Jacob Elkins

Doc

Data Structures

March 29, 2018

The Big O Notation

The big o notation is used to describe performance or complexity of an algorithm. This is specifically used to describe the worst-case scenario, and to describe the execution time required or the space used by an algorithm (Bell). In big o notation, there are four core common orders of growth, O(1), O(N), O(N^2), and O(2^N).

The O(1) notation "describes an algorithm that will always execute in the same time (or space) regardless of the size of the input data set" (Bell).

```
bool IsFirstElementNull ( IList<string> elements)
{
return elements[0] == null;
}
```

The O(N) notation "describes an algorithm whose performance will grow linearly and in direct proportion to the size of the input data set" (Bell).

```
bool ContainsValue(IList<string> elements, string value)
{
   foreach (var element in elements)
   {
      if (element == value) return true;
   }

   return false;
}
```

The example above is a demonstration of how "big o favors the worst-case performance scenario; matching string could be found during any iteration of the for loop and the function would return early, but big o notation will always assume the upper limit where the algorithm will perform the max number of iterations" (Bell).

The O(N^2) notation “represents and algorithm whose performance is directly proportional to the square of the size of the input data set. This is common with algorithms that involve nested iterations over the data set” (Bell). The deeper the nesting goes, the exponent on the notation will increase by one.

```
bool ContainsDuplicates(IList<string> elements)
{
   for (var outer = 0; outer < elements.Count; outer++)
   {
      for (var inner = 0; inner < elements.Count; inner++)
      {
         // Don't compare with self
         if (outer == inner) continue;

         if (elements[outer] == elements[inner]) return true;
      }
   }

   return false;
}
```

The O(2^N) notation “denotes an algorithm whose growth doubles with each addition to the input data set. The growth curve of an O(2^N) function is exponential – starting off very shallow, then rising meteorically” (Bell).

```
int Fibonacci(int number)
{
   if (number <= 1) return number;

   return Fibonacci(number - 2) + Fibonacci(number - 1);
}
```

The two main uses for big o notation is in mathematics and computer science. In mathematics, it is commonly used to describe how closely a finite series approximates a given function. In computer science it is used in the analysis of algorithms.

Biography

Bell, R. (2009, June 09). A beginner's guide to Big O notation. Retrieved March 30, 2018, from https://rob-bell.net/2009/06/a-beginners-guide-to-big-o-notation/