A Summary of the Current State of the Art Algorithms for the Asymmetric Traveling Salesman Problem

Alex Wong (asw2181) - asw2181@columbia.edu

April 30, 2017

Abstract

We look at some of the state of the art algorithms for the Asymmetric Traveling Salesman problem and its current known inapproximability bounds. The first algorithm, by Asadpour et al.[2], details an $O(\log\ n\ /\ \log\log\ n)$-approximation algorithm for the general case of ATSP for costs satisfying the triangle inequality. The second algorithm, by Svensson et al.[10], details a constant factor approximation algorithm for a special case of ATSP that contains only two different edge weights of arbitrary weight where the costs satisfy the triangle inequality. Lastly, an algorithm by Karpinski et al.[7] gives us a hardness of approximation bounds of 75 / 74 for ATSP.

1 Introduction

In Section 2, we provide an overview of an $O(\log\ n\ /\ \log\log\ n)$-approximation algorithm for the general case of ATSP for costs satisfying the triangle inequality which comes from the work of Asadpour et al. [2]. The $O(\log\ n\ /\ \log\log\ n)$-approximation algorithm introduces the concept of the "thinness" of a spanning tree where the thinness correlates with the approximation bound of ATSP. This is an improvement over the decades-long standing $\Theta(\log n)$-approximation bound given by the work of Frieze et al. [4].

a. For $i = 1, ..., |N|$ we assign node i to set S with probability 1/2 and to set $N - S$ with probability 1/2. For each directed edge (i, j), let $Z_{(i,j)} = 1$ if $(i, j) \in$ directed cut from S to N-S, else 0 otherwise. Thus, we get the equation:

$$E[Z_{(i,j)}] = P[(i, j) \in \text{directed cut from S to N-S}]$$

$$= P[i \in S, j \in N - S] = P[i \in S]P[j \in N - S] = 1/2 \cdot 1/2 = 1/4$$

□

2 An $O(\log\ n\ /\ \log\log\ n)$-approximation algorithm for ATSP

2.1 An $O(\log\ n\ /\ \log\log\ n)$-approximation algorithm

The following is the $O(\log\ n\ /\ \log\log\ n)$-approximation algorithm for ATSP (see [2] sect. 1):

Throughout the rest of this section, we will divide each step of the algorithm into subsection(s) to describe it in more detail and/or prove its correctness.

Algorithm 1 An $O(\log\, n\, /\, \log\log\, n)$-approximation algorithm for ATSP

Input: A set V consisting of n points and a cost function $c : VxV \to \mathbb{R}^+$ where the costs satisfy the triangle inequality.
Output: An $O(\frac{logn}{loglogn})$-approximation to the ATSP instance described by the input's V and c.
Define $P := T := \{\{1\}, \ldots, \{d\}\}$
while $\#P > 1$ **do**
  Choose $C' \in \mathcal{C}_p(P)$ with $C' := \text{argmin}_{C \in \mathcal{C}_p(P)}\, \varrho(C)$
  Find an optimal partition tree $T_{C'}$
  Update $P := (P \backslash C') \cup \{\bigcup_{t \in C'} t\}$
  Update $T := T \cup \{\bigcup_{t \in \tau} t : \tau \in T_{C'} \backslash \mathcal{L}(T_{C'})\}$
end while
return T

2.2 Preliminaries and Notation

In this subsection, we describe much of the notation that will be used throughout the rest of this section.

2.3 2ab

b. Deterministic Algorithm (MCE) for the Maximum Directed Cut problem:
We first define α as the deterministic assignment of nodes x_i, $i = 1, ..., |N|$, either in set S or in set $N - S$. At the beginning of the algorithm, $\alpha = \emptyset$. The following is the MCE algorithm: For $i = 1, ..., |N|$,

if $E[Z \mid \alpha_{i-1} \cup \{x_i \in S\}] \geq E[Z \mid \alpha_{i-1} \cup \{x_i \in N - S\}]$

then $\alpha = \alpha \cup \{x_i \in S\}$

else $\alpha = \alpha \cup \{x_i \in N - S\}$

Algorithm 2 Build tree

Define $P := T := \{\{1\}, \ldots, \{d\}\}$
while $\#P > 1$ **do**
  Choose $C' \in \mathcal{C}_p(P)$ with $C' := \text{argmin}_{C \in \mathcal{C}_p(P)}\, \varrho(C)$
  Find an optimal partition tree $T_{C'}$
  Update $P := (P \backslash C') \cup \{\bigcup_{t \in C'} t\}$
  Update $T := T \cup \{\bigcup_{t \in \tau} t : \tau \in T_{C'} \backslash \mathcal{L}(T_{C'})\}$
end while
return T

