication and division), comparisons and logical operations (negation, disjunction and conjunction). Even if on a computer the time needed by the logical circuits is not the same for all operations above, in order to simplify the analysis we shall consider that all operations have a cost equal to one. We shall consider that the running time of the algorithm is the sum of the number of execution of each basic operation. In practice (since the analysis aim is to compare algorithms) is no need to count all operations involved in an algorithm but only the most important of them, called dominant operation. The dominant operation is the most frequent one (usually it appears in the body of the innermost loop). For instance for a search algorithm the dominant operation is that of comparison between the value we are searching for and the elements of the array. Example 1. Let us consider the problem of computing Pn i=1 i. The input data is n and the problem’s size can be considered to be n. The corresponding algorithm can be described as follows:



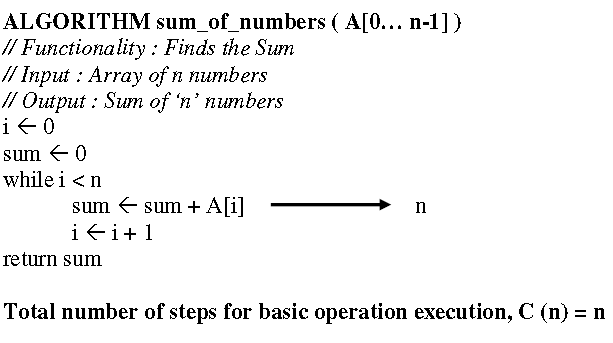
By summing up the cost of all operations we shall obtain: T(n) = n(c3 +c4 +c5)+c1 +c2 +c3.

**Framework for Analysis**

We use a hypothetical model with following assumptions  
• Total time taken by the algorithm is given as a function on its input size  
• Logical units are identified as one step  
• Every step require ONE unit of time  
• Total time taken = Total Num. of steps executed  
Input’s size: Time required by an algorithm is proportional to size of the problem  
instance. For e.g., more time is required to sort 20 elements than what is required to sort 10 elements.  
Units for Measuring Running Time: Count the number of times an algorithm’s basic operation is executed.

(Basic operation: The most important operation of the algorithm,  
the operation contributing the most to the total running time.) For e.g., The basic operation is usually the most time-consuming operation in the algorithm’s innermost loop.

Consider the following example:



**NOTE:  
Constant of fastest growing term is insignificant:**

Complexity theory is an Approximation theory. We are not interested in exact time required by an algorithm to  
solve the problem. Rather we are interested in order of growth. i.e  
 How much faster will algorithm run on computer that is twice as fast?  
 How much longer does it take to solve problem of double input size?  
We can crudely estimate running time by  
http://www.allsyllabus.com/aj/note/Computer_Science/Analysis_and_Design_of_Algorithms/Unit2/index2.PNG  
Where,  
T (n): running time as a function of n.  
Cop : running time of a single operation.  
C (n): number of basic operations as a function of n.