

# Sums and Asymptotics

---

# Reading

---

- Malik Magdon-Ismael. Discrete Mathematics and Computing.
  - Chapter 9

# Overview

---

- Maximum Substring Sum
- Computing Sums
- Asymptotics: Big-Theta, Big-Oh and Big-Omega
- Integration Method

# Maximum Substring Sum

---

- Look at this sequence of numbers:

$1, -1, -1, 2, 3, 4, -1, -1, 2, 3, -4, 1, 2, -1, -2, 1$

- What is the largest sum of 7 consecutive numbers?

$$2 + 3 + 4 - 1 - 1 + 2 + 3 = 12$$

– This is known as the max substring sum

- More generally, compute the maximum substring sum for

$$a_1, a_2, a_3, a_4, \dots, a_{n-1}, a_n$$

– where  $n$  measures the size/length of the input

# Maximum Substring Sum, cont'd

---

- Can you come up with an algorithm for max substring sum?
  1. Iterate over all pairs  $(i, j)$  of start and end positions.
    - Brute-force but effective and easy to analyze
    - How many loops do we have?
    - 3 loops: one loop for each of  $i$  and  $j$ , and 1 loop to calculate sum
  2. Iterate over all starting positions  $i$  and ending positions  $j > i$ .
    - More efficient than 1
    - How many loops do we have?
    - 2 loops: one loop each over all  $i$  and all  $j$
  3. Divide and conquer
    - Divide array into two halves and recursively calculate max in each half
    - Also look at max sum that contains the middle
  4. Suppose you are keeping track of the current cumulative sum
    - What happens if a sum is negative (assuming positive numbers exist)?
      - Should reset sum to next number
    - If current sum is larger than the largest so far, set largest to current

# Maximum Substring Sum, cont'd

- Different algorithms have different runtimes (check book exercises for specific algorithms)

- three-loop version:  $T_1 = 2 + \sum_{i=1}^n \left[ 2 + \sum_{j=1}^n \left( 5 + \sum_{k=i}^j 2 \right) \right]$

- What does  $\sum_{i=1}^n$  mean?
- Sum all entries, increasing  $i$  by 1 each time

- two-loop version:  $T_2(n) = 2 + \sum_{i=1}^n \left( 3 + \sum_{j=i}^n 6 \right)$

- A recursive algorithm:

$$T_3(n) = \begin{cases} 3 & n = 1 \\ 2T_3\left(\frac{1}{2}n\right) + 6n + 9 & n > 1 \text{ (even)} \\ T_3\left(\frac{1}{2}(n+1)\right) + T_3\left(\frac{1}{2}(n-1)\right) + 6n + 9 & n > 1 \text{ (odd)} \end{cases}$$

- A fast algorithm:  $T_4(n) = 5 + \sum_{i=1}^n 10$

- But which one is fastest?

# Evaluate Runtimes

- Let's plug in some values of  $n$  and see what happens

$n$	1	2	3	4	5	6	7	8	9	10
$T_1(n)$	11	29	58	100	157	231	324	438	575	737
$T_2(n)$	11	26	47	74	107	146	191	242	299	362
$T_3(n)$	3	27	57	87	123	159	195	231	273	315
$T_4(n)$	15	25	35	45	55	65	75	85	95	105

- Which algorithm is best?
  - Clearly,  $T_1$  is worse than  $T_2$  but hard to compare  $T_2$  and  $T_3$
  - $T_4$  seems best on most inputs
- We need:
  - Simple formulas for  $T_1(n), \dots, T_4(n)$ : we need to compute sums and solve recurrences.
  - A way to compare runtime-*functions* that captures the essence of the algorithm.