After the algorithm runs, α will contain the assignment of the nodes that has a directed cut with at least 1/4 of the edges. How $E[Z]$ is determined is as follows:

For $1 \leq i \leq |N|$:

$$E[Z \mid \alpha_i] = max \begin{cases} E[Z \mid \alpha_{i-1} \cup \{x_i \in S\}] = cut\{S_i, N - S_i\} + \frac{1}{4}(\#\text{out edges from } x_i \text{ to } x_{i+1},...,x_{|N|}) \\ E[Z \mid \alpha_{i-1} \cup \{x_i \in N - S\}] = cut\{S_i, N - S_i\} + \frac{1}{4}(\#\text{in edges from } x_{i+1},...,x_{|N|} \text{ to } x_i) \end{cases}$$

c. The worst-case approximation ratio of the simple local search algorithm is unbounded. Take for example a node i and some number n of nodes where all edges are directed towards i. If we put node i in set S and the rest of the nodes in set $N - S$, then all the edges are directed from $N - S$ to S, giving a cut with 0 edges since only edges from S to $N - S$ count. Using only one local move will not improve the cut, regardless of which node we decide to move. If we move node i to $N - S$, then all nodes will be on that side. If we move any of the n nodes from $N - S$ to S, then the edge will reside within S, still not improving the cut. We can see that the maximum cut would have the n number of nodes in set S and node i in set $N - S$, which would give us a cut with n number of edges. Thus, we can see that the worst-case approximation ratio is unbounded using the simple local search algorithm since we can scale n to any arbitrarily large number.

d. We can improve the local search so that it guarantees ratio 1/4. The improvement of the local search algorithm is summarized below:

Given some directed graph $D = (N, E)$ and a random assignment of the nodes $i = 1, ..., |N|$ into either set S or $N - S$, while $\exists$ a node i that has more neighbors on the same side than the opposite side, move i to the opposite side. Based on slides 23 and 24 of the Lecture 5 slides, we will end up with $\geq 1/2|E|$ of edges that go between the sets S and $N - S$. From this, we find the value of the directed cut of these edges. Then, we move all the current nodes that are in S to $N - S$ and the current nodes of $N - S$ to S and find the value of the directed cut of these edges. We take the configuration of S and $N - S$ based on the larger of the two directed cuts that we found, which at least one of them $\geq 1/4|E|$. One of the cuts will be $\geq 1/4|E|$ because there are a total of $\geq 1/2|E|$ edges in the cut, so if $\leq 1/4|E|$ edges go from S to $N - S$ in one configuration, then there must be $\geq 1/4|E|$ edges going from $N - S$ to S, so we would move the nodes to orient the direction of the edges to go from S to $N - S$.

e. We can prove that the optimal value of the ILP is equal to the optimal value of the MAX DICUT problem by showing a mapping between ILP and MAX DICUT:

$x_i \in \{0, 1\}, \forall i \in N \Leftrightarrow$ the two different classes that $node_i$ can be in:
1.$x_i = 1 \Leftrightarrow node_i$ being in class S
2.$x_i = 0 \Leftrightarrow node_i$ being in class N-S

$y_{ij} \leq x_i \; and \; y_{ij} \leq 1 - x_j, \forall (i, j) \in E \Leftrightarrow$ the 4 different possibilities that the nodes can be configured in the set:
1. if $x_i = 1$ and $x_j = 0$, then $y_{ij} = 1 \Leftrightarrow$ if $node_i \in S$ and $node_j \in N - S$, then edge (i, j) is in the cut.
2. if $x_i = 0$ and $x_j = 1$, then $y_{ij} = 0 \Leftrightarrow$ if $node_i \in N - S$ and $node_j \in S$, then edge (i, j) is not in the cut.

