Intractable Problems

- Clique
- Independent Set
- Vertex Cover
- Set Cover
- Set Packing
- Satisfiability Problem
- Hamiltonian Cycle and Path
- Traveling Salesman Problem
- Graph Coloring
- Circuit Satisfiability
- Knapsack
- Subset Sum
- Prime and Factor
- Partition

Imdad ullah Khan

Efficiently Solvable Problems

So far we dealt with problems like sort n numbers, find connected components, find shortest $s \to t$ path, find MST, find the best alignment, find matching

- We devised **efficient** algorithms for them

  Efficient in the sense that the search space generally is exponential

  ▷ Brute force algorithm would take exponential time

  - Only one ordering out of $n!$ permutation is sorted
  - Out of the possible n^{n-2} spanning trees (for K_n) only one is a MST
  - There could be exponentially many paths from s to t
  - Exponentially many alignments between two sequences

- Used greedy algorithms, dynamic programming to avoid exponential time

  ▷ Divide and Conquer typically is used to reduce already polynomial time

Efficiently Solvable Problems

Efficiently Solvable Problem

$\exists$ an $O(n^k)$ worst case time algorithm for instances of size n, constant k

- Does not mean that n^{70} is OK, or no difference between n^2 and n^3
- We try to improve the polynomial's degree for polynomial time algorithms
  - Such as divide and conquer or design better data structures

Hard (Intractable) Problems

Efficiently Solvable Problem

$\exists$ an $O(n^k)$ worst case time algorithm for instances of size n, constant k

- Now we study negative results
- Characterize problems for which we don't have good news
- Cannot say they are not efficiently solvable (just don't know yet)
- We might need to focus on approximation or special cases

Hard (Intractable) Problem

- No known $O(n^k)$ algorithm
- Exponential time is sufficient $O(n^n), O(n!), O(k^n)$

We establish that these "hard problems" are in some sense are equivalent

Hard Problems: Genres of Problems

We discuss six basic genres of hard problems and paradigmatic examples

- Packing problems: SET-PACKING, INDEPENDENT-SET
- Covering problems: SET-COVER, VERTEX-COVER
- Constraint satisfaction problems: SAT, 3-SAT
- Sequencing problems: HAMILTONIAN-CYCLE, TSP
- Numerical problems: SUBSET-SUM, KNAPSACK
- Partitioning problems: 3D-MATCHING, 3-COLORING
- Number Theory problems: FACTOR, DISCRETE-LOG