# Computing Sums: Tool 1: Constant Rule

- $S_1 = \sum_{i=1}^{10} 3$   
 $= 3 + 3 + 3 + 3 + 3 + 3 + 3 + 3 + 3 + 3 = 3 \times 10$

- $S_2 = \sum_{i=1}^{10} j$   
 $= j + j + j + j + j + j + j + j + j + j = j \times 10$

- $S_3 = \sum_{i=1}^{10} i$   
 $= 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10$   
 $= \frac{1}{2} \times 10 \times 11$

- The *index of summation* is  $i$  in these examples.
- **Constants (independent of summation index) can be taken outside the sum.**

$$S_1 = \sum_{i=1}^{10} 3 = 3 \sum_{i=1}^{10} 1 = 3 \times 10$$

$$S_2 = \sum_{i=1}^{10} j = j \sum_{i=1}^{10} 1 = j \times 10$$

## Computing Sums: Tool 2: Addition Rule

---

$$\begin{aligned} S &= \sum_{i=1}^5 (i + i^2) \\ &= (1 + 1^2) + (2 + 2^2) + (3 + 3^2) + (4 + 4^2) + (5 + 5^2) \quad \text{[rearrange terms]} \\ &= (1 + 2 + 3 + 4 + 5) + (1^2 + 2^2 + 3^2 + 4^2 + 5^2) \\ &= \sum_{i=1}^5 i + \sum_{i=1}^5 i^2 \end{aligned}$$

- **The sum of terms added together is the addition of the individual sums**

$$\sum_i (a(i) + b(i) + \dots) = \sum_i a(i) + \sum_i b(i) + \dots$$

# Computing Sums: Tool 3: Common Sums

---

$$\sum_{i=k}^n 1 = n - k + 1$$

$$\sum_{i=1}^n f(x) = nf(x)$$

$$\sum_{i=0}^n r^i = \frac{1 - r^{n+1}}{1 - r} \quad [r \neq 1]$$

$$\sum_{i=1}^n i = \frac{1}{2}n(n + 1)$$

$$\sum_{i=1}^n i^2 = \frac{1}{6}n(n + 1)(2n + 1)$$

$$\sum_{i=1}^n i^3 = \frac{1}{4}n^2(n + 1)^2$$

## Computing Sums: Tool 3: Common Sums, cont'd

---

$$\sum_{i=0}^n 2^i = 2^{n+1} - 1$$

$$\sum_{i=0}^n \frac{1}{2^i} = 2 - \frac{1}{2^n}$$

$$\sum_{i=1}^n \log i = \log(n!)$$

# Computing Sums: Example

$$\sum_{i=1}^n (1 + 2i + 2^{i+2}) =$$

$$= \sum_{i=1}^n 1 + \sum_{i=1}^n 2i + \sum_{i=1}^n 2^{i+2} \quad \text{[addition rule]}$$

$$= \sum_{i=1}^n 1 + 2 \sum_{i=1}^n i + 4 \sum_{i=1}^n 2^i \quad \text{[constant rule]}$$

$$= \sum_{i=1}^n 1 + 2 \sum_{i=1}^n i + 4(\sum_{i=0}^n (2^i) - 1) \quad \text{[algebra]}$$

$$= n + 2 \times \frac{1}{2} n(n + 1) + 4 \times (2^{n+1} - 1 - 1) \quad \text{[common sums]}$$

$$= n + n(n + 1) + 2^{n+3} - 8 \quad \text{[algebra]}$$

# Computing Sums: Tool 3: Nested Sum Rule

- To compute a nested sum, start with the innermost sum and proceed outward

$$\begin{aligned} S_1 &= \sum_{i=1}^3 \sum_{j=1}^3 1 \\ &= \sum_{j=1}^3 1 + \sum_{j=1}^3 1 + \sum_{j=1}^3 1 = 3 + 3 + 3 = 9 \end{aligned}$$

- Note that the  $j$  variables are local to each sum (same as in a loop in your code)

$$\begin{aligned} S_2 &= \sum_{i=1}^3 \sum_{j=1}^i 1 \\ &= \sum_{j=1}^1 1 + \sum_{j=1}^2 1 + \sum_{j=1}^3 1 = 1 + 2 + 3 = 6 \end{aligned}$$