3. if $x_i = 0$ and $x_j = 0$, then $y_{ij} = 0 \Leftrightarrow$ if $node_i \in N - S$ and $node_j \in N - S$, then edge (i, j) is not in the cut.
4. if $x_i = 1$ and $x_j = 1$, then $y_{ij} = 0 \Leftrightarrow$ if $node_i \in S$ and $node_j \in S$, then edge (i, j) is not in the cut.

maximize $\sum_{(i,j)\in E} y_{ij} \Leftrightarrow$ maximize # of directed edges where $i \in S, j \in N - S, (i, j) \in E$

Thus the optimal value of this ILP is equal to the optimal value of the MAX DICUT problem and we can obtain an optimal solution to the MAX DICUT problem from an optimal solution to the ILP by placing $node_i$ in set S in accordance with $x_i = 1$ and $node_i$ in set $N - S$ in accordance with $x_i = 0$.

f. If we take a LP reduction, then we know that:

$$Pr[\text{edge in cut}] = Pr[x_i \in S, x_j \in N - S] = Pr[x_i \in S]Pr[x_j \in N - S]$$

$$= (\frac{1}{4} + \frac{x_i}{2})(\frac{3}{4} - \frac{x_j}{2})$$
$$= \frac{3}{16} - \frac{x_j}{8} + \frac{3x_i}{8} - \frac{x_i x_j}{4}$$
$$= \frac{3}{16} - \frac{2x_j}{16} + \frac{6x_i}{16} - \frac{4x_i x_j}{16}$$
$$= \frac{6x_i - 4x_i x_j - 2x_j + 3}{16}$$
$$= \frac{6x_i - 2x_j(2x_i - 1) + 3}{16}$$
$$\geq \frac{6y_{ij} - 2x_j(2y_{ij} - 1) + 3}{16}$$
$$\geq \frac{6y_{ij} + 2(y_{ij} - 1)(2y_{ij} - 1) + 3}{16}$$
$$= \frac{6y_{ij} + (2y_{ij} - 2)(2y_{ij} - 1) + 3}{16}$$
$$= \frac{6y_{ij} + 4y_{ij}^2 - 2y_{ij} - 4y_{ij} + 2 + 3}{16}$$
$$= \frac{4y_{ij}^2 + 5}{16}$$
$$= \frac{y_{ij}^2}{4} + \frac{5}{16}$$
$$= \frac{4y_{ij}^2}{16} + \frac{5}{16} \geq 1/2OPT$$

Problem 2

a. The quadratic program for this problem is given below:

$$\max \frac{1}{2} \sum_{i<j} (1 - (y_i \cdot y_j \cdot z_{ij}))$$

$$\text{s.t. } y_i \in \{-1, +1\}, \forall i = 1, ..., n$$

$$z_{ij} \in \{-1, +1\}, i < j$$

The correspondence between the problem and the quadratic program are as follows:

$$\text{object } i \in \text{class } S \Leftrightarrow y_i = +1$$

$$\text{object } i \in \text{class } S' \Leftrightarrow y_i = -1$$

$$\text{similar constraint } i \sim j \Leftrightarrow z_{ij} = -1$$

$$\text{dissimilar constraint } i \nsim j \Leftrightarrow z_{ij} = +1$$

For objects i, j that are in the same class with a similarity constraint:

$$y_i y_j z_{ij} = (-1)(-1)z_{ij} \text{ or } (+1)(+1)z_{ij} = 1 \cdot z_{ij} = (+1)(-1) = 1 \Leftrightarrow 1 - y_i y_j z_{ij} = 2$$

For objects i, j that are in the same class with a dissimilarity constraint:

$$y_i y_j z_{ij} = (-1)(-1)z_{ij} \text{ or } (+1)(+1)z_{ij} = 1 \cdot z_{ij} = (+1)(+1) = -1 \Leftrightarrow 1 - y_i y_j z_{ij} = 0$$

For objects i, j that are in different classes with a similarity constraint:

$$y_i y_j z_{ij} = (-1)(+1)z_{ij} \text{ or } (+1)(-1)z_{ij} = -1 \cdot z_{ij} = (-1)(-1) = -1 \Leftrightarrow 1 - y_i y_j z_{ij} = 0$$

For objects i, j that are in different classes with a dissimilarity constraint:

$$y_i y_j z_{ij} = (-1)(+1)z_{ij} \text{ or } (+1)(-1)z_{ij} = -1 \cdot z_{ij} = (-1)(+1) = 1 \Leftrightarrow 1 - y_i y_j z_{ij} = 2$$

where the objective function is maximizing the number of satisfied constraints.
Thus, we have given a quadratic program for this problem.

