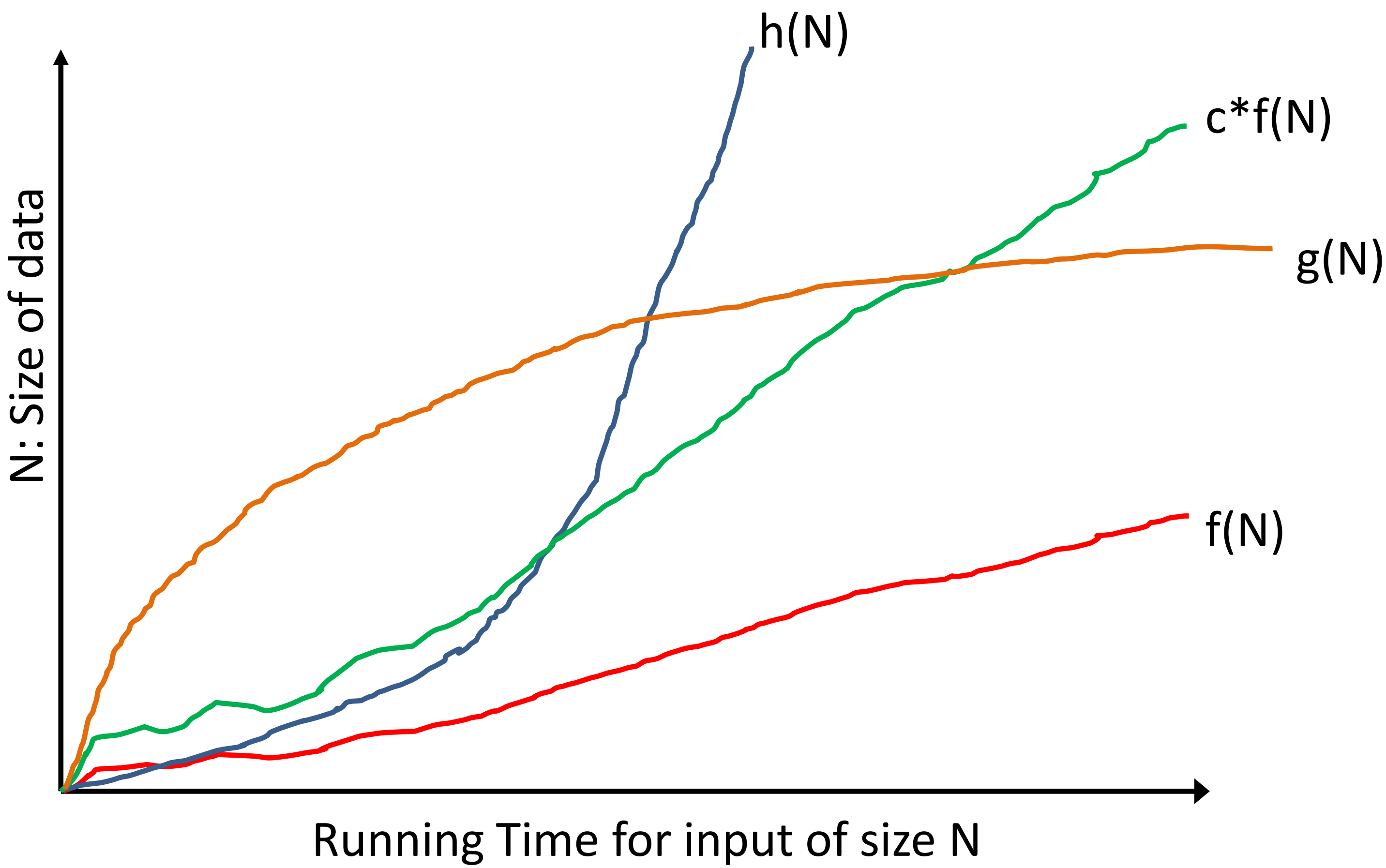
Analysis of Algorithms: Methods and Examples

CSE 2320 – Algorithms and Data Structures
Vassilis Athitsos
University of Texas at Arlington

