(II) Asymptotic Analysis

Sorting Problem

Input: seq of n numbers $a_1, a_2, \ldots, a_n$
Output: a permutation $a'_1, a'_2, \ldots, a'_n$ s.t.
$\ a'_1 \leq a'_2 \leq \ldots \leq a'_n$

Alg: Bubble-Sort

Input: $A = [a_1, \ldots, a_n]$  
for $i := n$ to 2:
  for $j := 1$ to $i-1$:
      swap $A[j]$ and $A[j+1]$

Output: the modified $A$

$A : 53, 12, 15, 0, 4, 97, 22$

$i := n - 1$
$2(n-1) \geq 5(n-1)\ \Rightarrow \ \ \ i := n - 1$
$5(n-2) \gg \text{u.c.}$

$i := 7$

$A : 12, 15, 0, 4, 97, 22$

$j := 1$

$j := 2$

$j := 3$

$j := 4$

$j := 5$

$j := 6$  
$12, 15, 0, 4, 53, 22, 97$
Correctness

Invariant:
at the end of $i$-th iteration, $A[i:j]$ are sorted.

Efficiency?

Running time:

$$f(n) = \sum_{i=n}^{2} 5(i-1) = 5 \cdot \frac{(n-1+1)(n-1)}{2} = \frac{5n(n-1)}{2}$$

At the end of $i$-th iteration:

$A : 53, 12, 15, 0, 4, 97, 22$

$i=7: 12, 15, 0, 4, 53, 22, 97$

$i=6: 12, 0, 4, 15, 22, 53, 97$

$i=5: 0, 4, 12, 15, 22, 53, 97$
Measure Running Time:

Desired properties:

1. A function $f(n)$ scales with $n$. \( \rightarrow \) e.g. len of the array $A$.
2. Platform/hardware indep.
3. Not depend on any single input inst.

Example Algorithm: Ticks-Sort

Input: $A = (a_1, \ldots, a_n)$

If $A = (10, 9, 8, 7, 6, 5)$:

$\rightarrow$ output $(5, 6, 7, 8, 9, 10)$

else:

call Bubble-Sort.
- $f(n)$ vs $g(n)$. $\rightarrow$ not depend on any particular $n$.

- $f(n)$: running time of Alg 1
- $g(n)$: time of Alg 2.

$\rightarrow \infty$

- $f(n)$ covers the worst case input:
  
  Alg: Trick-Sorting 2.
  If $A$ is sorted
  return $A$

  else:
  call Bubble-Sort($A$)

- Faster than bubble sort when $A$ is sorted.
- Doesn't affect worst-case comparison.
How to obtain $f(n)$?

- # of unit operations
- RAM (Random Access Machine)
- Unit operation:
  - Read & Write to any position in the memory
  - Addition, Subtraction, Multiplication, etc.
  - Every numbers involved can be represented in e.g. 64 bits
Big-O notation

Def: Given any function $g(n)$, say $f(n) = O(g(n))$ if

$$\exists C > 0, \ n_0 > 0, \ s.t. \ f(n) \leq C \cdot g(n), \ \forall n \geq n_0.$$

$$\iff \lim_{n \to \infty} \frac{f(n)}{g(n)} \leq C, \ \text{for some constant } C > 0$$

Assume $\lim_{n \to \infty} f(n) = 0$, $\lim_{n \to \infty} g(n) > 0$

Example: $f(n) = 3n^2 + 2n$, $g(n) = n^2 - 10n$.

Claim: $f(n) = O(g(n))$

Proof:
1. Let $C = 100$, $n_0 = 2000$, $f(n) = 3n^2 + 2n \leq C \cdot g(n) = 100n^2 - 1000n$.

$$\Rightarrow C \cdot g(n) - f(n) = 97n^2 - 1002n \geq 0 \ \forall n \geq n_0 = 2000$$

2. $\lim_{n \to \infty} \frac{f(n)}{g(n)} = \lim_{n \to \infty} \frac{3n^2 + 2n}{n^2 - 10n} = \lim_{n \to \infty} \frac{3 + \frac{2}{n}}{1 - \frac{10}{n}} = 3 \leq 4$
\( \Omega \)-notation and \( \Omega \)-notation

- Def: Given functions \( g(n) \), \( f(n) \), say \( f(n) = \Omega(g(n)) \) if

\[
\exists C > 0, \ n_0 > 0, \ \text{s.t.} \ f(n) \geq Cg(n), \ \forall n > n_0
\]

\[
\iff \exists C > 0, \ \text{s.t.} \ \lim_{n \to \infty} \frac{f(n)}{g(n)} \geq C
\]

- Example: \( f(n) = 3n^2 + 2n \), \( g(n) = n^2 - 10n \).