b. To get a randomized approximation algorithm whose expected number of satisfied constraints is at least 0.878OPT, we apply a SDP (VP) relaxation on some quadratic program. Notice that the quadratic program that we gave in part **a** maps to the integer quadratic program for MAX CUT (shown in slide 31 of lecture 5 slides) with weights of edges = 1. We show this by mapping from our constraints problem that for objects i, j that are in the same class with a similarity constraint:

$$y_i y_j z_{ij} = (-1)(-1)z_{ij} \text{ or } (+1)(+1)z_{ij} = 1 \cdot z_{ij} = (+1)(-1) = 1 \Leftrightarrow 1 - y_i y_j z_{ij} = 2$$

and for objects i, j that are in different classes with a dissimilarity constraint:

$$y_i y_j z_{ij} = (-1)(+1)z_{ij} \text{ or } (+1)(-1)z_{ij} = -1 \cdot z_{ij} = (-1)(+1) = 1 \Leftrightarrow 1 - y_i y_j z_{ij} = 2$$

map to the MAX CUT problem where some edge $(i, j) \in$ cut. Similarly, we can map from our constraints problem that for objects i, j that are in the same class with a dissimilarity constraint:

$$y_i y_j z_{ij} = (-1)(-1)z_{ij} \text{ or } (+1)(+1)z_{ij} = 1 \cdot z_{ij} = (+1)(+1) = -1 \Leftrightarrow 1 - y_i y_j z_{ij} = 0$$

and for objects i, j that are in different classes with a similarity constraint:

$$y_i y_j z_{ij} = (-1)(+1)z_{ij} \text{ or } (+1)(-1)z_{ij} = -1 \cdot z_{ij} = (-1)(-1) = -1 \Leftrightarrow 1 - y_i y_j z_{ij} = 0$$

map to the MAX CUT problem where some edge $(i, j) \notin$ cut. The objective function of the constraints problem is maximizing the number of satisfied constraints, which maps to the objective function of the MAX CUT of maximizing the weight of the cut, but since all edge weights = 1, it would be essentially maximizing the number of edges in the cut.

Since we have shown that the constraints problem maps to the MAX CUT problem, we can take the SDP relaxation of the IQP for MAX CUT and utilize randomized vector rounding on the SDP, with the analysis being shown from slides 31 to 37 of the lecture 5 slides that the $E[weight\ of\ cut] \geq 0.878OPT$ which implies $E[\#\ satisfied\ constraints] \geq 0.878OPT$.

Problem 3

a. **Lemma 3a** If I is an independent set of G then I^k is an independent set of $G^{(k)}$.

Proof: For I to be an independent set of G, there must be no adjacent edges between any of the nodes of I. Given the definition of $E^{(k)}$ of $G^{(k)}$,

$$E^{(k)} = \{(u, v) \in N^{(k)} \times N^{(k)} \mid \exists i, j \in [k], (u_i, u_j) \in E \text{ or } (v_i, v_j) \in E \text{ or } (u_i, v_j) \in E\}$$

we can see that there are no edges between any pairs (u, v) of k-tuples of nodes that only contain nodes of I. When taking the Cartesian product of I, I^k, we create k-tuples using only those nodes from I so $I^k \subseteq N^{(k)}$ since $I \subseteq N$. Also, as explained earlier, the k-tuples of I^k do not have adjacent edges between them since they only contain nodes of I, thus showing that I^k is an independent set of $G^{(k)}$. □

b. **Lemma 3b** If J is an independent set of $G^{(k)}$ then we can construct in polynomial time an independent set I of G of size at least $|J|^{1/k}$ and conclude that $\alpha(G^{(k)}) = (\alpha(G))^k$

Proof: From Lemma 3a, if I is an independent set of G, then I^k is an independent set of $G^{(k)}$. Since we have defined J to be an independent set of $G^{(k)}$, we can use Lemma 3a to say that $J = I^k$. The size of J is then equal to the size of I^k, which is $|I|^k$, thus $|J| = |I|^k \implies |I| = |J|^{1/k}$, showing that we can construct in polynomial time an independent set I of G of size at least $|J|^{1/k}$. Also, if given the maximum independent set of G, $\alpha(G)$, we can use our earlier conclusion and see that $(\alpha(G^{(k)}))^{1/k} = \alpha(G) \implies \alpha(G^{(k)}) = (\alpha(G))^k$. □

c. **Lemma 3c** If the Maximum Independent Set problem can be approximated in polynomial time within some constant factor $c > 1$, then it has a PTAS.

