Rounding Techniques in Approximation Algorithms

Lecture 8: Splitting off to Approximate Prize-Collecting Problems

Lecturer: Nathan Klein *Scribe: Thodoris Tsilivis*

1 Introduction

This lecture will focus on a variant of the metric Traveling Salesperon Problem, or metric TSP. Here we are given a set of vertices V and a symmetric metric $c : V \times V \to \mathbb{R}_{\geq 0}$, and our goal is to output a minimum cost Hamiltonian cycle. We will use that it is equivalent to find a minimum cost spanning Eulerian multi-subgraph, which follows because such a graph can be shortcut into a Hamiltonian cycle with no greater cost due to the triangle inequality.[1]

1.1 Christofides-Serdyukov Algorithm

There is a simple and beautiful $\frac{3}{2}$ approximation for metric TSP from Christofides [Chr76] (and independently Serdyukov [Ser78]) in the 1970s.

Algorithm 1 Christofides-Serdyukov Algorithm

Find a minimum cost spanning tree T of G
Let O be the set of vertices with odd degree in T. Compute the minimum cost perfect matching M on O.
Return $T \uplus M$.

To prove that this is in fact a $\frac{3}{2}$ approximation, we will break down the cost of the solution into the cost of the spanning tree T and the cost of the matching M.

For the spanning tree, we can readily argue that its cost is at most OPT. That is because any Hamiltonian cycle can be decomposed into a spanning tree plus an edge. That spanning tree cannot have cost smaller than the minimum cost spanning tree, so $c(T) \leq c(OPT)$.

For the matching, we will consider the odd degree vertices of T. Now, notice that since there are an even number of odd vertices (by the handshake lemma), the optimal solution can be decomposed into two collections of paths with degree 1 on every odd vertex and degree 2 on every even vertex of T. Using the triangle inequality these paths can be shortcut into two matchings M_1 and M_2 of no greater cost. Thus, $c(M_1) + c(M_2) \leq c(OPT)$, implying that either M_1 or M_2 has cost at most $\frac{OPT}{2}$. So, the minimum cost matching found by the algorithm obeys $c(M) \leq \frac{OPT}{2}$.

Thus we can combine our two arguments, to get that the cost of $T \uplus M$ is at most $OPT + \frac{OPT}{2} = \frac{3}{2} \cdot OPT$.

1.2 Polyhedral View

Another approach towards solving the problem is to try to formulate an LP and come up with a rounding argument. Towards that end, we define the Subtour LP relaxation for TSP.

[1] Formally, given an Eulerian graph, let $v_1, \ldots, v_1$ be an Eulerian tour in which vertices may appear more than once. Keep only the first occurrence of every vertex in this sequence, except v_1 for which we keep the first and last. The resulting Hamiltonian cycle has no greater cost than the Eulerian graph due to the triangle inequality.

$$P_{\text{Sub}} := \begin{cases} x(\delta(S)) \geq 2 & \forall\, S \subsetneq V \\ x(\delta(v)) = 2 & \forall v \in V \\ x_{\{u,v\}} \geq 0 & \forall\, u, v \in V \end{cases}$$

Theorem 1.1 ([Wol80]). *The integrality gap of P_{Sub} is at most $\frac{3}{2}$.*

Proof. Similarly to before, we will need to connect the cost $c(x)$ to the cost of some MST and the cost of some matching.

To bound the cost of the MST, we start by noticing the relationship between P_{Sub} and P_{2-EC}. That is that for any $x \in P_{Sub}$ it also holds that $x \in P_{2-EC}$. Furthermore, any $x \in P_{Sub}$ is parsimonious since we have the. constraints $x(\delta(v)) = 2$ for all $v \in V$. Thus we can use the lemma from Lecture 6 to argue that $x \cdot \left(1 - \frac{1}{n}\right) \in P_{ST}$. So, x dominates a convex combination of spanning trees of G, implying that $c(T) \leq c(x)$ for any MST T.

