

Matroids and Greedy Algorithms

Siddharth Mahendraker

July 11, 2013

Abstract

This is the abstract.

Contents

1.0 Introduction	1
2.0 Matroid Theory	2
3.0 Greedy Algorithms and Matroids	2
4.0 Optimality	2
5.0 Example: Kruskals Algorithm	2
References	3

1.0 Introduction

Combinatorial optimization is a mathematical framework for finding optimal objects amongst a large set of objects. A famous example of a combinatorial optimization problem is the traveling salesman problem, or TSP. Assume a salesman has a list of cities he wants to visit, and knows the distance between various pairs of cities. The TSP asks: What is the shortest path which takes the salesman through each city once and back to his starting point? Here, the optimal object is a shortest path which passes through each city once and returns the salesman to his point of departure, and the set of objects is the set of all paths which can be taken. To see why this problem could be difficult, assume that every city is connected to every other city and we want to check all the paths to find the shortest one. If we fix a city of departure, there are $n - 1$ cities left to go to, where n is the total number of cities. Each of these choices represents a different path the salesman could take. In the next city, there are $n - 2$ cities left to go to, and so on. It turns out, given n cities there are $(n - 1)!/2$ unique paths which reach every city and return the salesman to his point of departure. Even for relatively small values of n , the number of possible paths is enormous; far larger than anything even a computer could hope to check. Consider for example the choice of $n = 64$. In this case, there are over 10^{86} possible paths to check! More paths than there are atoms in the universe! Using combinatorial optimization techniques, however, we can significantly reduce the amount of work needed to find an optimal path. Indeed, to date, instances of the TSP have been solved with tens of thousands of cities.

One particular strategy often used in combinatorial optimization is the greedy optimization strategy, or greedy algorithm. The greedy algorithm attempts to find the optimal object by making successive locally optimal decisions, in the hopes that this will lead to a globally optimal solution. For example, a greedy algorithm for the TSP could work as follows: Fix the city of departure. To reach another city, pick the shortest route out of the current city which leads to an unvisited city. Repeat until you return to the city of departure. Unfortunately, greedy strategies are often non-optimal. In the case of the TSP, this particular greedy algorithm will not yield the optimal path, and in certain situations, can in fact give the worst possible solution. More generally, it is known that there do not exist greedy algorithms which yield optimal solutions to the TSP.

However, for certain combinatorial optimization problems, greedy algorithms can be very useful. If they can be proven to be optimal, greedy algorithms are often far simpler to implement than other combinatorial optimization strategies, such as dynamic programming. Furthermore, they often consume less resources when implemented on a computer and thus run faster and more efficiently. Therefore, knowing when a greedy algorithm can be applied is very useful.

In this paper, we examine the necessary and sufficient conditions for a greedy algorithm strategy to be optimal. This is done through the study of matroids and their properties, which as we shall soon see, lead naturally to greedy algorithm strategies.

2.0 Matroid Theory

3.0 Greedy Algorithms and Matroids

4.0 Optimality

5.0 Example: Kruskals Algorithm

References