The Art of Computer Programming

1.2.10 Analysis of an Algorithm

Algorithm M

Donald Knuth

August 5, 2019

In chapter one of the first volume, Donald Knuth works through the derivation of determining the distribution for A (Big O). The variable A represents the number of times the algorithm must branch to assign a new value for the maximum value in the list. What is particulary interesting about this problem is that it pulls together quite a bit of the material discussed in earlier sections. In this document the details of this calculation are explored.

1 The Algorithm

Given n elements we will find m and j such that,

$$m = X[j] = \max_{0 \leq i \leq n-1} X[i]$$

where j is the largest index that satisfies this relation.

M1. [Initialize.] Set $j \leftarrow n-1$, $k \leftarrow n-2$, $m \leftarrow X[n-1]$

M1.1 Where m is set to $X[j] = \max_{0 \leq i \leq n-1} X[i]$

M2. [All tested?] If $k = 0$, the algorithm terminates.
M3. [Compare.] If $X[k] \leq m,$ go to M5
M4. [Change m.] Set $j \leftarrow k$, $m \leftarrow X[k]$, and m is a new current maximum.
M5. [Decrease k.] Decrease k by one and return M2.

In Python this may look like:

```python
def find_maximum(numbers):
  m = numbers[0]
  a = 0
  for idx in range(1, len(numbers)):
    m_test = numbers[idx]
    if m_test > m:
      a += 1
      m = numbers[idx]
  return a
```

Let A be the number of times the temporary maximum value is changed to a new value. With $n-1$ the number of times the algorithm has to check whether the integer is greater than the current temporary maximum. The quantity $n-1-A$ is the number of times the algorithm does not have to change the current maximum.

M1.	[Initialize.]	This happens 1 time.
M2.	[All tested?]	This happens n times.
M3.	[Compare.]	This happens $n-1$ times
M4.	[Change m.]	This happens A times.
M5.	[Decrease k.]	This happens $n-1$ times.

The number of times a given path is taken is given in Table 1. Typically the analysis would calculate a minimum value of A, as well as its maximum value. It would be useful to also have the average of A along with the standard deviation to understand how close to the average that A is expected to get. Well, the minimum value of A, the number of times the temporary maximum value is changed to a new value, happens if

$$X[n] = \max_{1 \leq k \leq n} X[k]$$

Table 1: Branches and iterations.

Step number	**Number of times**
$M1$	1
$M2$	n
$M3$	$n-1$
$M4$	A
$M5$	$n-1$

The maximum value will happen if each element of the list needs to be checked, or after $n-1$ iterations, thus the average value has to lie between:

$$0 <= A <= n-1$$

2 An Example

Table 2: The possible values of A.

Situation	**Value of A**	**Situation**	**Value of A**
$X[0] < X[1] < X[2] < X[3]$	3	$X[3] < X[0] < X[2] < X[1]$	1
$X[1] < X[0] < X[2] < X[3]$	2	$X[0] < X[3] < X[2] < X[1]$	1
$X[2] < X[0] < X[1] < X[3]$	2	$X[2] < X[3] < X[0] < X[1]$	1
$X[0] < X[2] < X[1] < X[3]$	2	$X[3] < X[2] < X[0] < X[1]$	1
$X[1] < X[2] < X[0] < X[3]$	2	$X[0] < X[2] < X[3] < X[1]$	1
$X[2] < X[1] < X[0] < X[3]$	1	$X[2] < X[0] < X[3] < X[1]$	1
$X[2] < X[1] < X[3] < X[0]$	0	$X[1] < X[0] < X[3] < X[2]$	1
$X[1] < X[2] < X[3] < X[0]$	0	$X[0] < X[1] < X[3] < X[2]$	2
$X[3] < X[2] < X[1] < X[0]$	0	$X[3] < X[1] < X[0] < X[2]$	1
$X[2] < X[3] < X[1] < X[0]$	0	$X[1] < X[3] < X[0] < X[2]$	1
$X[1] < X[3] < X[2] < X[0]$	0	$X[0] < X[3] < X[1] < X[2]$	2
$X[3] < X[1] < X[2] < X[0]$	0	$X[3] < X[0] < X[1] < X[2]$	2

The average value of A when $n = 4$ is

$$[(6)0 + (10)1 + (7)2 + (1)3]/24 = 27/24 = 9/8$$

In the Analysis of an Algorithm Knuth works out the case where $n = 3$ and in that case the average value of A is $5/6$. Consider the case where $x[2]$ is the greatest.

Table 3: The possible values of A.

Situation	**Realization**	**Value of A**
$X[1] < X[0] < X[3] < X[2]$	$[1, 0, 3, 2]$	1
$X[0] < X[1] < X[3] < X[2]$	$[0, 1, 3, 2]$	2
$X[3] < X[1] < X[0] < X[2]$	$[2, 1, 3, 0]$	1
$X[1] < X[3] < X[0] < X[2]$	$[2, 0, 3, 1]$	1
$X[0] < X[3] < X[1] < X[2]$	$[0, 2, 3, 1]$	2
$X[3] < X[0] < X[1] < X[2]$	$[1, 2, 3, 0]$	2

Since the maximum element is at $x[2]$ and the iteration starts at $x[0]$ it is only the magnitudes of $x[0]$ and $x[1]$ that make a difference. If $x[0] < x[1]$ then there are 2 shifts of the maximum, otherwise there is only one. Thus, the determining factor is the location of the maximum and the possible permutations of elements that exist before it. The *probability* that A has the value k will be,

$$p_{nk} = (\text{number of permutations of } n \text{ objects for which } A = k)/n!$$