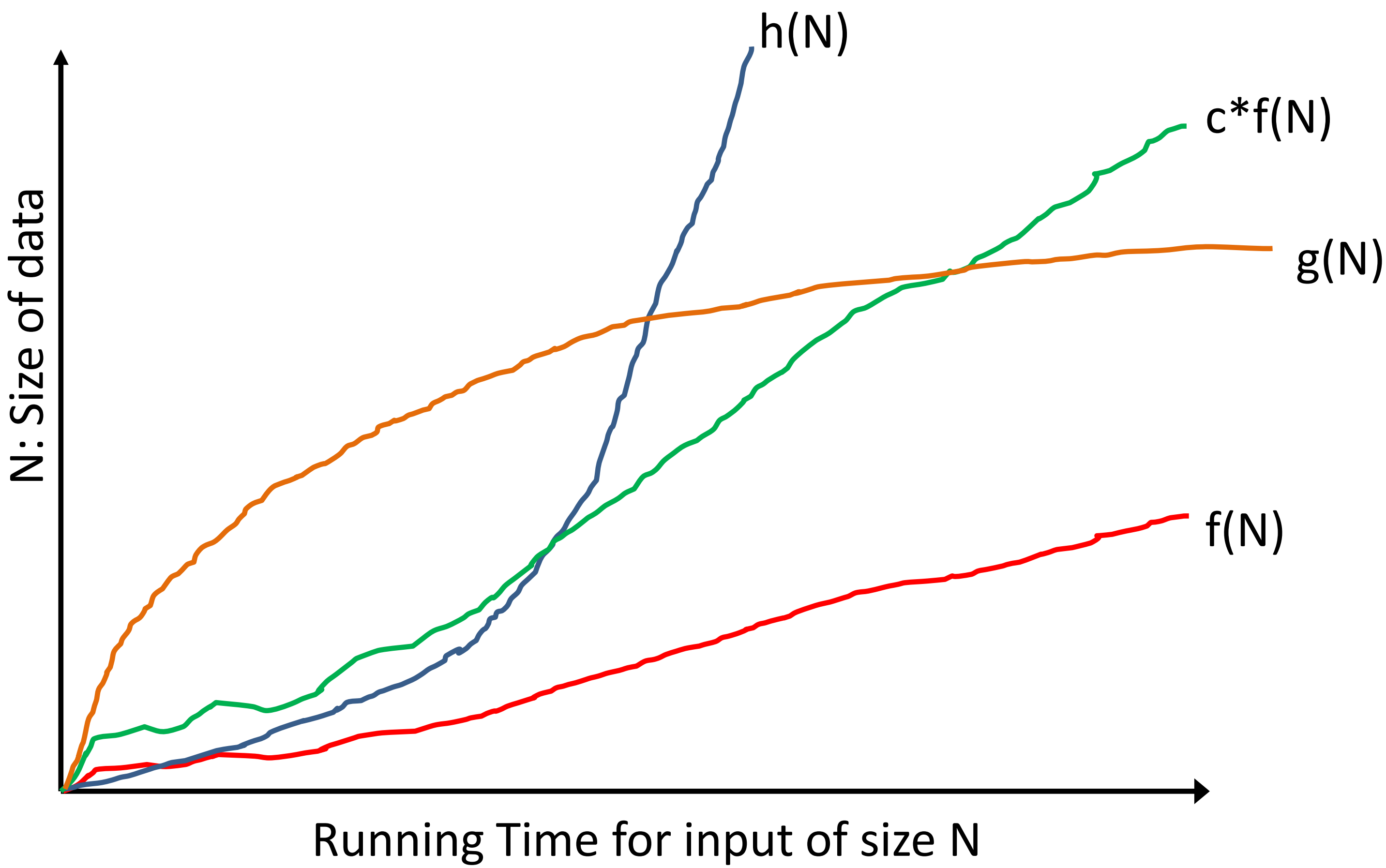
Why Asymptotic Behavior Matters

- Asymptotic behavior: The behavior of a function as the input approaches infinity.



