CS4234: Optimization Algorithms Lecture 6

The Asymmetric Traveling Salesman

Lecturer: Seth Gilbert *September 15, 2015*

Abstract

This short note gives a quick overview of the Asymmetric Traveling Salesman Problem, along with a $\log n$-approximation algorithm. (The remainder of the class can be found separately.)

1 The Asymmetric Traveling Salesman Problem

Today we consider the Asymmetric Traveling Salesman Problem, often abbreviated ATSP. The key difference between typical TSP and asymmetric-TSP is that the distance function may not be symmetric. That is, for two locations u and v, it is possible that $d(u, v) \neq d(v, u)$. We will continue to assume that $d(u, v) \geq 0$ for all pairs, and that the triangle inequality holds, i.e., $d(u, w) \leq d(u, v) + d(v, w)$. If the triangle inequality does not hold, we can simply consider the metric completion, i.e., define $d(u, v)$ to be the length of the shortest path between u and v. Thus, as in the symmetric case, we can find useful (approximate) solutions as long as either the distance function satisfies the triangle inequality or repeated visits are allowed.

1.1 Problem Definition

Given a set of cities (i.e., points), the goal of the traveling salesman problem is to find a minimum cost circuit that visits all the points. More formally, the problem is stated as follows:

Definition 1 *Given a set V of n points and a distance function $d : V \times V \to \mathbb{R}$, find a cycle C of minimum cost that contains all the points in V. The cost of a cycle $C = (e_1, e_2, \ldots, e_m)$ is defined to be $\sum_{e \in C} d(e)$, and we assume that the distance function is non-negative (i.e., $d(x, y) \geq 0$).*

The only difference from the standard definition is that the distance function may not be symmetric.

Clearly, the asymmetric-TSP problem is NP-hard (as it is a strict generalization of the symmetric version). Our goal today will be to develop an approximation algorithm.

1.2 Wikipedia Fail

Currently (as of September 2015) the Wikipedia entry for the Traveling Salesman Problem indicates that the asymmetric variant can be solved by reduction to the symmetric variant, i.e., by constructing a symmetric graph G' where the optimal cycle in G is equal to the optimal cycle in the asymmetric instance.

There is a sense in which this claim may be true, while at the same time being completely useless in practice. The first thing to note is that the reduction only holds for the version of TSP which allows for no repeated visits, and the distance function constructed does not satisfy the triangle inequality. Thus, there is no good approximation algorithm for the symmetric instance constructed.

Even for heuristics (with no approximation guarantee, but that perhaps tend to find a good solution in many cases), the instances constructed are particularly hard to solve well. Essentially, for the claim to be true, the new graph G' must have edges that are arbitrarily negative, e.g., $-w$ for some large w. Thus, the overall value of the cycle in the symmetric graph is approximately $-w$, where w is entirely unrelated to the value of the optimal cycle in the asymmetric instance.

From the perspective of approximation algorithms, then, this reduction is unhelpful: it is very easy to find a good approximation in the symmetric graph G' with cost approximateio $-w$, while failing entirely to find a cycle that is approximately good in the asymmetric instance. Heuristics have the same problem: they are easily biased by the large negative weight edges (which are irrelevant to the optimal solution), and hence tend not to converge.

Thus, while there is a sense in which the claim is correct, it is clearly useless for the perspective of approximation and likely useless for the purpose of heuristics and other approaches. I would not recommend trying to solve asymmetric-TSP via this type of reduction.

1.3 Cycle Cover

In order to solve asymmetric-TSP, we first look at another problem, one that can be seen as a relaxation of the original problem. Instead of finding exactly one cycle that visits all the nodes, we instead find an arbitrary number of cycles. Perhaps surprisingly, this relaxed version of the problem can be solved optimally.

More formally, the problem of cycle cover is as follows:

Definition 2 *Given a set V of n points and a distance function $d : V \times V \to \mathbb{R}$, find disjoint cycles $C_1, C_2, \ldots, C_k$ of minimum cost that contains all the points in V. The cost of a cycle $C = (e_1, e_2, \ldots, e_m)$ is defined to be $\sum_{e \in C} d(e)$, and we assume that the distance function is non-negative (i.e., $d(x, y) \geq 0$).*

That is, every point is contained in one of the cycles, and no two cycles overlap (i.e., contain a shared point). We assume that a cycle has to be of length at least 2.