  Claim: \( f(n) = \Omega(g(n)) \)

\[ \text{Pf: Let } C = 100, \ n_0 = 1. \ c \cdot f(n) = 300n^2 + 200n \geq g(n) = n^2 - 10n \text{ for any } n > 1. \]

\[ \text{Pf: } \lim_{n \to \infty} \frac{f(n)}{g(n)} = 3 \geq 1 = C > 0 \]
\( f(n) = O(g(n)) \), \( f(n) = \Omega(g(n)) \)

Def: \( f(n) = \Theta(g(n)) \) iff \( f(n) = O(g(n)) \) and \( \Omega(g(n)) \)

<table>
<thead>
<tr>
<th>( O )</th>
<th>( \Omega )</th>
<th>( \Theta )</th>
<th>( \Omega )</th>
<th>( \omega )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \leq )</td>
<td>( \geq )</td>
<td>( = )</td>
<td>( &lt; )</td>
<td>( &gt; )</td>
</tr>
</tbody>
</table>
\[ \text{Def: Given } \forall \text{ func } g(n), f(n), \text{ say } f(n) = o(g(n)) \text{ if } \lim_{n \to \infty} \frac{f(n)}{g(n)} = 0 \]

E.g. \( f(n) = 3n, \ g(n) = 0.1n^2 \)

\[ \text{Def: Given } \forall \text{ func } g(n), f(n), \text{ say } f(n) = o(g(n)) \text{ if } \lim_{n \to \infty} \frac{g(n)}{f(n)} = 0 \]

E.g. \( f(n) = 0.1n^2, \ g(n) = 3n \)
\( \Omega \| \omega \| \Theta \| \omega \)

\[
\begin{array}{c}
\leq 1 \geq 1 = < >
\end{array}
\]

1. \( f = O(g) \iff g = \Omega(f) \)
2. \( f = \Theta(g) \iff f = O(g) \) and \( f = \Omega(g) \)
3. \( f = o(g) \implies f = \Omega(g) \)
4. \( f = \omega(g) \iff f = \Omega(g) \)

E.g. \( f = n^2 - 10, \ g = 2n^2 \):
- \( f = o(g) \)
- \( f \neq O(g) \)
- \( f = \Omega(g) \)
- \( f \neq \omega(g) \)
D Typical functions

1. Polynomials: $n^c, c \text{ const.}$ e.g. $2n, 3n^2-4, 5n^2-3n^5+2n^4$

2. Logarithmic: $\log_b^c n := (\log_b n)^c, b, c, \text{ const }> 0$
   e.g. $\log_2 n, \log_{10} n, \log_2^2 n, \text{ etc.}$

3. Exponential: $c^n, c, \text{ const } > 1.$ e.g. $2^n, 3^n, e^n$

4. Hybrid of 1&2&3:
   $n\log n, 2^n, \frac{n^2}{\log n}, \frac{n^2}{\log n}, n + \log n, n\log n$
Comparison of different type of functions

1. Logarithmic $<$ Polynomial $<$ Exponential.

$$(\log n)^{100} = o(n^{0.0001})$$

$$n^{1000} = o(2^{0.0001n})$$

2. Polynomial v.s. Polynomials

- Only highest order matter.
  - e.g. $3n^2 + 2n - 1$ v.s. $0.1n^2$

$$3n^2 + 2n - 1 = \Theta(0.1n^2)$$

3. Exponential: $c_1^n$ v.s. $c_2^n$

If $c_1 < c_2 \Rightarrow c_1^n = o(c_2^n)$
4) Hybrid:

\[ \text{e.g. } n \log n \text{ vs. } n^2 \]

Claim: \( n \log n = o \left( n^2 \right) \)

\( n \cdot \log n \) vs. \( n \cdot n \)

\[
\lim_{n \to \infty} \frac{n \log n}{n \cdot n} = \lim_{n \to \infty} \frac{\log n}{n} = 0.
\]

Q: \( n \log n \) vs. \( n^2 \) ?
Conventions

- Ignore lower-order terms
  
  \( f(n) = O(g(n)) \),  \( g(n) = 3n^2 - n + \log n \)

  \( f(n) = n^2 \)

  \( f(n) = O(3n^2 - n + \log n) = O(3n^2) \)

- Ignore constant factors
  
  \( f(n) = O(3n^2) = O(n^2) \)

- \( f(n) = \Omega(g(n)) \) is "tight" \( \iff \ f(n) = \Theta(g(n)) \)

  \( = \Omega(g(n)) \)

- \( O(g(n)) = f(n) \)