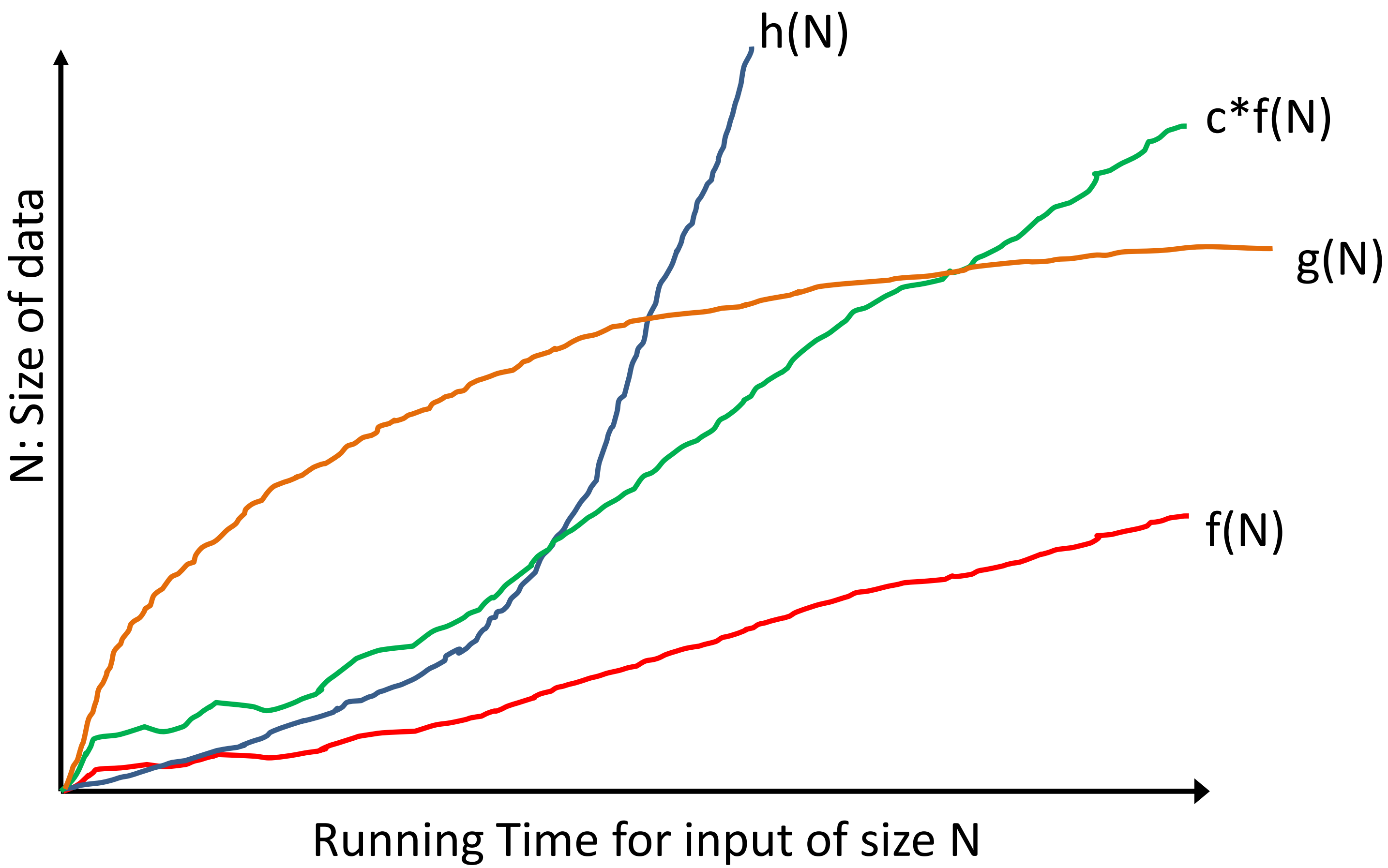
Why Asymptotic Behavior Matters

- Which of these functions is smallest asymptotically?



