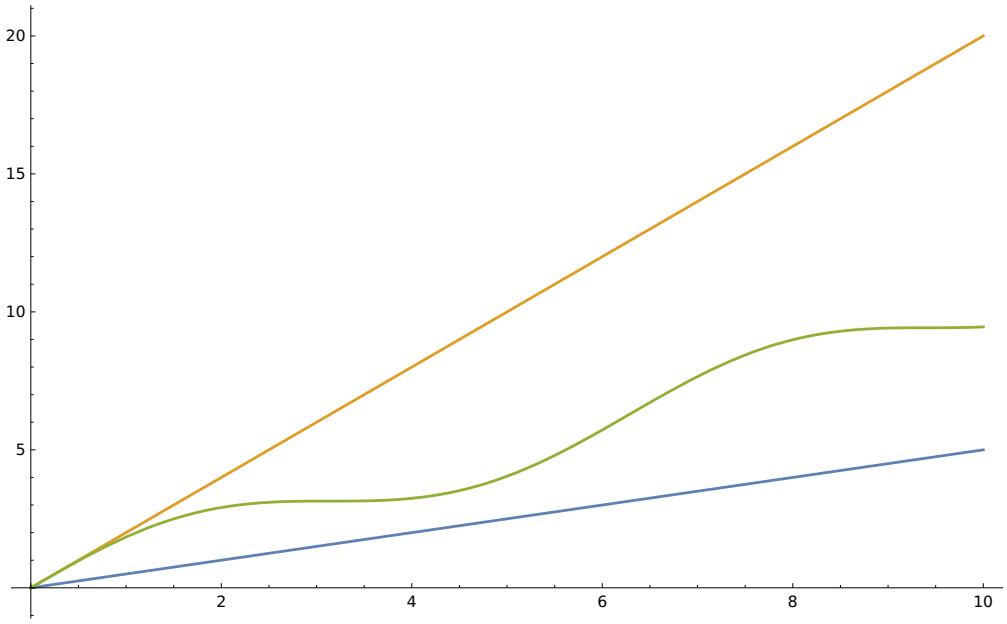
CSC165H1: Mathematical Expression and Reasoning for Computer Science **Fall 2018**

Lecture 14: October 30

Lecturer: Tom Fairgrieve *Note-Taker: Jim Gao (jim.gao@mail.utoronto.ca)*

14.1 Some Notes

$\mathcal{O}(n)$ refers to the set of functions that eventually stays between 2 lines through the origin.



14.2 Analyzing Code Runtimes

Consider the following code:

```
def print_list(lst: List[bool]) -> None:
        for item in lst:
                print (item)
```

We let n to indicate the size of `lst`.

No matter which method of counting we choose, all of them are $\Theta(n)$.

Definition 14.1 *A "basic operation" is any block of code whose running time does not depend on the size of the input/data.*

Some "basic operations" are:

- Comparison (`==`, `<`, `>`)
- Arithmetic (`+`, `*`)
- Assignment (`x=y`)

With this definition, we can prove the running time of `print_items` is $\Theta(n)$.

Claim 14.2 *The running time for* ***`print_items`*** *is* $\Theta(n)$.

Proof: For this algorithm, each iteration of the loop can be counted as a single basic operation, as nothing in it depends on the size of the loop.

The running time depends on the number of loop iterations. Since this is a for-loop over the `lst` argument, we know the loop runs n times.

Thus, the number of basic operations performed is $n \times 1 = n$ an dso the running time is $\Theta(n)$. ■

14.2.1 Summary of this approach

1. Identify your measure of the input/data size.
2. Identify the blocks of code that can be counted as a single basic operation, since they don't depend on the input size.
3. Identify any loops in the code. Figure out how many times the loops run, based on the input size.
4. Combine these observations to get an expression for the number of basic operations.
5. Connect this expresstion to Θ notation.

14.2.2 Example: Typical

Consider the function:

```
def print_sums(lst: List[float]) -> None:
        for item1 in lst:
                for item2 in lst:
                        print (item1 + item2)
```

Claim 14.3 *The function* ***`print_sums`*** *runs in time* $\Theta(n^2)$*, where* n *is the size of the list.*

Proof: Let n be the length of the list `lst`.

The inner loop (L3-4) runs n times, and each iteration has 1 basic operation (yellow).

The outer loop (L2-4) runs n times, and each iteration has n basic opeations.

So, the total number of basic operations is $n \times n = n^2$.

Then, we can conclude that the running time is $\Theta(n^2)$. ■

14.2.3 Example: Nesting Levels

Consider this function:

```
def f(lst: List[int]):
        for item in lst:
                for i in range(10):
                        print (item + i)
```

The runtime for this function is $\Theta(n)$, since the highlighted section is a basic operation. The proof can be done similarly to 14.2.2.

Note: From this example, it is important to not only look at the nesting levels.

14.2.4 Example: While-Loops

```
1 def g(lst: List[int]) -> None:
2         for item in lst:
3                 i = 0
4                 while i < len(lst):
5                         print (item + i)
6                         i = i + 2
```

Note that the inner for-loop (L4-6) is iterated $ceil(n/2)$ times.

Hence, the running time is:

$$n(1 + ceil(n/2)) \in \Theta(n^2)$$

14.2.5 Example: Putting it together

```
def c(lst: List[int]) -> None:
        print ("Here is the list")

        for item in lst:
                print (item)

        for item1 in lst:
                for item2 in lst:
                        print (item1 + item2)
```

The running time is:

$$1 + n + n^2 \in \Theta(n^2)$$