Lecture 28: Reductions and Decomposition

10/31/2020

Topological Sorting

Topological Sort

- Suppose we have tasks 0 through 7, where an arrow from v to w indicates that va must happen before w
  - What algorithm do we use to find a valid ordering for these tasks?

Solution

- Perform a DFS traversal from every vertex with indegree 0, NOT clearing markings in between traversals
  - Record DFS postorder in a list
  - Topological ordering is given by the reverse of that list
  - This algorithm fails if there is a cycle (There is no such thing as a topological sort with cycles)
- Another better topological sorting algorithm:
  - Run DFS from an arbitrary vertex
  - If not all marked, pick an unmarked vertex and do it again

Topological Sort

- The reason it's called a topological sort: Can think of this process as sorting our nodes so they appear in an order consistent with edges
  - When nodes are sorted in diagram, arrows all point rightwards

Depth First Search

- Be aware, that when people say "Depth First Search", they sometimes mean with restarts, and they sometimes mean without
- For example, DepthFirstPaths did not restart but Topological Sort restarts from every vertex with indegree 0

Directed Acyclic Graphs

- A topological sort only exists if the graph is a directed acyclic graph (DAG)

Shortest Paths on DAGs

- Dijkstra's can fail with negative edges

Challenge

- Try to come up with an algorithm for shortest paths on a DAG that works even if there are negative edges
- One simple idea: Visit vertices in topological order
  - On each visit, relax all outgoing edges
  - Each vertex is visited only when all possible info about it has been used!

The DAG SPT Algorithm: Relax in Topological Order

- We have to visit all the vertices in topological order, relaxing all edges as we go
  - Runtime is O(V + E)

Longest Paths

The Longest Paths Problem

- Consider the problem of finding the longest path tree (LPT) from s to every other vertex. The path must be simple (no cycles!)
- Some surprising facts
  - The best known algorithm is exponential (extremely bad)
  - Perhaps the most important unsolved problem in mathematics

The Longest Paths Problem on DAGs

- Difficult challenge
  - Solve the LPT problem on a directed acyclic graph
  - Algorithm must be O(E + V) runtime
- DAG LPT solution for solution G:
  - Form a new copy of the graph G' with signs of all edge weights flipped
  - Run DAGSPT on G' yielding result X
  - Flip signs of all values in X.distTo (X.edgeTo is already correct)

Reduction (170 Preview)

DAG Longest Paths and Reduction

- The problem solving we just used probably felt a little different than usual
  - Given a graph G, we created a new graph G' and fed it to a related (but different) algorithm, and then interpreted the result
- This process is known as reduction
  - Since DAG-SPT can be used to solve DAG-LPT, we say that "DAG-LPT reduces to DAG-SPT"

Reduction Analogy

- As a real-world analog, suppose we want to climb a hill. There are many ways to do this:
  - "Climbing a hill" reduces to "riding a ski lift"

Reduction Definition (Informal)

- "If any subroutine for task Q can be used to solve P, we say P reduces to Q"
- Can also define the idea formally, but beyond scope of class

Reduction is More than Sign Flipping

- Let's see a richer example

The Independent Set Problem

- An independent set is a set of vertices in which no two vertices are adjacent
- The Independent-set Problem:
  - Does there exist an independent set of size k?
  - i.e. color k vertices red, such that none touch

THe 3SAT Problem

- 3SAT: Given a boolean formula, does there exist a truth value for boolean variables that obeys a set of 3-variable disjunctive constraints?
  - Example: (x1 || x2 || !x3) && (x1 || !x1 || x1) && (x2 || x3 || x4)
  - Solution: x1 = true, x2 = true, x3 = true, x4 = false

3SAT Reduces to Independent Set

- Proposition: 3SAT Reduces to Independent Set
- Proof: Given an instance A of 3-SAT, create an instance G of Independent-set
  - For each clause in A, create 3 vertices in a triangle
  - Add an edge between each literal and its negation (can't both be true in 3SAT means can't be in same set in Independent-set)

Reduction

- Since IND-SET can be used to solve 3SAT, we say that "3SAT reduces to IND-SET"
  - Note: 3SAT is not a graph problem!
  - Note: Reductions don't always involve creating graphs

Reductions and Decomposition

- Arguably, we've been doing something like reduction all throughout the course
  - Abstract lists reduce to arrays
  - Percolation problem reduces to DisjointSets
- These examples aren't reductions exactly
  - We aren't just calling a subroutine
  - A better term would be decomposition: Taking a complex task and breaking it into smaller parts. This is the heart of computer science
    - Using appropriate abstractions makes problem solving vastly easier