Design and Analysis of Algorithms

9.3 Dealing with hard problems

Daniel Shannon

May 25th, 2022

9.3.2

- Brute force solution: Try every possivle solution and find the best.
- How long does this take for the following? Assume problem size is N.
  - Minimum Spanning Tree E^N
  - Traveling salesman $N!$
  - SAT 2^N

9.3.4

- A Hamiltonian cycle is a cycle that visits every vertex exactly once.
- Brute force: n! possible cycles.
- Design a more efficient backtracking algorithm.

Similar to backtracking for SAT, we can start searching at a node and mark each neighboring node as explored or not. If the explored node connects with another explored node, we stop searching. I think maybe the tallest branch with a height of V or V-1 of the resulting tree would a the Hamiltonian cycle.

9.3.6

- You have an N x N chessboard.
- You want to place N queens on the chessboard so that no two queens are in the same row or column.
- Brute force: N^{2N}
- Design a more efficient backtracking algorithm.

We could start at a vertex v and begin a search. Any vertex that is directly in line or diagonal to the current vertex can be a termination point on the tree. Now how to do this optimally...? Maybe include back, cross, and forward edges.

9.3.8

- You are given a weighted graph.
- Find a least-cost cycle that visits every node exactly once.
- In the naïve brute-force algorithm, how many possible cycles are there?
- Design a branch-and-bound solution to the traveling salesman problem.

This is similar to the hamiltonian cycle problem, but with each node bound by a budget. Starting at v_i, we can explore each of the paths and determine the next node v_{i+1} and determine which will use the least of the available budget. So at each step we are looking for the smallest budget to be removed and we can remove branches of the search tree.