Algorithms

Algorithm: what is it?
Algorithms

Algorithm: what is it?

Some representative problems:

- Interval Scheduling

Different variants, e.g.:

1. select largest # non-overlapping intervals
2. weighted intervals: select non-overlapping intervals with largest sum of weights
Algorithms

Algorithm: what is it?

Some representative problems:
- Interval Scheduling
- Bipartite Matching

\[ \text{max # couples could be smaller than } \min(x, y) \]
Algorithms

Algorithm: what is it?

Some representative problems:
- Interval Scheduling
- Bipartite Matching
- Independent Set

Find largest k vertices with no edges between them (indep. set) NP-complete
Algorithms

Algorithm: what is it?

Some representative problems:
- Interval Scheduling
- Bipartite Matching
- Independent Set
- Area of a Polygon
How to decide which algorithm is better?

Search problem

Input: a sequence of n numbers (in an array A) and a number x
Output: YES, if A contains x, NO otherwise

$O(n)$
Algorithms

How to decide which algorithm is better?

Search problem

Input: a sequence of n numbers (in an array A) and a number x
Output: YES, if A contains x, NO otherwise

What if A is already sorted? $O(\log n)$
Running Time

\( O(n) \) - running time of the linear search

\( O(\log n) \) - running time of the binary search

**Def**: Big-Oh (asymptotic upper bound)

\[ f(n) = O(g(n)) \text{ if there exists a constant } c > 0 \text{ and a constant } n_0 \text{ such that for every } n \geq n_0 \text{ we have } f(n) \leq c g(n) \]

Examples:

\( n, \ n^3, \ \log n, \ 2^n, \ 7n^2 + n^3/3, \ 1, \ 1 + \log n, \ n \log n, \ n + \log n, \ \frac{n^3}{\log n}, \ \sqrt{n} \log n, \ \frac{n^3}{\log n} \)
Running Time

Def: Big-Oh (asymptotic upper bound)

\[ f(n) = O(g(n)) \text{ if there exists a constant } c > 0 \text{ and a constant } n_0 \]

\[ \text{such that for every } n \geq n_0 \text{ we have } f(n) \leq c \cdot g(n) \]

Example: Prove that \( n = O(n^3) \)

Want to find \( c > 0 \) and \( n_0 \) st.

\[ \forall n \geq n_0 : \quad f(n) \leq c \cdot g(n) \]

In our case:

\[ n \leq c \cdot n^3 \quad \forall n \geq n_0 \]

Take: \( c = 1 \quad n_0 = 1 \)
Running Time

Def: **Big-Oh (asymptotic upper bound)**

\[ f(n) = O(g(n)) \text{ if there exists a constant } c > 0 \text{ and a constant } n_0 \text{ such that for every } n \geq n_0 \text{ we have } f(n) \leq c \cdot g(n) \]

Example: Prove that \( n^3 = O(7n^2 + n^3/3) \)

We want to find \( c > 0 \) and \( n_0 \) such that

\[ \forall n \geq n_0 : n^3 \leq c \cdot \left( 7n^2 + \frac{n^3}{3} \right) \quad \text{take: } c = 3, n_0 = 1 \]

\[ 21n^2 + n^3 \]
Running Time

Def: Big-Oh (asymptotic upper bound)

\[ f(n) = O(g(n)) \] if there exists a constant \( c > 0 \) and a constant \( n_0 \) such that for every \( n \geq n_0 \) we have \( f(n) \leq c \cdot g(n) \)

Example: Prove that \( n^3 = O(n^3/3 - 7n^2) \)

\[
\begin{align*}
\text{find } & \quad c, n_0 \text{ s.t. } \\
\forall n \geq n_0 : & \quad n^3 \leq c \cdot (n^3/3 - 7n^2) \\
\text{take: } & \quad c = 9 \\
\Rightarrow & \quad n^3 \leq 9 \cdot (n^3/3 - 7n^2) = 3n^3 - 63n^2 \\
63n^2 & \leq 2n^3 \\
63 & \leq 2n \\
\text{choose: } & \quad n_0 = \frac{63}{2}
\end{align*}
\]
Running Time

Def: **Big-Oh (asymptotic upper bound)**

\[ f(n) = O(g(n)) \text{ if there exists a constant } c > 0 \text{ and a constant } n_0 \text{ such that for every } n \geq n_0 \text{ we have } f(n) \leq c g(n) \]

Example: Prove that \( \log_{10} n = O(\log n) \)

And that \( \log n = O(\log_{10} n) \)

we mean:
\[ \log n = \log_{10} n \]

\[ \log_{a} b = \frac{\log b}{\log a} \]

\[ \log_{10} n = \frac{\log n}{\log_{10}} \]

Choose: \( c = \frac{\log n}{\log_{10}} \)

Take \( c = 1 \text{ and } n_0 = 1 \)
**Running Time**

**Def**: **Big-Oh (asymptotic upper bound)**

\[ f(n) = O(g(n)) \] if there exists a constant \( c > 0 \) and a constant \( n_0 \) such that for every \( n \geq n_0 \) we have \( f(n) \leq c \cdot g(n) \)

**Example**: what about \( 3^n \) and \( 2^n \)

\[ 3^n \neq O(2^n) \]

We want to show that no \( c, n_0 \) pair exists.