Notice that if $k = 1$, i.e., if we find just one cycle that covers everything, then we have solved the asymmetric-TSP problem. Thus, this problem is often seen as a relaxation of ATSP, i.e., it is identical to ATSP except that a single constraint has been deleted, i.e., the constraint $k = 1$. This also immediately implies that $OPT(CC) \leq OPT(TSP)$, where $OPT(CC)$ refers to the optimal cycle cover while $OPT(TSP)$ refers to the optimal TSP tour. (Notice that the optimal TSP tour is also a valid cycle cover, to the optimal cycle cover cannot be worse than that.)

Consider the following linear program for finding the minimal cost cycle cover, where we have variables x_e for every $e \in E$:

$$\begin{aligned} \min \sum_{e \in E} x_e w(e) & \\ \text{such that} & \\ \forall u \in V \quad & \textstyle\sum_{v:(u,v) \in E} x_{u,v} = 1 \\ \forall u \in V \quad & \textstyle\sum_{v:(v,u) \in E} x_{v,u} = 1 \\ \forall e \in E \quad & x_e \geq 0 \end{aligned}$$

Imagine, for now, that the x_e varaibles are integral, i.e., $x_e \in \{0, 1\}$. The idea of this linear program is that every node has exactly one incoming edge selected and exactly one outgoing edge selected. That is, if you examine simply the edges where $x_e = 1$, we find a collection of cycles. (To construct each cycle, simply follow the outgoing edges sequentially until eventually we revisit a node we have already seen.) Similarly, any cycle cover satisfies this linear program, and so the integral solution is easily seen to be the optimal cycle cover.

The interesting fact is that, for this linear program, there is always an optimal solution that is integral. We will not prove this fact here. (Talk to me if you want to hear more.)

From this point on, we will assume we can find the optimal solution to the minimum cost cycle cover problem.

1.4 Approximation Algorithm

We now consider the following recursive algorithm, given $n \geq 2$ points and distances d:

1. Find a cycle cover $C_1, C_2, \ldots, C_k$.
2. Choose one point $x_j \in C_j$ from each cycle. This yields k points $x_1, x_2, \ldots, x_k$. Notice that $k \leq n/2$ (since each cycle is of length ≥ 2).
3. If $k = 1$, return the cycle C_1.
4. If $k > 1$, recursively calculate the asynchronous-TSP cycle of points $(x_1, x_2, \ldots, x_k)$. Let D be the cycle returned.
5. Paste together the cycle D with the cycles $C_1, C_2, \ldots, C_k$ and return that new cycle.

In order to paste cycle D with the cycles returned in Step 1, proceed as follows, building the new cycle as you go:

1. Start with $j = 1$ and at node $x = x_1$ in cycle D.
2. Repeat k times:
   (a) Follow cycle C_j until it returns to point x.
   (b) Follow cycle D from x to y.
   (c) Set $x = y$.
   (d) Let j be the cycle where $x \in C_j$.
3. Short-cut all the repeated nodes in the newly constructed cycle.

It is relatively easy to see that the result is a valid TSP tour: recursively we know that the cycle D is a valid TSP tour of the points $x_1, \ldots, x_k$, and the cycle cover guarantees that every other point is incluced in some cycle.

We now consider the approximation guarantee. As already stated, we know that $OPT(CC) \leq OPT(TSP)$. Notice that in total, there are at most $\log n$ recursive calls, and hence at most $\log n$ executions of the cycle cover algorithm—this follows because each cycle must contain at least 2 nodes.

Let $CC_1, CC_2, \ldots, CC_{\log n}$ be the (at most $\log n$) executions of cycle cover. The total cost of the resulting cycle is at most $\sum_{j=1}^{\log n} CC_j$, i.e., the sum of the cost of the cycle covers. (This is because pasting simply sums the edge costs, perhaps reducing the cost a bit by short-cutting.)

Finally, notice that each cycle cover $CC_j \leq OPT(TSP)$. Imagine we are executing a cycle cover on points $x_1, x_2, \ldots, x_k$ at some stage j of the recursion. If we begin with $OPT(TSP)$ for all the points and short-cut to points $x_1, \ldots, x_k$, we get a new cycle that has cost at most $OPT(TSP)$. So the optimal TSP tour on points $x_1, x_2, \ldots, x_k$ is no more than $OPT(TSP)$. And we know that the cost of CC_j is at most the cost of the optimal TSP tour on $x_1, x_2, \ldots, x_k$. Putting these together, we get that $CC_j \leq OPT(TSP)$.

Overall, this leads to the conclusion that the cycle constructed has cost at most $\log n \cdot OPT(TSP)$.