CSCA48 Winter 2018
Week 10:Algorithm Analysis

Marzieh Ahmadzadeh, Nick Cheng

University of Toronto Scarborough

UNIVERSITY OF TORONTO SCARBOROUGH

Algorithm

- Definition:
  - Solving a problem step-by-step in finite amount of time.
- Analysis:
  - How much of computer resources does an algorithm use?
    - Memory Space
      - We are not interested in this for now
    - CPU time
      - Dependent to input size
- How to measure running time?
  - Experimental method
  - Theoretical analysis

Experimental Method

- Implement the algorithm using one of the programming languages.
- Start the timer
- Execute the algorithm
- Stop the timer
- Plot the results to compare.

Disadvantage of experimental method

- It needs implementing the algorithms
- Results are only indicative of the running time on data included in the experiment
- The same hardware and software in similar execution atmosphere must be used to compare two algorithms.

Theoretical Analysis

- Requirement
  - Uses a pseudo-code instead of a code to describe the algorithm
  - Allows us to compute the running time of the algorithm independent of the environment and language.
  - Takes into account the worst case scenario for inputs
    - Best case is not the reality.
    - Average case coverage is difficult to determine
  - Describe the execution time as a function of input size.
  - Assumes that every primitive operation takes one `unit of time` in a certain machine model.
    - expression evaluation, assignment, indexing into an array, function call, returning from a function, comparison, follow an object reference

Example

- In worst case scenario:

```
Function  find_max(a_list, n)
    max ← a_list[0]
    for i ← 1 to n − 1 do
        if a_list[i] > max
             max = a_list[i]
    return max
```

Code	# of Operations
max ← a_list[0]	2
for i ← 1 to n − 1	$2n + 1$
if a_list[i] > max	2(n-1)
max = a_list[i]	2(n-1)
Increment i	2(n-1)
return max	1
	8n - 2

`a_list` is a list containing n comparable data

If `n = 100` then 798 operations is required to execute this code
If `n = 10` then 78 operations is required

Time Complexity

- Is the time that is taken to run an algorithm when `maximum` amount of resources is required.
  - i.e. worst case complexity
- Since complexity varies with different input size, it is shown by a function:
  - `T(n) = r`, where `n`: input size, `r`: running time, `T(n)`: worst case complexity.
  - $n \in N$, $r \in R^{+}$
  - For previous example: `T(n) = 8n - 2`
- We don't need to know the exact value of T(n), instead an `upper bound` (i.e. `dominant term`) is used as an estimate.

Upper-bound (1)

- Suppose `T(n) = 8n - 2`
- We define the order of magnitude as the dominant part of T(n) that increases fastest as the value of n increases.
  - Order of magnitude is n (i.e. $O(n)$) $since\ \ \ni c,\ \forall n \in N: T(n) \leq c.f(n)$
  - $T(n) = 8n - 2, f(n) = n, c = 8$
- Big-O notation is used to show the order of magnitude.
  - `T` has order `n`
  - `T` is big- oh of `n`

Upper-bound (2)

- Suppose `T(n) = 2n + 10`, then 2n is only dominant if n is large enough

- So we define asymptotic upper-bound:
  - Let $f: N \rightarrow R^{+}$, $\mathrm{T}: N \rightarrow R^{+}$ be functions then:
    $$O(f(n)) = \{T(n) \mid \exists\, c, \exists\, b, \forall n \geq b: T(n) \leq c.f(n)\}$$

