Sums and Asymptotics
Reading

  – Chapter 9
Overview

- Maximum Substring Sum
- Computing Sums
- Asymptotics: Big-Theta, Big-Oh and Big-Omega
- Integration Method
• 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
  \[ a_1, a_2, a_3, a_4, \ldots, a_{n-1}, a_n \]
  – 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
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

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

• 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 $T_1(n), \ldots, T_4(n)$: 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

- $S_1 = \sum_{i=1}^{10} 3$
  \[= 3 + 3 + 3 + 3 + 3 + 3 + 3 + 3 + 3 = 3 \times 10\]
- $S_2 = \sum_{i=1}^{10} j$
  \[= j + j + j + j + j + j + j + j + j = j \times 10\]
- $S_3 = \sum_{i=1}^{10} i$
  \[= 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = \frac{1}{2} \times 10 \times 11\]

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

\[
S_1 = \sum_{i=1}^{10} 3 = \sum_{i=1}^{10} 1 = 3 \times 10
\]
\[
S_2 = \sum_{i=1}^{10} j = j \sum_{i=1}^{10} 1 = j \times 10
\]
Computing Sums: Tool 2: Addition Rule

\[ S = \sum_{i=1}^{5} (i + i^2) \]

\[ = (1 + 1^2) + (2 + 2^2) + (3 + 3^2) + (4 + 4^2) + (5 + 5^2) \quad \text{[rearrange terms]} \]
\[ = (1 + 2 + 3 + 4 + 5) + (1^2 + 2^2 + 3^2 + 4^2 + 5^2) \]
\[ = \sum_{i=1}^{5} i + \sum_{i=1}^{5} i^2 \]

• The sum of terms added together is the addition of the individual sums

\[ \sum_{i} (a(i) + b(i) + \cdots) = \sum_{i} a(i) + \sum_{i} b(i) + \cdots \]
Computing Sums: Tool 3: Common Sums

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

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

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

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

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

\[
\sum_{i=1}^{n} i^3 = \frac{1}{4} n^2(n + 1)^2
\]
\[ \sum_{i=0}^{n} 2^i = 2^{n+1} - 1 \]

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

\[ \sum_{i=1}^{n} \log i = \log n! \]
Computing Sums: Example

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

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

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

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

\[ = n + n(n + 1) + 2^{n+3} - 8 \quad \text{[algebra]} \]
Computing Sums: Tool 3: Nested Sum Rule

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

\[
S_1 = \sum_{i=1}^{3} \sum_{j=1}^{3} 1 \\
= \sum_{j=1}^{3} 1 + \sum_{j=1}^{3} 1 + \sum_{j=1}^{3} 1 = 3 + 3 + 3 = 9
\]

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

\[
S_2 = \sum_{i=1}^{3} \sum_{j=1}^{i} 1 \\
= \sum_{j=1}^{1} 1 + \sum_{j=1}^{2} 1 + \sum_{j=1}^{3} 1 = 1 + 2 + 3 = 6
\]

• 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 $T_2$

\[
T_2(n) = 2 + \sum_{i=1}^{n} \left( 3 + \sum_{j=i}^{n} 6 \right)
\]

\[
= 2 + 3 \sum_{i=1}^{n} 1 + \sum_{i=1}^{n} \sum_{j=i}^{n} 6
\]

\[
= 2 + 3n + \sum_{i=1}^{n} \sum_{j=i}^{n} 6
\]

\[
= 2 + 3n + 6 \sum_{i=1}^{n} \sum_{j=i}^{n} 1
\]

\[
= 2 + 3n + 6 \sum_{i=1}^{n} (n - i + 1)
\]

\[
= 2 + 3n + 6(n + (n - 1) + \cdots + 1)
\]

\[
= 2 + 3n + 6 \times \frac{1}{2} n(n + 1)
\]

\[
= 2 + 6n + 3n^2
\]

[sum rule]

[constant rule]

[common sum]

[constant rule]

[innermost sum]

[common sum]

[common sum]

[algebra]
Practice: Compute a Formula for the Sum:

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

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

[innermost sum]

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

[constant rule]

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

[common sum]

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

[algebra, constant rule]

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

[sum rule]

$$= \frac{1}{8} n^2 (n + 1)^2 + \frac{1}{12} n(n + 1)(2n + 1)$$

[common sums]

$$= \frac{1}{12} n + \frac{3}{8} n^2 + \frac{5}{12} n^3 + \frac{1}{8} n^4$$

[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 \( T_1(n) \), ..., \( T_4(n) \))

• 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 \( n \to \infty \)
– Ignore additive and multiplicative constants (minutia). We care about growth rate.