We will now bound the cost of the matching. Towards that end we need to define the dominant of the matching polytope which is:

$$P_{\text{M}}^{\uparrow} := \begin{cases} x(\delta(S)) \geq 1 & \forall\, S \subsetneq V,\, S \cap odd(T) \text{ is odd} \\ 0 \leq x_e & \forall\, e \in E \end{cases}$$

It turns out that this polyhedron has integral vertices (proving this is a bonus problem on Homework 3). But now, $\frac{x}{2} \in P_{\text{M}}^{\uparrow}$ and has cost $\frac{c(x)}{2}$. So the minimum cost matching has cost at most $\frac{c(x)}{2}$, and combining the tree and the matching gives us the theorem. □

1.3 Prize Collecting TSP

In this problem we are given a set of vertices V, a symmetric metric $c : V \times V \to \mathbb{R}_{\geq 0}$ and a penalty $\pi(v)$ for all vertices in V. Now our goal is to output a Hamiltonian cycle on some $S \subseteq V$ such that the cost of the cycle plus the penalty of the vertices not visited is minimized. That is we want $S = \arg\min_{S \subseteq V} \text{cost}(S) + \sum_{v \notin S} \pi(v)$.

It is quickly apparent that we cannot run the Christofides-Serdyukov Algorithm, since that would require to somehow know which subset S of V we should get the Hamiltonian cycle on. However, we can still consider whether an LP can yield something interesting.

Our new LP variables should be x_e for each $e \in E$, again indicating whether edge e is used in the Hamiltonian cycle, and y_v for each $v \in V$, which will indicate if vertex v is visited in the cycle. We can now define to prize collecting TSP polytope as follows:

$$P_{PCTSP} = \begin{cases} x(\delta(v)) = 2y_v & \forall v \in V - \{r\} \\ x(\delta(r)) \leq 2 & \\ x(\delta(S)) \geq 2y_v & \forall S \subseteq V - \{r\}, v \in S \\ y_r = 1 & \\ x_e \geq 0 & \forall e \in E \\ y_v \geq 0 & \forall v \in V \;. \end{cases}$$

Our objective will be $\min \sum_{e \in E} c_e x_e + \sum_{v \in V} \pi_v (1 - y_v)$, as we want to minimize the cost of the tour plus the cost of the penalties on the vertices we do not visit.

To use this LP to get an approximate solution for the problem, we need to now define the splitting off operation, which allows us to remove pairs of edges adjacent to some vertex v from a graph without decreasing the pairwise connectivity of vertices in $V \setminus \{v\}$.

2 Splitting Off

Theorem 2.1 (Splitting Off [Fra92]). *Given a multigraph $G = (V, E)$ and any vertex $u \in V$ of even degree $2k$, if u is not incident to any cut edge (an edge whose removal would disconnect the graph), the edges uv adjacent to u can be paired up into groups $(ua_1, ub_1), (ua_2, ub_2), \ldots (ua_k, ub_k)$ such that after removing all the edges adjacent to u and adding edges $\{a_i, b_i\}$ for all i the pairwise connectivities a, b for any vertices $a, b \neq u$ do not decrease.*

Even though splitting off is defined for integral graphs, this procedure can naturally be extended to fractional graphs. The basis of the argument is that we can multiply all variables x_e's of the fractional graph with a large number r so that all of them become integral. Then, we can partition any edge of value x_e into $rx_e \in \mathbb{Z}$ parallel edges, and proceed with splitting off as you would in an integral multigraph. After all operations are applied, you can reverse the multiplication and get the desired fractional graph. Furthermore, observe that we can choose r so that all vertices have even degree: in this way, **any vertex in a fractional graph can be split off to obtain a new fractional graph**. This will be useful shortly, and in fact allows us to extend the above theorem to integral graphs if the output graph is allowed to be fractional as demonstrated in the below Fig. 1.

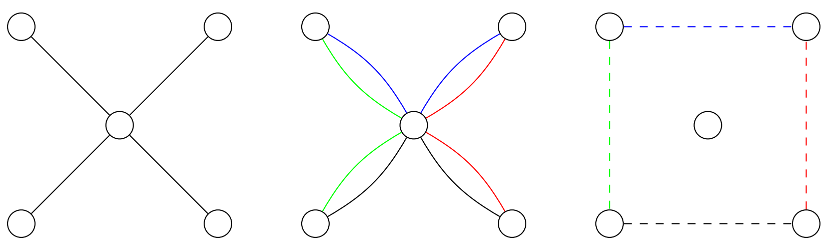

Figure 1: Consider the graph on the left. The middle vertex cannot be split off because it is adjacent to four cut edges. However, if we think of this as a fractional graph, we can multiply the edges by two and then split off to obtain a valid (but fractional) split-off solution where all edges have value $\frac{1}{2}$.