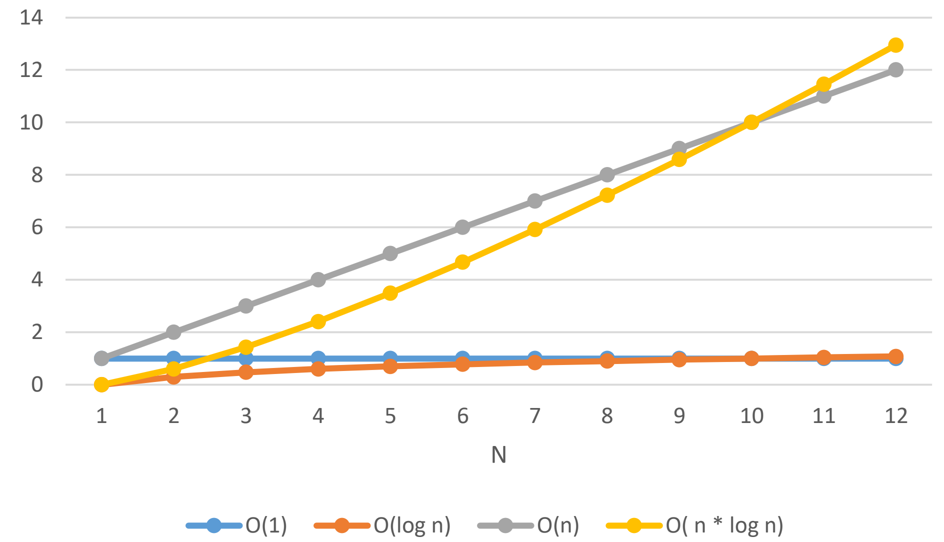
- $T(n) = (2n + 10), f(n) = n\,, c = 4, n \geq 11$

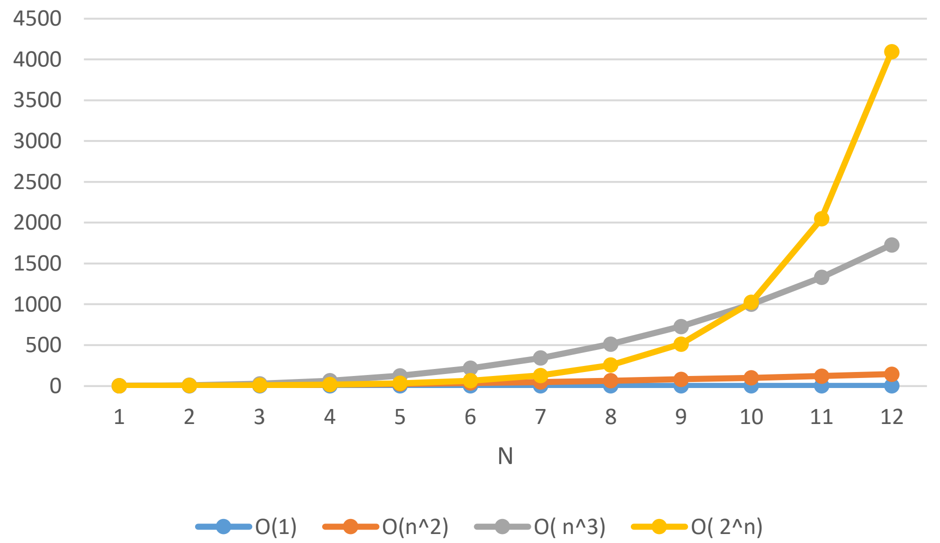
Example:

- If $T(n) = 3n^2 + 5n - 7$ then running time is O(n^2)
- Find c and b such that $3n^2 + 5n - 7 \leq cn^2$
  - c = 4, b = 2
  - So $3n^2 + 5n - 7 \in$ O(n^2)

Important Functions

- Constant : O(*1*)
- Logarithmic: O(log *n*)
- Linear: O(*n*)
- N-Log-N: O(*n* log *n*)
- Quadratic: O(n^2)
- Cubic : O(n^3)
- Exponential: O(2^n)





Examples:

- Find the running time of
  - Insertion & removal into/from
    - a list at a certain index where it is implemented by a single linked list (with a pointer to tail and without a pointer to tail)
    - the end of a list, where it is implemented by a single linked list (with and without tail)
    - the end of a list, where it is implemented by a double linked list
  - Appending to a list, implemented by a single linked list
  - Finding an item in the list
  - Insertion & removal into/from a
    - dictionary
    - PQ
    - BST
    - Heap
  - Bubble sort