LP-based solution methods for the asymmetric TSP

Vardges Melkonian

Department of Mathematics, Ohio Universtiy, Athens, Ohio 45701, USA
vardges@math.ohiou.edu

Abstract
We consider an LP relaxation for ATSP. We introduce concepts of high-value and high-flow cycles in LP basic solutions and show that the existence of this kind of cycles would lead to constant-factor approximation algorithms for ATSP. The existence of high-flow cycles is motivated by computational results and theoretical observations.
(*Keywords:* TSP; Linear programming; Network flows; Approximation algorithm)

1. Introduction

In Traveling Salesman problem (**TSP**), it is required to find a minimum cost Hamiltonian tour, that is, a cycle passing through each node exactly once. A nice collection of papers tracing the history and research on the problem can be found in Lawler et al.[7]. Most of the research on TSP algorithms has concentrated on the undirected version of TSP. The best approximation factor for the case when the arc costs satisfy the triangle inequality in undirected networks is 1.5 and was obtained by Christofides ([1]). Far less research is done on the version of TSP in directed graphs to which we will refer as Asymmetric TSP (**ATSP**). The best approximation ratio of $O(\log n)$ was achieved first by Frieze et al. [2] and later by Kleinberg and Williamson [6]. The current best approximation algorithm for ATSP is by Kaplan et al. [5] and achieves approximation ratio $0.842 \log n$. However the best-known lower bound on the approximation factor is only 117/116 [9]. This low lower bound and the discrepancy between the best approximation factors for symmetric and asymmetric cases give hopes that there should be an algorithm with a constant approximation factor for ATSP. In this paper we will explore one direction that ultimately could lead to such an algorithm.

The LP rounding technique of Jain [4] provided a new powerful tool for designing approximation algorithms for network design problems. By proving the existence of high-value variables in basic solutions of an LP relaxation, Jain gave a 2-approximation algorithm for the generalized Steiner tree problem. Melkonian and Tardos [8] extended Jain's technique to obtain an approximation algorithm for a class of directed network design problems. In this paper, we will explore the possibilities of applying Jain's technique to ATSP. We suggest an algorithm which is based on solving the LP relaxation of ATSP. We were not able to prove an approximation factor for the algorithm but our conjecture is that it is a 2-approximation algorithm. We will give some arguments and examples supporting the conjecture.

The paper is structured as follows. In section 2, it is shown that any basic solution of an LP relaxation of ATSP has a variable with value at least 1/2. In section 3, the concepts of high-value and high-flow cycles are introduced. It is shown that the existence of this kind of cycles in the basic solutions would lead to constant factor approximation algorithms for ATSP. The existence of high-flow cycles is motivated by computational results and theoretical observations.

2. High value variables in basic solutions

We consider the case of ATSP when the arc costs satisfy the triangle inequality. In that case the problem is equivalent to requiring the tour to visit every node at least (instead of exactly) once. Then the ATSP can be given by the integer program (IP_{ATSP}):

$$\begin{aligned} \min \quad & \textstyle\sum_{e\in E} c_e x_e && && (1)\\ \text{s.t.} \quad & \textstyle\sum_{e\in\delta^+(S)} x_e \geq 1, && \text{for each } S \subset V, && (2)\\ & \textstyle\sum_{i:i\to j\in E} x_{ij} = \sum_{k:j\to k\in E} x_{jk}, && \text{for each } j \in V, && (3)\\ & x_e \quad binary, && \text{for each } e \in E. && (4) \end{aligned}$$

Any solution satisfying (2) and (3) is a tour that visits each node at least once. Thus, by traversing the optimal output tour of (IP_{ATSP}) and shortcuting the nodes that have been previously visited, we will get a TSP tour T of no greater cost (due to triangle inequlity). Since any TSP tour is a solution to (IP_{ATSP}), T is an optimal tour.

The linear programming relaxation (LP_{ATSP}) of the integer program (IP_{ATSP}) is obtained by replacing the integer requirements of x_e with $x_e \geq 0$. We note that though (LP_{ATSP}) has an exponential number of constraints, it can be solved in polynomial time by designing a polynomial-time separation oracle [3] or by reformulation as a polynomial-size LP using auxiliary variables [10]. Vempala and Yannakakis, aiming to obtain an LP-rounding approximation algorithm for ATSP with triangle inequality, gave an analysis of basic solutions of

(LP_{ATSP}) in [11]. Their arguments are similar to Jain's ([4]) arguments. The main result they get in [11] is the following property of basic solutions of (LP_{ATSP}):

Theorem 1 *[11] The number of non-zero edges in a basic solution is at most* $3|V| - 2$.

This property implies the existence of a high-value variable in any basic solution:

Theorem 2 *In any basic solution of* (LP_{ATSP}) *there is at least one variable with value at least* $1/2$.