- More generally

- Using the fact that  $\sum_{j=1}^i 1 = i$ :

$$S(n) = \sum_{i=1}^n \sum_{j=1}^i 1 = \sum_{i=1}^n i = \frac{1}{2}n(n+1)$$

## Computing a formula for $T_2$

$$\begin{aligned}T_2(n) &= 2 + \sum_{i=1}^n \left( 3 + \sum_{j=i}^n 6 \right) && \text{[sum rule]} \\&= 2 + 3 \sum_{i=1}^n 1 + \sum_{i=1}^n \sum_{j=i}^n 6 && \text{[constant rule]} \\&= 2 + 3n + \sum_{i=1}^n \sum_{j=i}^n 6 && \text{[common sum]} \\&= 2 + 3n + 6 \sum_{i=1}^n \sum_{j=i}^n 1 && \text{[constant rule]} \\&= 2 + 3n + 6 \sum_{i=1}^n (n - i + 1) && \text{[innermost sum]} \\&= 2 + 3n + 6(n + (n - 1) + \dots + 1) && \text{[common sum]} \\&= 2 + 3n + 6 \times \frac{1}{2} n(n + 1) && \text{[common sum]} \\&= 2 + 6n + 3n^2 && \text{[algebra]}\end{aligned}$$

## Practice: Compute a Formula for the Sum:

$$\sum_{i=1}^n \sum_{j=1}^i ij$$

$$\sum_{i=1}^n \sum_{j=1}^i ij = \sum_{i=1}^n \sum_{j=1}^i ij \quad \text{[innermost sum]}$$

$$= \sum_{i=1}^n i \sum_{j=1}^i j \quad \text{[constant rule]}$$

$$= \sum_{i=1}^n i \frac{1}{2} i(i+1) \quad \text{[common sum]}$$

$$= \frac{1}{2} \sum_{i=1}^n (i^3 + i^2) \quad \text{[algebra, constant rule]}$$

$$= \frac{1}{2} (\sum_{i=1}^n i^3 + \sum_{i=1}^n i^2) \quad \text{[sum rule]}$$

$$= \frac{1}{8} n^2 (n+1)^2 + \frac{1}{12} n(n+1)(2n+1) \quad \text{[common sums]}$$

$$= \frac{1}{12} n + \frac{3}{8} n^2 + \frac{5}{12} n^3 + \frac{1}{8} n^4 \quad \text{[algebra]}$$

# Summary of Max Substring Sum Algorithms

- Runtimes

$$T_1(n) = 2 + \frac{31}{6}n + \frac{7}{2}n^2 + \frac{1}{3}n^3$$

$$T_2(n) = 2 + 6n + 3n^2$$

$$3n(\log_2 n + 1) - 9 \leq T_3(n) \leq 12n(\log_2 n + 3) - 9$$

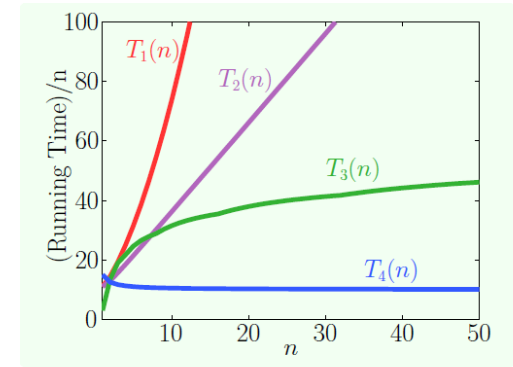
$$T_4(n) = 5 + 10n$$

– (“simple” formulas for  $T_1(n), \dots, T_4(n)$ )

- **So, which algorithm is best?**

- Computers solve problems with big inputs. We care about large  $n$ .
- Compare runtimes *asymptotically* in the input size  $n$ . That is  $n \rightarrow \infty$
- Ignore additive and multiplicative constants (minutia). We care about *growth rate*.

- Algorithm 4 is *linear* in  $n$ ,  $\frac{T_4(n)}{n} \rightarrow \text{constant}$ .