Proof: We first define $\alpha(G)$ as the maximum independent set of G. From the PCP theorem, we can trivially say that the Maximum Independent Set problem has a 2-approximation algorithm. Let us define I as an independent set of G that agrees with the 2-approximation algorithm. This means that the size of I will be $1/2$ the size of the maximum independent set of G, thus $\frac{|\alpha(G)|}{|I|} = 2$. Now, using Lemma 3b, if we take the k-th power of the graph G, we know that $\alpha(G^{(k)}) = (\alpha(G))^k$ and by Lemma 3a, I^k is an independent set of $G^{(k)}$. Thus, for some k-th power graph $G^{(k)}$, the approximation ratio is $\frac{|\alpha(G)|^k}{|I|^k} = (\frac{|\alpha(G)|}{|I|})^k = 2^k$ which shows that the approximation ratio grows as we apply the approximation algorithm to larger and larger k-th powers of graph G. Thus, the Maximum Independent Set problem can not be approximated in polynomial time within some constant factor, which means it does not have a PTAS unless $\mathbf{P = NP}$. □

Problem 4

a. **Lemma 4a** MDAS can be trivially approximated within a factor of 2.

Proof: Suppose we have an arbitrary ordering of nodes $v_1, ..., v_n$ and we have two subsets of edges A_1 and A_2 where

$$A_1 = \{(v_i, v_j) \mid (v_i, v_j) \in A, i < j\}$$

$$A_2 = \{(v_i, v_j) \mid (v_i, v_j) \in A, i > j\}$$

By separating the edges into these two subsets, the only way for A_1 to contain a cycle is if there is an edge where $i > j$, which would be contained in the A_2 subset, and the only way for A_2 to contain a cycle is if there is an edge where $i < j$, which would be contained in the A_1 subset. Thus, neither subset will contain a cycle and at least one of the two subsets will contain at least $|A|/2$ edges since $A = A_1 + A_2$. Thus, MDAS can be trivially approximated within a factor of 2 by taking the larger of the two subsets. □

b. **Theorem 4.2** The Maximum Directed Acyclic Subgraph (MDAS) problem does not have a PTAS unless $\mathbf{P = NP}$.

Proof: A known problem that does not have a PTAS is the Maximum Independent Set (MIS) problem. MIS does not have a PTAS for any graph with a maximum degree ≥ 3. We can do a linear reduction $MIS(3) \leq_L MDAS$: Given an undirected graph $G = (N, E)$ with maximum degree 3, we construct a directed graph $D = (V, A)$ where $V = \{(u_1, u_2) \mid u \in N\}$ and $A = \{(u_1, u_2) \mid u \in N\} \cup \{(u_2, v_1), (v_2, u_1) \mid (u, v) \in E\}$. We let $\alpha(G)$ denote the size of the maximum independent set of G and $\gamma(D)$ denote the number of edges of the maximum acyclic subgraph of D. From this, we have the following lemma:

Lemma 4.2.1 For all $\epsilon > 0$, if we are given an acyclic subgraph of D that has at least $(1 - (\epsilon/13))\gamma(D)$ edges, then we can compute in polynomial time an independent set G that has at least $(1 - \epsilon)\alpha(G)$ nodes.

Proof:
(4.2a) If I is an independent set of G, then $D' = (V, A')$ where $A' = \{(u_1, u_2) \mid u \in I\} \cup \{(u_2, v_1), (v_2, u_1) \mid (u, v) \in E\}$ is an acyclic subgraph of D. We can see that all edges $\{(u_2, v_1), (v_2, u_1) \mid (u, v) \in E\}$ do not create a cycle because as we have shown in Lemma 4a, a set of edges $\{(v_i, v_j) \mid (v_i, v_j) \in A, i > j\}$ do not create a cycle. We also know that the independent set I contains nodes that do not have any adjacent edges with each other. Thus, since A' contains all the linear reduction of edges of E, the only way to keep the edge set acyclic would be to only add the edges $\{(u_1, u_2) \mid u \in N\}$ where the nodes do not have any adjacent edges, which would be the independent set I and gives us $\{(u_1, u_2) \mid u \in I\}$. Thus, if I is an independent set of G, then A' is an acyclic subgraph of D.

(4.2b) Now we show that if H is an acyclic subgraph of D with h edges, we can then derive efficiently from H an independent set of G with at least $h - 2|E|$ nodes. As we have shown in **4.2a**, acyclic subgraphs of D includes all edges $\{(u_2, v_1), (v_2, u_1) \mid (u, v) \in E\}$. We

can see that each edge of E corresponds to two edges in A. If we took away all those edges from H, we would be left with edges $\{(u_1, u_2) \mid u \in N\}$ which we know are nodes that could create an independent set, as we have shown in **4.2a**. Thus, we can derive efficiently from H an independent set of G with at least $h - 2|E|$ nodes.