We will not discuss running time in this lecture. See [BN23; BKN23] for details: everything we discuss can be made to run in polynomial time.

2.1 Decomposition Theorem and 2-Approximation

We will now use this to prove the following theorem from [BN23].

Theorem 2.2 ([BN23]). *Let $(x, y) \in P_{PCTSP}$. Then there exists a distribution μ over trees T rooted at r such that:*

$$\mathbb{P}[e \in T] \le x_e, \forall e \in E \ , \qquad \text{and} \qquad \mathbb{P}[v \in T] = y_v, \forall v \in V \ .$$

Notice that access to this distribution μ already suffices to design an algorithm that yields a 2-approximation to this problem. Consider an algorithm that samples one tree T from μ, doubles its edges, and shortcuts the resulting Eulerian graph. The expected cost of the resulting cycle is at most $2c(x)$ and the expected penalty is $\sum_{v \in V} \pi_v(1 - y_v)$. Since the LP relaxation has cost $c(x)$ and penalty $\sum_{v \in V} \pi_v(1 - y_v)$, we have proven that this algorithm is a 2-approximation. (And in fact, the ratio is even better if the penalty cost incurred by the LP was greater than 0).

2.2 Proof of the Decomposition Theorem

We now need to show how the distribution μ can be constructed. The argument will be the following:

1. Begin from a point in P_{PCTSP}. Apply splitting off until the graph has two vertices left (one of them being the root r). At this point, show the theorem holds. In other words, show how to construct a distribution for two vertices.

2. Undo the splitting off vertex by vertex, at each point extending the existing distribution over trees into one that includes the new vertex.

For the base case with two vertices (root r and v), the distribution is fairly simple. With probability y_v return the tree with vertices y and v connected and with probability $1 - y_v$ return just the root r.

For the inductive hypothesis, suppose our current set of vertices V has size at least 2. Let s be the vertex with the minimum y_s. Fully split off s and apply the inductive hypothesis on $G' = V \setminus s$. That yields a distribution μ with the desired properties on G'.

We now need to undo the splitting off operation. Since for a single vertex, multiple splitting off operations may have been required. We will analyze what happens for one of them and that will suffice to prove that the IH holds after undoing all operations. Towards that end, consider a specific splitting off operation for vertex s and for its edges (su, sv) in which these two edges decreased by δ and edge uv increased by δ for some $\delta > 0$.

First initialize for every vertex in V counters spare$[v] = 0$. Now consider all trees in the distribution that include uv. If the distribution sets probability mass on these trees less than δ (that is $\sum_{T:uv \in T} \mu(T) = \gamma < \delta$) then define $\mathcal{T}$ to be all such tress. Otherwise, collect trees from the support of μ until we obtain probability mass exactly δ (we can do so WLOG by splitting trees in the support into identical copies and adjusting their probabilities as needed). Using one of the two we can properly define a collection of trees $\mathcal{T}$ such that:

$$uv \in T, \forall T \in \mathcal{T} \ , \qquad \text{and} \qquad \sum_{T \in \mathcal{T}} \mu(T) = \gamma \text{ for some } \gamma \leq \delta \ .$$

Now proceed with the following steps for each $T \in \mathcal{T}$:

1. If $s \notin T$, remove uv from T and add su, sv.

2. If $s \in T$, add back one of edges su or sv so that the graph remains a tree. If su is added, increase spare$[v]$ by $\mu(T)$ and otherwise increase spare$[u]$ by $\mu(T)$.

Finally, increase both spare$[u]$ and spare$[v]$ by $\frac{\delta - \gamma}{2}$. We are now ready to prove the following claim which holds after all splitting off operations are undone.