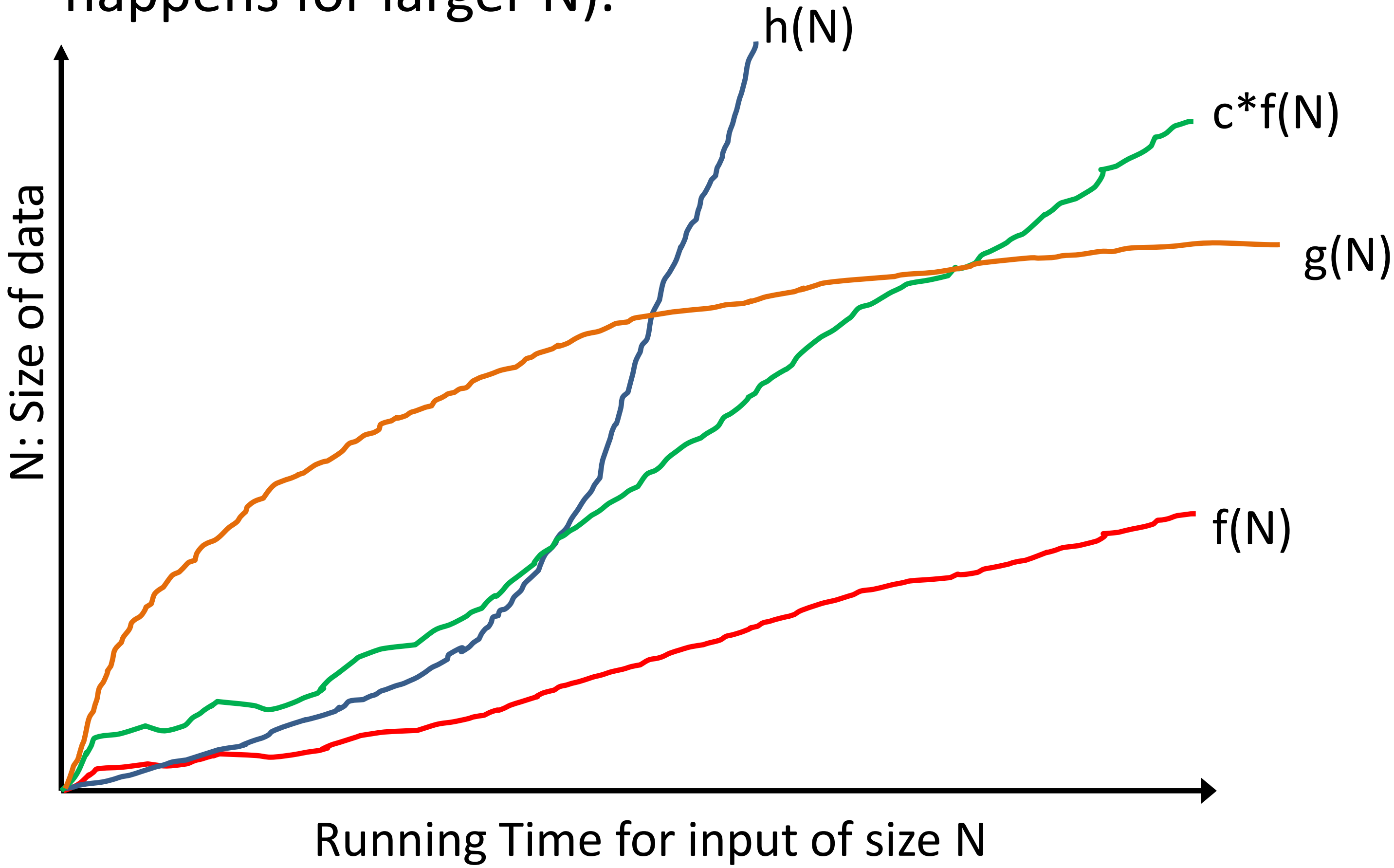
Why Asymptotic Behavior Matters

- Which of these functions is smallest asymptotically?
  - g(N) seems to grow very slowly after a while.