(4.2c) Lastly, since G has maximum degree 3, we can show that $\alpha(G) \geq |E|/6$. Without loss of generality, lets take a node $u \in N$ that has degree 3, which would imply 4 connected nodes. If we were to make all these nodes connected together, we would have a complete graph where each of the 4 nodes would have a maximum degree of 3. This complete graph has a 6 total edges. From this complete graph, we can only choose any one node as the maximum independent set $\alpha(G)$ as all nodes are adjacent to all other nodes. Thus, we can see that $\alpha(G) \geq |E|/6$.

Let us do the linear reduction from MIS to MDAS. From **4.2b** we know we can get an acyclic subgraph of D with $|I| + 2|E|$ edges. If $I = \alpha(G)$, then getting an acyclic subgraph D with $\alpha(G) + 2|E|$ would imply a maximum acyclic subgraph $\gamma(D)$. We can manipulate our conclusion from **4.2c** where $\alpha(G) \geq |E|/6 \implies |E| \leq 6 \cdot \alpha(G)$. Thus, we can say

$$\gamma(D) = \alpha(G) + 2|E| \leq \alpha(G) + 12 \cdot \alpha(G) = 13 \cdot \alpha(G)$$

Based on this L-reduction, it satisfies the first property that $OPT_{MDAS} \leq \alpha \cdot OPT_{MIS}$ where $\alpha = 13$. We can also see it satisfies the second property $|C_1 - OPT_{MIS}| = \beta \cdot |C_2 - OPT_{MDAS}|$ where $\beta = 1$. Combining both properties, we get the relative error:

$$\frac{|C_1 - OPT_{MIS}|}{OPT_{MIS}} \leq \frac{\alpha \cdot \beta \cdot |C_2 - OPT_{MDAS}|}{OPT_{MDAS}} \leq \alpha\beta\epsilon$$

Showing that the relative error of MIS is $13 \cdot \epsilon$ in relation to the relative error of MDAS being ϵ. Thus, if we wanted to express the relative error of MIS as just ϵ, that would mean that the relative error of MDAS would be $\epsilon/13$. Thus for all $\epsilon > 0$, if we are given an acyclic subgraph of D that has at least $(1 - (\epsilon/13))\gamma(D)$ edges, then we can compute in polynomial time an independent set G that has at least $(1 - \epsilon)\alpha(G)$ nodes, which proves the lemma. □

But because MIS does not have a PTAS as we have proven in problem 3, and since $MIS \leq_L MDAS$, that tells us that MDAS also does not have a PTAS unless $\mathbf{P = NP}$, thus the theorem is proved[**?**]. □

References

[1] Anari, N., and Gharan, S. O. Effective-resistance-reducing flows and asymmetric TSP. *In: Proceedings of FOCS* (2015).

[2] Asadpour, A., Goemans, M. X., Madry, A., Gharan, S. O., and Saberi, A. An $\mathcal{O}$(logn / log logn)-approximation algorithm for the asymmetric traveling salesman problem. *In: Proceedings of SODA* (2010), pages 379–389.

[3] Christofides, N. Worst-case analysis of a new heuristic for the traveling salesman problem. *Technical Report, DTIC Document* (1976).

[4] Frieze, A. M., Galbiati, G., and Maffioli, F. On the worse-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks* (1982), 12:23–39.

[5] Gharan, S. O., and Saberi, A. The asymmetric traveling salesman problem on graphs with bounded genus. *In: Proceedings of SODA* (2011), pages 967–975.

[6] Gharan, S. O., Saberi, A., and Singh, M. A randomized rounding approach to the traveling salesman problem. *In: Proceedings of FOCS* (2011).

[7] Karpinski, M., Lampis, M., and Schmied, R. New approximability bounds for TSP. *J. Comput. Syst. Sci.* (2015), 81(8):1665–1667.

[8] Papadimitriou, C. H., and Yannakakis, M. The traveling salesman problem with distances one and two. *Math. Oper. Res.* (1993), 18(1):1–11.

[9] Svensson, O. Approximating ATSP by relaxing connectivity. *In: Proceedings of FOCS* (2015).

[10] Svensson, O., Tarnawski, J., and Végh, L. A. Constant factor approximation for ATSP with two edge weights. *In: Proceedings of IPCO* (2016), pages 226–237.