Claim 2.3. $\sum_{T \in \mathcal{T} | s \in T} \mu(T) + \sum_{v \in V} spare[v] = y_s$

Proof. To prove this we need to consider what the steps we defined for each T do. In case (1) we know that the sum on the LHS increases by $\mu(T)$, while in case (2) we know that some counter spare increases $\mu(T)$. Thus after going over all trees the LHS is exactly $\sum_{T \in \mathcal{T}} = \gamma$ and by adding as a final step $\delta - \gamma$ to spares for u and v we get exactly δ on the LHS. To finish the proof of the claim, we use that the degree of s was $2y_s$ so the sum over all δ is y_s (as every splitting operation decreases *two* edges adjacent to s by δ and increases an edge not adjacent to s by δ). □

To finalize the proof for the induction step, we need to take care of the spare counters and ensure that s is in the tree with probability y_s in our new distribution. To that end, consider some v such that spare$[v] \neq 0$ and notice that

$$\mathbb{P}\left[v \in T, s \notin T\right] \geq \mathbb{P}[v \in T] - \mathbb{P}[s \in T] \geq y_s - \mathbb{P}\left[s \in T\right] \geq \text{spare}[v],$$

where for the first inequality we used that y_s was the smallest y value and in the second inequality we used Claim 2.3. That means that can find a collection of trees that includes v but not s with mass spare$[v]$. Modify each tree in this collection by adding the edge (s, v) and set spare$[v] = 0$. Claim 2.3 still holds as we have increased $\mathbb{P}\left[s \in T\right]$ be spare$[v]$ and decreased the sum of spares by the same amount. After performing this operation for all vertices with spare$[v] > 0$, the probability s is in T is exactly y_s as required.

3 Improved Approximation

The natural idea to improving this beyond a factor of 2 would be to apply a matching similar to the Christofides-Serdyukov algorithm. However, it's difficult to bound its cost since the sampled vertices in the algorithm are not guaranteed to be the vertices that some optimal solution visits.

However, there is room for improvement. Since the 2-approximation gets a factor of 1 on the penalty cost, intuitively we should be able to trade off to some degree between the two factors. [BKN23] employs the following strategy to achieve a 1.599 approximation: after sampling a tree, take the subtrees which only contain vertices with small y_v. With some probability, prune all such subtrees. Otherwise, double the edges in these subtrees. In this way, we lose slightly on the penalties for vertices with small y_v and reduce to the case in which all y_v are fairly large. In such a situation, we can show $\alpha \mathbb{I}\left\{T\right\} + \beta x \in P_M^{\uparrow}$ for some $\alpha + \beta < 1$, allowing us to correct the parity of the tree with lower cost than simply doubling all of the edges. In some way we are interpolating between the algorithm which doubles the tree and the matching-based algorithm we analyzed in Section 1.2. The key fact here is that once we have removed subtrees with small y_v values, say $y_v \leq \kappa$, all cuts either have at least three edges in three or have $x(\delta(S)) \geq 2\kappa$. See [BKN23] for further details.

The main remaining open question is whether or not there exists an algorithm that achieves $\frac{3}{2}$ approximation, or even better an approximation-preserving reduction from TSP.

References

[BKN23] Jannis Blauth, Nathan Klein, and Martin Nägele. *A Better-Than-1.6-Approximation for Prize-Collecting TSP*. 2023. arXiv: 2308.06254 [cs.DS] (cit. on pp. 3, 5).

[BN23] Jannis Blauth and Martin Nägele. "An Improved Approximation Guarantee for Prize-Collecting TSP". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. STOC 2023. Orlando, FL, USA: Association for Computing Machinery, 2023, 1848–1861. ISBN: 9781450399135. DOI: 10.1145/3564246.3585159 (cit. on p. 3).

[Chr76] Nicos Christofides. *Worst Case Analysis of a New Heuristic for the Traveling Salesman Problem*. Report 388. Pittsburgh, PA: Graduate School of Industrial Administration, Carnegie-Mellon University, 1976 (cit. on p. 1).

[Fra92] András Frank. "On a theorem of Mader". In: *Discrete Mathematics* 101.1 (1992), pp. 49–57. ISSN: 0012-365X. DOI: https://doi.org/10.1016/0012-365X(92)90589-8 (cit. on p. 3).

[Ser78] A. I. Serdyukov. "O nekotorykh ekstremal'nykh obkhodakh v grafakh". In: *Upravlyaemye sistemy* 17 (1978), pp. 76–79 (cit. on p. 1).

[Wol80] Laurence A. Wolsey. "Heuristic analysis, linear programming and branch and bound". In: *Combinatorial Optimization II*. Vol. 13. Mathematical Programming Studies. Springer Berlin Heidelberg, 1980, pp. 121–134 (cit. on p. 2).