CSE 421: Introduction to Algorithms

Induction - Graphs

Shayan Oveis Gharan
Efficient = Polynomial Time

An algorithm runs in polynomial time if $T(n) = O(n^d)$ for some constant $d$ independent of the input size $n$.

Why Polynomial time?

If problem size grows by at most a constant factor then so does the running time

- E.g. $T(2N) \leq c(2N)^k \leq 2^k(cN^k)$
- Polynomial-time is exactly the set of running times that have this property

Typical running times are small degree polynomials, mostly less than $N^3$, at worst $N^6$, not $N^{100}$
Why it matters?

- #atoms in universe < $2^{240}$
- Life of the universe < $2^{54}$ seconds
- A CPU does < $2^{30}$ operations a second

If every atom is a CPU, a $2^n$ time ALG cannot solve $n=350$ if we start at Big-Bang.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$n$</th>
<th>$n \log_2 n$</th>
<th>$n^2$</th>
<th>$n^3$</th>
<th>$1.5^n$</th>
<th>$2^n$</th>
<th>$n!$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 10$</td>
<td></td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>4 sec</td>
</tr>
<tr>
<td>$n = 30$</td>
<td></td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>18 min</td>
<td>$10^{25}$ years</td>
</tr>
<tr>
<td>$n = 50$</td>
<td></td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>11 min</td>
<td>36 years</td>
<td>very long</td>
</tr>
<tr>
<td>$n = 100$</td>
<td></td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>1 sec</td>
<td>12,892 years</td>
<td>$10^{17}$ years</td>
<td>very long</td>
</tr>
<tr>
<td>$n = 1,000$</td>
<td></td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>18 min</td>
<td>very long</td>
<td>very long</td>
<td>very long</td>
</tr>
<tr>
<td>$n = 10,000$</td>
<td></td>
<td>&lt; 1 sec</td>
<td>&lt; 1 sec</td>
<td>2 min</td>
<td>very long</td>
<td>very long</td>
<td>very long</td>
</tr>
<tr>
<td>$n = 100,000$</td>
<td></td>
<td>&lt; 1 sec</td>
<td>2 sec</td>
<td>12 days</td>
<td>very long</td>
<td>very long</td>
<td>very long</td>
</tr>
<tr>
<td>$n = 1,000,000$</td>
<td></td>
<td>1 sec</td>
<td>20 sec</td>
<td>31,710 years</td>
<td>very long</td>
<td>very long</td>
<td>very long</td>
</tr>
</tbody>
</table>

Not only get very big, but do so *abruptly*, which likely yields erratic performance on small instances.
Why “Polynomial”?

Point is not that \( n^{2000} \) is a practical bound, or that the differences among \( n \) and \( 2n \) and \( n^2 \) are negligible. Rather, simple theoretical tools may not easily capture such differences, whereas exponentials are qualitatively different from polynomials, so more amenable to theoretical analysis.

- “My problem is in P” is a starting point for a more detailed analysis
- “My problem is not in P” may suggest that you need to shift to a more tractable variant
Graphs
Graphs
Undirected Graphs $G=(V,E)$
Graphs don’t Live in Flat Land

Geometrical drawing is mentally convenient, but mathematically irrelevant:

4 drawings of a single graph:
Directed Graphs

- Multi edge
- Self loop

Node connections:
- 1 to 2, 3, 10
- 2 to 3, 10, 11
- 3 to 4, 5
- 4 to 5
- 5 to 6
- 6 to 7
- 7 to 8
- 8 to 9
- 9 to 10
- 10 to 11
- 11 to 12, 13
- 12 to 13
Terminology

• **Degree of a vertex:** # edges that touch that vertex

\[
\text{deg}(6) = 3
\]

• **Connected:** Graph is connected if there is a path between every two vertices

• **Connected component:** Maximal set of connected vertices
Terminology (cont’d)

• **Path**: A sequence of distinct vertices s.t. each vertex is connected to the next vertex with an edge.

• **Cycle**: Path of length > 2 that has the same start and end.

• **Tree**: A connected graph with no cycles.
Degree Sum

Claim: In any undirected graph, the number of edges is equal to \((1/2) \sum_{\text{vertex } v} \deg(v)\)

Pf: \(\sum_{\text{vertex } v} \deg(v)\) counts every edge of the graph exactly twice; once from each end of the edge.

\(|E|=8\)

\[\sum_{\text{vertex } v} \deg(v) = 2 + 2 + 1 + 1 + 3 + 2 + 3 + 2 = 16\]
**Odd Degree Vertices**

**Claim**: In any undirected graph, the number of odd degree vertices is even.

**Pf**: In previous claim we showed sum of all vertex degrees is even. So there must be even number of odd degree vertices, because sum of odd number of odd numbers is odd.

4 odd degree vertices 3, 4, 5, 6

\[ \text{even} = \deg(v_1) + \deg(v_2) + \deg(v_{\ldots}) + \deg(v_n) \]

Even degree vertices: even degree + even degree = even
Odd degree vertices: odd degree + odd degree = even

Adding any odd number changes the parity. If we have odd # odd degree vertices the sum is odd.
Claim: If G has no cycle, then it has a vertex of degree \( \leq 1 \) 
(So, every tree has a leaf)

Pf: (By contradiction)
Suppose every vertex has degree \( \geq 2 \).
Start from a vertex \( v_1 \) and follow a path, \( v_1, \ldots, v_i \) when we are at \( v_i \) we choose the next vertex to be different from \( v_i \neq v_i \).
We can do so because \( \deg(v_i) \geq 2 \).
The first time that we see a repeated vertex \( (v_j = v_i) \) we get a cycle.
We always get a repeated vertex because \( G \) has finitely many vertices.
Trees and Induction

Claim: Show that every tree with \( n \) vertices has \( n-1 \) edges.

Pf: By induction.

Base Case: \( n=1 \), the tree has no edge

IH: Suppose every tree with \( n-1 \) vertices has \( n-2 \) edges

IS: Let \( T \) be a tree with \( n \) vertices.
So, \( T \) has a vertex \( v \) of degree 1.
Remove \( v \) and the neighboring edge, and let \( T' \) be the new graph.
We claim \( T' \) is a tree: It has no cycle, and it must be connected.
So, \( T' \) has \( n-2 \) edges and \( T \) has \( n-1 \) edges.
Induction

Induction in 311:
Prove $1 + 2 + \cdots + n = n(n + 1)/2$

Induction in 421:
Prove all trees with $n$ vertices have $n - 1$ edges
Let $G = (V, E)$ be a graph with $n = |V|$ vertices and $m = |E|$ edges.

**Claim:** $0 \leq m \leq \binom{n}{2} = \frac{n(n-1)}{2} = O(n^2)$

**Pf:** Since every edge connects two distinct vertices (i.e., $G$ has no loops) and no two edges connect the same pair of vertices (i.e., $G$ has no multi-edges) It has at most $\binom{n}{2}$ edges.