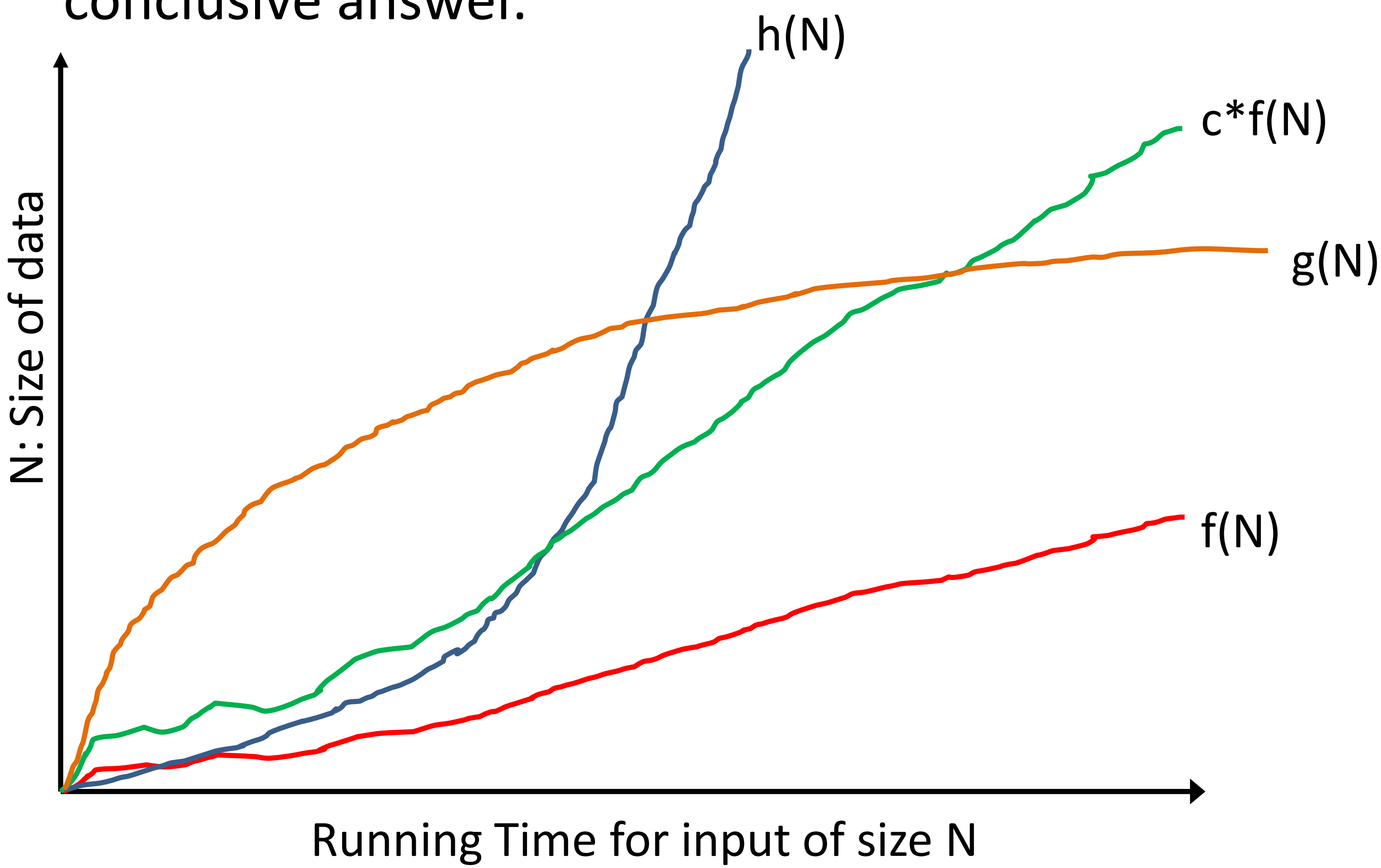
Why Asymptotic Behavior Matters

- Which of these functions is smallest asymptotically?
  - However, the picture is not conclusive (need to see what happens for larger N).



Why Asymptotic Behavior Matters

- Which of these functions is smallest asymptotically?
  - Proving that $g(N) = O(f(N))$ would provide a conclusive answer.



Using Limits

- if $\lim_{N \to \infty} \frac{g(N)}{f(N)}$ is a constant, then g(N) = O(f(N)).
  - "Constant" includes zero, but does NOT include infinity.

- if $\lim_{N \to \infty} \frac{f(N)}{g(N)} = \infty$ then g(N) = O(f(N)).

- if $\lim_{N \to \infty} \frac{f(N)}{g(N)}$ is a constant, then g(N) = Ω(f(N)).
  - Again, "constant" includes zero, but not infinity.

- if $\lim_{N \to \infty} \frac{f(N)}{g(N)}$ is a **non-zero** constant, then g(N) = Θ(f(N)).
  - In this definition, both zero and infinity are excluded.

Using Limits - Comments

- The previous formulas relating limits to big-Oh notation show once again that big-Oh notation ignores:
  - constants
  - behavior for small values of N.
- How do we see that?

Using Limits - Comments

- The previous formulas relating limits to big-Oh notation show once again that big-Oh notation ignores:
  – constants
  – behavior for small values of N.
- How do we see that?
  – In the previous formulas, it is sufficient that the limit is equal to a constant. **The value of the constant does not matter.**
  – In the previous formulas, only **the limit at infinity** matters. This means that we can ignore behavior up to any finite value, if we need to.

Using Limits: An Example

- Show that $\frac{n^5+3n^4+2n^3+n^2+n+12}{5n^3+n+3} = \Theta(???)$.

Using Limits: An Example

- Show that $\frac{n^5+3n^4+2n^3+n^2+n+12}{5n^3+n+3} = \Theta(n^2)$.

- Let $g(n) = \frac{n^5+3n^4+2n^3+n^2+n+12}{5n^3+n+3}$
- Let $f(n) = n^2$.

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \lim_{n\to\infty} \left(\frac{n^5 + 3n^4 + 2n^3 + n^2 + n + 12}{5n^3 + n + 3} \frac{1}{n^2} \right)$$

$$= \lim_{n\to\infty} \left(\frac{n^5 + 3n^4 + 2n^3 + n^2 + n + 12}{5n^5 + n^3 + 3n^2} \right) = \frac{1}{5}$$

Using Limits: An Example

- Show that $\frac{n^5+3n^4+2n^3+n^2+n+12}{5n^3+n+3} = \Theta(n^2)$.

- Let $g(n) = \frac{n^5+3n^4+2n^3+n^2+n+12}{5n^3+n+3}$
- Let $f(n) = n^2$.

- In the previous slide, we showed that $\lim_{n\to\infty} \frac{g(n)}{f(n)} = \frac{1}{5}$
- Therefore, $g(n) = \Theta(f(n))$.

Big-Oh Transitivity

- If $g(N) = O\big(f(N)\big)$ and $f(N) = O(h(N))$, then $g(N) = O(h(N))$.
- How can we prove that?

Big-Oh Transitivity

- If $g(N) = O\big(f(N)\big)$ and $f(N) = O(h(N))$, then $g(N) = O(h(N))$.
- How can we prove that? Using the definition of the big-Oh notation.
- $g(N) < c_0 f(N)$ for all $N > N_0$.
- $f(N) < c_1 h(N)$ for all $N > N_1$.
- Set:
  - $c_2 = c_0 * c_1$
  - $N_2 = \max(N_0, N_1)$
- Then, $g(N) < c_2 h(N)$ for all $N > N_2$.