# Asymptotically Linear Functions: $\Theta(n)$ , big-Theta-of- $n$

- We say an algorithm runs in “big-Theta-of- $n$ ” time, i.e.,

$$T \in \Theta(n), \text{ if there are positive constants } c, C \text{ for which}$$

$$c \cdot n \leq T(n) \leq C \cdot n$$

$$\frac{T(n)}{n} \xrightarrow{n \rightarrow \infty} \begin{cases} \infty & T \in \omega(n), & "T > n" \\ \text{constant} > 0 & T \in \Theta(n), & "T = n" \\ 0 & T \in o(n), & "T < n" \end{cases}$$

- Linear means in  $\Theta(n)$ :

$$2n + 7, \quad 2n + 15\sqrt{n}, \quad 10^9n + 3, \quad 3n + \log n, \quad 2^{\log_2 n + 4}$$

- Not linear means not in  $\Theta(n)$ :

$$10^{-9}n^2, \quad 10^9\sqrt{n} + 15, \quad n^{1.0001}, \quad n^{0.9999}, \quad n \log n, \quad \frac{n}{\log n}, \quad 2^n$$

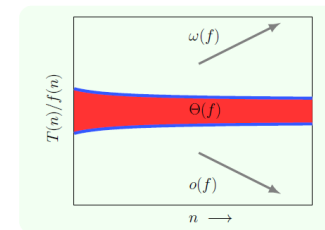
- Other runtimes frequently appearing in practice

log	linear	loglinear	quadratic	cubic	super-polynomial	exponential	factorial	BAD
$\Theta(\log n)$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(n^3)$	$\Theta(n^{\log n})$	$\Theta(2^n)$	$\Theta(n!)$	$\Theta(n^n)$

# General Asymptotics: $\Theta(f)$ , big-Theta-of- $f$

- Sometimes, we want to measure performance w.r.t. a specific function  $f$

$$\frac{T(f)}{f(n)} \xrightarrow{n \rightarrow \infty} \begin{cases} \infty & T \in \omega(f), & "T > f" \\ \text{constant} > 0 & T \in \Theta(f), & "T = f" \\ 0 & T \in o(f), & "T < f" \end{cases}$$



$T \in o(f)$	$T \in O(f)$	$T \in \Theta(f)$	$T \in \Omega(f)$	$T \in \omega(f)$
$"T < f"$	$"T \leq f"$	$"T = f"$	$"T \geq f"$	$"T > f"$
	$T(n) \leq Cf(n)$	$cf(n) \leq T(n) \leq Cf(n)$	$cf(n) \leq T(n)$	

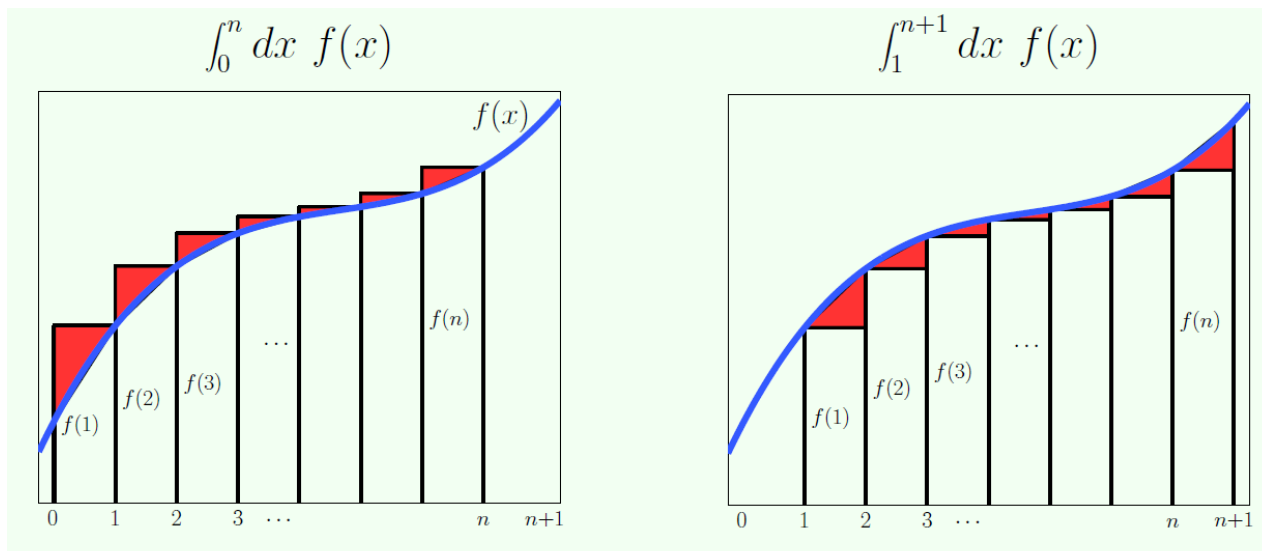
- Examples:

- For polynomials, growth rate is the highest order

$$\begin{aligned} \Theta(2n^2) &= n^2 \\ \Theta(n^2 + n\sqrt{n}) &= \Theta(n^2) \\ \Theta(n^2 + \log^{256} n) &= \Theta(n^2) \\ \Theta(n^2 + n^{1.99} \log^{256} n) &= \Theta(n^2) \end{aligned}$$

# The Integration Method

- One application of big-Theta reasoning
  - You can approximate an integral with the upper and lower integration method



- *Theorem [Integration Bound].* For a monotonically increasing function  $f$ ,

$$\int_{m-1}^n f(x) dx \leq \sum_{i=m}^n f(i) \leq \int_m^{n+1} f(x) dx$$

- (If  $f$  is monotonically decreasing, the inequalities are reversed.)

# Integration for Quickly Getting Asymptotic Behavior

- Integer Powers. Set  $f(x) = x^k$ :

$$\sum_{i=1}^n i^k \approx \int_0^n x^k dx$$
$$\int_0^n x^k dx = \frac{n^{k+1}}{k+1}$$
$$\frac{n^{k+1}}{k+1} \in \Theta(n^{k+1})$$

# Integration for Quickly Getting Asymptotic Behavior, cont'd

- Stirling's Approximation for  $\ln n!$ . Set  $f(x) = \ln x$ :

$$\ln n! = \sum_{i=1}^n \ln i \leq \int_1^{n+1} \ln x \, dx$$
$$\int_1^{n+1} \ln x \, dx =$$
$$= [x \ln(x) - x]_1^{n+1}$$
$$= (n+1) \ln(n+1) - n$$

- So finally:

$$((n+1) \ln(n+1) - n) \in \Theta(n \ln n)$$

# Integration for Quickly Getting Asymptotic Behavior, cont'd

- Analyzing a recurrence.  $T_1 = 1; T_n = T_{n-1} + n\sqrt{n} - \ln n$

– First, unfold the recurrence:

$$\begin{aligned}T_n &= T_{n-1} + n\sqrt{n} - \ln n \\T_{n-1} &= T_{n-2} + (n-1)\sqrt{n-1} - \ln(n-1) \\&\vdots \\T_3 &= T_2 + 3\sqrt{3} - \ln 3 \\T_2 &= T_1 + 2\sqrt{2} - \ln 2\end{aligned}$$

– Sum all terms together (note that all  $T_{n-1}, \dots, T_1$  terms cancel out)

$$\begin{aligned}T_n &= 1 + 2\sqrt{2} + \dots + n\sqrt{n} - (\ln 2 + \ln 3 + \dots + \ln n) \\&= \sum_{i=1}^n i\sqrt{i} - \sum_{i=1}^n \ln i\end{aligned}$$

- Why does the 2<sup>nd</sup> sum start counting from 1?

– We know that  $\sum_{i=1}^n i\sqrt{i} \in \Theta(n^{5/2})$

– and  $\sum_{i=1}^n \ln i = \ln n! \in \Theta(n \ln n)$

- What does that mean for  $T(n) = \sum_{i=1}^n i\sqrt{i} - \sum_{i=1}^n \ln i$ ?  
 $T(n) \in \Theta(n^{5/2})$