Suppose there were \( n_0, c \):

We know that \( \forall n \geq n_0 : 3^n \leq c \cdot 2^n \)

\[ \left( \frac{3}{2} \right)^n \leq c \]

Thus \( c \) is unbounded.

This contradicts \( c \) being a constant.

\[ 2^n = O(3^n) \]

Take \( c = 1 \) and \( n_0 = 1 \)

\( \forall n \geq n_0 : 2^n \leq 3^n \)

\[ 1 \leq \left( \frac{3}{2} \right)^n \quad \checkmark \]
Running Time

\( O(n) \) - running time of the linear search

\( O(\log n) \) - running time of the binary search

Def: **Big-Omega** (asymptotic lower bound)

\( f(n) = \Omega(g(n)) \) if there exists a constant \( c > 0 \) and a constant \( n_0 \) such that for every \( n \geq n_0 \) we have \( f(n) \geq c \cdot g(n) \)

Examples:

\( n, \ n^3, \ \log n, \ 2^n, \ 7n^2 + n^3/3, \ 1, \ 1 + \log n, \ n \log n, \ n + \log n \)
Running Time

$O(n)$ - running time of the linear search

$O(\log n)$ - running time of the binary search

Def: **Theta (asymptotically tight bound)**

$f(n) = \Theta(g(n))$ if there exists constants $c_1, c_2 > 0$ and a constant $n_0$ such that for every $n \geq n_0$ we have $c_1 g(n) \leq f(n) \leq c_2 g(n)$

Examples:

$n, \ n^3, \ \log n, \ 2^n, \ 7n^2 + n^3/3, \ 1, \ 1 + \log n, \ n \log n, \ n + \log n$
A survey of common running times

**Linear in n**

1. for $i=1$ to $n$ do
2. something

Also linear:

1. for $i=1$ to $n$ do
2. something
3. for $i=1$ to $n$ do
4. something else

if 2 data sets:

- n elements
- k elements

linear: $O(n+k)$

for a graph with $n$ vertices and $m$ edges:

- linear: $O(n+m)$
Example (linear time):

Given is a point $A = (a_x, a_y)$ and $n$ points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ specifying a polygon. Decide if $A$ lies inside or outside the polygon.

1. Take a line to a point outside the polygon.
2. Compute $k$ intersections with the polygon (loop through the edges).
3. If even then OUT, otherwise IN.
A survey of common running times

Example (linear time):

Given are \( n \) points \((x_1,y_1), (x_2,y_2), \ldots, (x_n,y_n)\) specifying a polygon. Compute the area of the polygon.

\[ \text{triangulation: } O(n^2) \]
A survey of common running times

Example (linear time):

Given are \( n \) points \((x_1,y_1), (x_2,y_2), \ldots, (x_n,y_n)\) specifying a polygon. Compute the area of the polygon.
Example (linear time):

Given are $n$ points $(x_1,y_1), (x_2,y_2), \ldots, (x_n,y_n)$ specifying a polygon. Compute the area of the polygon.

```plaintext
area = 0
for i = 1 to n:
    // processing $x_i, y_i$ to $x_{i+1}, y_{i+1}$
    trapezarea = $(x_2-x_1) \cdot \frac{y_1+y_2}{2}$
    area += trapezarea
return |area|
```
A survey of common running times

\[ O(n \log n) \]

1. for \( i = 1 \) to \( n \) do
2. \( \quad \) for \( j = 1 \) to \( \log(n) \) do
3. \( \quad \) something

Or:

1. for \( i = 1 \) to \( n \) do
2. \( \quad j = n \)
3. \( \quad \) while \( j > 1 \) do
4. \( \quad \) something
5. \( \quad j = j/2 \)

\[ 0(n) \]

for \( i = 1 \) to \( \sqrt{n} \):
  for \( j = 1 \) to \( \sqrt{n} \):
    something \( \, / \, O(1) \)
A survey of common running times

Quadratic

1. for i=1 to n do
2. for j=1 to n do
3. something
A survey of common running times

Cubic
A survey of common running times

\[ O(n^c) \] - polynomial (if \( k \) is a constant)

Remark: if \( k \) is a part of the input

\[ O(n^k) \]

Brute force running time:

\[ O(n^k) \]

\[ \to \text{ not polynomial} \]

Decision version of the independent set problem:

given a graph and \( k > 0 \)

\( \exists \) an independent set of size \( k \)

\( k = 3 \)
A survey of common running times

Exponential, e.g., $O(2^n)$