Big-Oh Hierarchy

- $1 = O(\log(N))$
- $\log(N) = O(N)$
- $N = O(N^2)$
- If $c \geq d \geq 0$, then $N^d = O(N^c)$.
  - Higher-order polynomials always get larger than lower-order polynomials, eventually.
- For any d, if $c > 1$, $N^d = O(c^N)$.
  - Exponential functions always get larger than polynomial functions, eventually.
- You can use these facts in your assignments.
- You can apply transitivity to derive other facts, e.g., that $\log(N) = O(N^2)$.

Using Substitutions

- If $\lim_{x \to \infty} h(x) = \infty$, then:

$$g(x) = O\big(f(x)\big) \Rightarrow g(h(x)) = O(f(h(x))).$$

- How do we use that?
- For example, prove that $\log\big(\sqrt{N}\big) = O(\sqrt{N})$.

Using Substitutions

- If $\lim\limits_{x\to\infty} h(x) = \infty$, then:

$$g(x) = O\big(f(x)\big) \Rightarrow g(h(x)) = O(f(h(x))).$$

- How do we use that?
- For example, prove that $\log\left(\sqrt{N}\right) = O(\sqrt{N})$.
- Use $\mathrm{h}(\mathrm{x}) = \sqrt{N}$. We get:

$$\log(\mathrm{N}) = \mathrm{O}(\mathrm{N}) \Rightarrow \log\left(\sqrt{N}\right) = O\left(\sqrt{N}\right)$$

Summations

- Summations are formulas of the sort: $\sum_{k=0}^{n} f(k)$
- Computing the values of summations can be handy when trying to solve recurrences.
- Oftentimes, establishing upper bounds is sufficient, since we use big-Oh notation.
- If $f(k) \geq 0$, then: $\sum_{k=0}^{n} f(k) \leq \sum_{k=0}^{\infty} f(k)$
- Sometimes, summing to infinity give a more simple formula.

Geometric Series

- A geometric series is a sequence C_k of numbers, such that $C_k = D * C_{k-1}$, where D is a constant.
- How can we express C_1 in terms of C_0?
  - $C_1 = D * C_0$
- How can we express C_2 in terms of C_0?
  - $C_2 = D * C_1 = D^2 * C_0$
- How can we express C_k in terms of C_0?
  - $C_k = D^k * C_0$
- So, to define a geometric series, we just need two parameters: D and C_0.

Summation of Geometric Series

- This is supposed to be a review of material you have seen in Math courses:
- Suppose that $\mathbf{0} < \ \boldsymbol{x} < \mathbf{1}$:
- Finite summations: $\sum_{k=0}^{n} x^k = \dfrac{1 - x^{n+1}}{1 - x}$
- Infinite summations: $\sum_{k=0}^{\infty} x^k = \dfrac{1}{1 - x}$
- Important to note: $\sum_{k=0}^{n} x^k \leq \sum_{k=0}^{\infty} x^k = \frac{1}{1 - x}$

  Therefore, $\sum_{k=0}^{n} x^k = O(1)$. Why?

Summation of Geometric Series

- This is supposed to be a review of material you have seen in Math courses:
- Suppose that $\mathbf{0} < \ \boldsymbol{x} < \mathbf{1}$:
- Finite summations: $\sum_{k=0}^{n} x^k = \dfrac{1 - x^{n+1}}{1 - x}$
- Infinite summations: $\sum_{k=0}^{\infty} x^k = \dfrac{1}{1 - x}$
- Important to note: $\sum_{k=0}^{n} x^k \leq \sum_{k=0}^{\infty} x^k = \frac{1}{1 - x}$
  Therefore, $\sum_{k=0}^{n} x^k = O(1)$. Why?
  – Because $\frac{1}{1 - x}$ is independent of n.

Summation of Geometric Series

- Suppose that $x > 1$: The formula for finite summations is the same, and can be rewritten as:
- $\sum_{k=0}^{n} x^k = \frac{x^{n+1}-1}{x-1}$
- This can be a handy formula in solving recurrences:
- For example:

$$1 + 5 + 5^2 + 5^3 + \ldots + 5^n = \frac{5^{n+1} - 1}{5 - 1} = O(5^n)$$

Harmonic Series

- $H_N = \sum_{k=1}^{N} \frac{1}{k}$
- $\ln(N) \leq H_N \leq \ln(N) + 1$
- The above formula shows that the harmonic series can be easily approximated by the natural logarithm.
- It follows that $H_N = O(\log(N))$. Why?
- $\ln(N) = \log_e N \; = \; \frac{\log_2 N}{\log_2 e} = \frac{1}{\log_2 e} \log_2 N = O(\log(N))$
- $H_N = O(\ln(n)) = O(\log(N))$