• Algorithm 4 is linear in \( n \), \( \frac{T_4(n)}{n} \to \text{constant} \).
Asymptotically Linear Functions: $\Theta(n)$, big-Theta-of-$n$

- We say an algorithm runs in “big-Theta-of-$n$” time, i.e.,
  
  $T \in \Theta(n)$, if there are positive constants $c, C$ for which
  
  $c \cdot n \leq T(n) \leq C \cdot n$

\[
\lim_{n \to \infty} \frac{T(n)}{n} = \begin{cases} 
\infty & T \in \omega(n), \quad "T > n" \\
\text{constant} > 0 & T \in \Theta(n), \quad "T = n" \\
0 & T \in o(n), \quad "T < n"
\end{cases}
\]

- Linear means in $\Theta(n)$:
  
  $2n + 7, \ 2n + 15\sqrt{n}, \ 10^9n + 3, \ 3n + \log n, \ 2^{\log_2 n + 4}$

- Not linear means not in $\Theta(n)$:
  
  $10^{-9}n^2, \ 10^9\sqrt{n} + 15, \ n^{1.0001}, \ n^{0.9999}, \ n\log n, \ \frac{n}{\log n}, \ 2^n$

- Other runtimes frequently appearing in practice
  
  $\log \quad \text{linear} \quad \log\text{linear} \quad \text{quadratic} \quad \text{cubic} \quad \text{cubical} \quad \text{super-polynomial} \quad \text{exponential} \quad \text{factorial} \quad \text{BAD} \quad \Theta(\log n) \quad \Theta(n) \quad \Theta(n\log n) \quad \Theta(n^2) \quad \Theta(n^3) \quad \Theta(n^{\log n}) \quad \Theta(2^n) \quad \Theta(n!) \quad \Theta(n^n)$
General Asymptotics: $\Theta(f)$, big-Theta-of-$f$

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

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

- Examples:
  - For polynomials, growth rate is the highest order
    \[
    \Theta(2n^2) = n^2 \\
    \Theta(n^2 + n\sqrt{n}) = \Theta(n^2) \\
    \Theta(n^2 + \log^{256} n) = \Theta(n^2) \\
    \Theta(n^2 + n^{1.99}\log^{256} n) = \Theta(n^2)
    \]
The Integration Method

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

\[
\int_{0}^{n} d x \ f(x) 
\]

• 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.)
• **Integer Powers.** Set \( f(x) = x^k \):

\[
\sum_{i=1}^{n} i^k \approx \int_{0}^{n} x^k \, dx
\]

\[
\int_{0}^{n} x^k \, dx = \frac{n^{k+1}}{k + 1}
\]

\[
\frac{n^{k+1}}{k + 1} \in \Theta(n^{k+1})
\]
• 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:

$$((n + 1) \ln(n + 1) - n) \in \Theta(n \ln n)$$
• **Analyzing a recurrence.** $T_1 = 1; T_n = T_{n-1} + n\sqrt{n} - \ln n$
  
  – First, unfold the recurrence:
    
    \[
    T_n = T_{n-1} + n\sqrt{n} - \ln n \\
    T_{n-1} = T_{n-2} + (n - 1)\sqrt{n - 1} - \ln(n - 1) \\
    \vdots \\
    T_3 = T_2 + 3\sqrt{3} - \ln 3 \\
    T_2 = T_1 + 2\sqrt{2} - \ln 2
    \]
  
  – Sum all terms together (note that all $T_{n-1}, \ldots, T_1$ terms cancel out)
    
    \[
    T_n = 1 + 2\sqrt{2} + \cdots + n\sqrt{n} - (\ln 2 + \ln 3 + \cdots + \ln n)
    \]
    
    \[
    = \sum_{i=1}^{n} i\sqrt{i} - \sum_{i=1}^{n} \ln i
    \]
  
  • Why does the 2\text{nd} 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 $T(n) = \sum_{i=1}^{n} i\sqrt{i} - \sum_{i=1}^{n} \ln i$?
    
    $T(n) \in \Theta(n^{5/2})$