Approximation by Integrals

- Suppose that f(x) is a monotonically increasing function:
  - This means that $x \leq y \Rightarrow f(x) \leq f(y)$.
- Then, we can approximate summation $\sum_{k=m}^{n} f(k)$ using integral $\int_{m}^{n+1} f(x)dx$.
- Why? Because $f(k) \leq \int_{k}^{k+1} f(x)dx$.
- Why? $\int_{k}^{k+1} f(x)dx$ is the average value of $f(x)$ in the interval $[k, k+1]$.
- For every x in the interval $[k, k+1]$, $x \geq \ k$.Since $f(x)$ is increasing, if $x \ \geq \ k$ then $f(x) \geq \ f(k)$.

Solving Recurrences: Example 1

- Suppose that we have an algorithm that at each step:
  - takes $O(N^2)$ time to go over N items.
  - eliminates one item and then calls itself with the remaining data.
- How do we write this recurrence?

Solving Recurrences: Example 1

- Suppose that we have an algorithm that at each step:
  - takes $O(N^2)$ time to go over N items.
  - eliminates one item and then calls itself with the remaining data.
- How do we write this recurrence?
- $g(N) = g(N-1) + N^2$

$$= g(N-2) + (N-1)2 + N^2$$

$$= g(N-3) + (N-2)2 + (N-1)2 + N^2$$

$$\ldots$$

$$= 1^2 + 2^2 + \ldots + N^2$$

$$= \sum_{k=1}^{N} k^2.$$ How do we approximate that?

Solving Recurrences: Example 1

- We approximate $\sum_{k=1}^{N} \boldsymbol{k^2}$ using an integral:

- Clearly, $f(x) = x^2$ is a monotonically increasing function.

- So, $\sum_{k=1}^{N} k^2 \leq \int_{1}^{N+1} x^2 dx = \frac{(N+1)^3 - 1^3}{3}$

$$= \frac{N^3 + 2N^2 + 2N + 1 - 1}{3} = \Theta(N^3)$$

Solving Recurrences: Example 2

- Suppose that we have an algorithm that at each step:
  - takes O(log(N)) time to go over N items.
  - eliminates one item and then calls itself with the remaining data.
- How do we write this recurrence?

Solving Recurrences: Example 2

- Suppose that we have an algorithm that at each step:
  - takes O(log(N)) time to go over N items.
  - eliminates one item and then calls itself with the remaining data.
- How do we write this recurrence?
- $g(N) = g(N-1) + \log(N)$

$$= g(N-2) + \log(N-1) + \log(N)$$

$$= g(N-3) + \log(N-2) + \log(N-1) + \log(N)$$

…

$$= \log(1) + \log(2) + \ldots + \log(N)$$

$= \sum_{k=1}^{N} log(k)$. How do we compute that?

Solving Recurrences: Example 2

- We process $\sum_{\boldsymbol{k=1}}^{\boldsymbol{N}} \boldsymbol{log}(\boldsymbol{k})$ using the fact that:
  $\log(a) + \log(b) = \log(ab)$

- $$\begin{aligned}\sum_{\mathrm{k=1}}^{\mathrm{N}} \log(\mathrm{k}) &= \log(1) + \log(2) + \ldots + \log(\mathrm{N}) \\ &= \log(N!) \\ &\cong \log\left(\left(\frac{N}{e}\right)^N\right) \\ &= N \log\left(\frac{N}{e}\right) \\ &= N \log(N) - N \log(e) = O(N \log(N))\end{aligned}$$

Solving Recurrences: Example 3

- Suppose that we have an algorithm that at each step:
  - takes O(1) time to go over N items.
  - calls itself 3 times on data of size N-1.
  - takes O(1) time to combine the results.
- How do we write this recurrence?

Solving Recurrences: Example 3

- Suppose that we have an algorithm that at each step:
  - takes O(1) time to go over N items.
  - calls itself 3 times on data of size N-1.
  - takes O(1) time to combine the results.
- How do we write this recurrence?
- $g(N) = 3g(N-1) + 1$

$$= 3^2 g(N-2) + 3 + 1$$

$$= 3^3 g(N-3) + 3^2 + 3 + 1$$

…

$$= 3^{N-1} g(1) + 3\underbrace{^{N-2} + 3^{N-3} + 3^{N-4} + \cdots}_{\text{finite summation}} + 1$$

Note: $g(1)$ is just a constant

Solving Recurrences: Example 3

- Suppose that we have an algorithm that at each step:
  - takes O(1) time to go over N items.
  - calls itself 3 times on data of size N-1.
  - takes O(1) time to combine the results.
- How do we write this recurrence?
- $g(N) = 3g(N-1) + 1$

$$= 3^2 g(N-2) + 3 + 1$$

$$= 3^3 g(N-3) + 3^2 + 3 + 1$$

$$\ldots$$

$$= 3^{N-1} g(1) + 3^{N-2} + 3^{N-3} + 3^{N-4} + \cdots + 1$$

$$= O(3^N) + O(3^N) = O(3^N)$$

Solving Recurrences: Example 4

- Suppose that we have an algorithm that at each step:
  - calls itself N times on data of size N/2.
  - takes O(1) time to combine the results.
- How do we write this recurrence?

Solving Recurrences: Example 4

- Suppose that we have an algorithm that at each step:
  - calls itself N times on data of size N/2.
  - takes O(1) time to combine the results.
- How do we write this recurrence? Let $\mathrm{n} = \log N$.
- $g(2^n) \; = 2^n g(2^{n-1}) + 1$

$$= 2^n \, 2^{n-1} g(N-2) + 2^n + 1$$

$$= 2^n \, 2^{n-1} \, 2^{n-2} g(N-3) + 2^n \; 2^{n-1} + 2^n + 1$$

$$= \left(\prod_{k=n-2}^{n} 2^n \right) g(N-3) + 1 + \sum_{k=n-1}^{n} \prod_{i=k}^{n} 2^n$$

Solving Recurrences: Example 4

- Suppose that we have an algorithm that at each step:
  - calls itself N times on data of size N/2.
  - takes O(1) time to combine the results.
- How do we write this recurrence? Let $\mathrm{n} = \log N$.
- $g(2^n) = 2^n g(2^{n-1}) + 1$

$$= 2^n\, 2^{n-1} g(N-2) + 2^n + 1$$

$$= 2^n\, 2^{n-1}\, 2^{n-2} g(N-3) + 2^n\ 2^{n-1} + 2^n + 1$$

$$= \left(\prod_{k=n-3}^{n} 2^n\right) g(N-4) + 1 + \sum_{k=n-2}^{n} \prod_{i=k}^{n} 2^n$$

Solving Recurrences: Example 4

- Suppose that we have an algorithm that at each step:
  - calls itself N times on data of size N/2.
  - takes O(1) time to combine the results.
- How do we write this recurrence? Let $\mathrm{n} = \log N$.
- $g(2^n) \ = 2^n g(2^{n-1}) + 1$

$$= 2^n\, 2^{n-1} g(N-2) + 2^n + 1$$

$$= 2^n\, 2^{n-1}\, 2^{n-2} g(N-3) + 2^n\ 2^{n-1} + 2^n + 1$$

$$= \ \left(\prod_{k=2}^{n} 2^n\right) g(1) + 1 + \sum_{k=3}^{n} \prod_{i=k}^{n} 2^n$$

Solving Recurrences: Example 4

$$\left(\prod_{k=2}^{n} 2^{n}\right) = 2^{n}2^{n-1}2^{n-2} \ \ldots 2^{2}2^{1}$$

$$= 2^{(n+n-1+n-2+\cdots+1)}$$

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

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

$$= N^{\frac{\log(N)+1}{2}}$$

Substituting N for 2^{n}

$$= O(N^{\frac{\log(N)}{2}})$$

Solving Recurrences: Example 4

- Let $\mathrm{X} = (\prod_{k=2}^{n} 2^n)$ (which we have just computed).

$$\sum_{k=3}^{n} \prod_{i=k}^{n} 2^n < X + \frac{X}{2} + \frac{X}{4} + \ \dots \ \Rightarrow$$

$\sum_{k=3}^{n} \prod_{i=k}^{n} 2^n < 2X \Rightarrow$ (taking previous slide into account)

$$\sum_{k=3}^{n} \prod_{i=k}^{n} 2^n = O(N^{\frac{\log(N)}{2}})$$

Solving Recurrences: Example 4

- Based on the previous two slides, we can conclude that the solution of: $g(N) = Ng(N/2) + 1$ is that:

$$g(N) = O(N^{\frac{\log(N)}{2}})$$

Big-Oh Notation: Example Problem

- Is $N = O(\sin(N)\,N^2)$?
- Answer:

Big-Oh Notation: Example Problem

- Is $N = O(\sin(N)\, N^2)$?
- Answer: no!
- Why? $\sin(N)$ fluctuates forever between -1 and 1.
- As a result, $\sin(N)\, N^2$ fluctuates forever between negative and positive values.
- Therefore, for every possible $c_0 > 0$ and N_0, we can always find an $N > N_0$ such that:

$$N > c_0 \sin(N) N^2$$