Proof: Assume the opposite: all the variables have values strictly less than 1/2. Cut constraints (2) imply that for any node $i \in V$, $\sum_{j \neq i} x_{ij} \geq 1$. Thus, for any node, there should be at least 3 non-zero arcs leaving it. This makes the total number of non-zero arcs at least $3|V|$ which contradicts Theorem 1. □

Unfortunately the existence of high-value variables in basic solutions of ATSP does not lead to iterative rounding algorithms with constant approximation factors as it does in the case of other network design problems considered in [4] and [8]. In the case of those problems, by including the high-value arc in the solution, it is possible to get a residual instance of LP which has the same structure as the original one; and thus there is a high-value variable in the solution of the residual instance which allows to iterate. It is not known how to achieve that in the case of ATSP. Particularly, it is not clear how to do the reduction so that the balance constraints which are essential for showing the existence of a high-value variable still hold in the residual instance. Based on this discussion we think that to get an LP rounding algorithm with a good approximation factor it is not enough to have just one high value variable. In the next section we discuss what could be a possible alternative to that.

3. High-value and high-flow cycles

The main difference of ATSP from connectivity problems is that the nodes not only should be connected to each other but also the connection should be realized by the means of edge-disjoint directed cycles. Thus, the directed cycles are supposed to play an important role for designing algorithms for ATSP, and particularly when designing LP rounding algorithms.

High-value cycles

Suppose in any basic solution of (LP_{ATSP}) we had a directed cycle C with at least $1/k$-value on each of the C-arcs for some integer k. Then it would be possible to create a residual instance by the following reduction: (i) include all C-arcs in the solution; (ii) keep all balance constraints (3); (iii) keep the cut constraint corresponding to $S \subset V$ if either $\{\text{all C-nodes}\} \subseteq S$ or $\{\text{all C-nodes}\} \cap S = \emptyset$.

The advantage of this kind of reduction is that it keeps the balance constraints satisfied; and in that case it is more likely that arguments similar to [11] could prove the existence of a high-value variable in the reduced instance. But the existence of a $1/k$-cycle for a fixed integer k is an open question. In our computational results we had several instances with no 1/2-cycles; for example, see figure 1 (however there were 1/3-cycles in all instances).

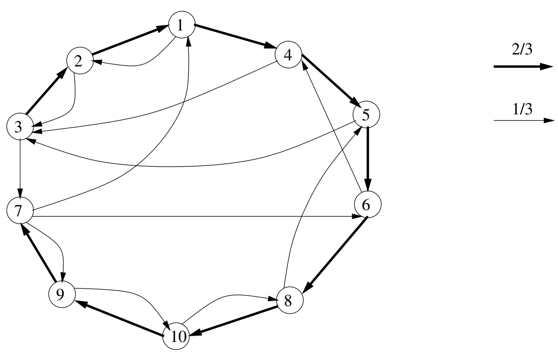

Figure 1: High-flow cycle

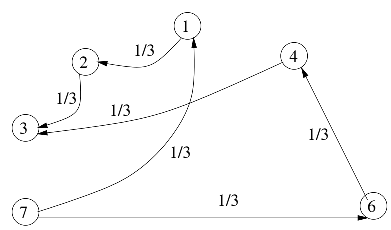

Figure 2: Disjoint paths

So our observation is that a high-value cycle is more likely to lead to an iterative rounding algorithm (with a constant approximation factor) than a high-value variable, but the existence of such a cycle is questionable. Then what could be a substitute for the cycle? The triangle inequality allows us to utilize so-called high-flow cycles which are relaxed versions of high-value cycles and are introduced next.

High-flow cycles

To introduce the concept of high-flow cycles consider the example of figure 1 again. All the arcs but $7 \to 3$ on the cycle $C = 3 \to 2 \to 1 \to 4 \to 5 \to 6 \to 8 \to 10 \to 9 \to 7 \to 3$ have values 2/3. But note that using the 1/3-arcs we can send a flow of 2/3 from node 7 to node 3 by disjoint paths $P_1 = 7 \to 6 \to 4 \to 3$ and $P_2 = 7 \to 1 \to 2 \to 3$ (see figure 2). By analyzing the cost of cycle C we get that it is at most 1.5 times more expensive than the optimal solution.

$$Cost(C) = c_{73} + \sum_{(i,j)\in C:(i,j)\neq(7,3)} c_{ij} = \frac{3}{2}\left(\frac{1}{3}c_{73} + \frac{1}{3}c_{73} + \sum_{(i,j)\in C:(i,j)\neq(7,3)} \frac{2}{3}c_{ij}\right) \leq$$

$$\frac{3}{2}\left(\frac{1}{3}cost(P_1) + \frac{1}{3}cost(P_2) + \sum_{(i,j)\in C:(i,j)\neq(7,3)} \frac{2}{3}c_{ij}\right) \leq \frac{3}{2}Cost(LP_{ATSP}) \;\leq\; \frac{3}{2}Cost(IP_{ATSP})$$

Note that the first inequality is obtained by applying the triangle inequality. Thus, though cycle C does not have high values on all its arcs, it is a high-flow cycle, and this allows to show that C has a cost not far from the optimum.

Now we give the formal definition of high-flow cycles. As above, given a fractional solution to (LP_{ATSP}), we will consider the fractional values as flows between the nodes.

Definition 1 *A directed cycle C is called α-flow cycle for some $0 \leq \alpha \leq 1$ if (i) for any arc $i \to j$ on C, there is flow of value at least α from i to j through directed paths; (ii) no two arcs from C use the same portion of flow for realizing their α-flows.*

Note that the fractional flow of any arc can be split to realize the α-flows of different arcs lying on cycle C. The analysis for the example of figure 1 can be generalized to any high-flow Hamiltonian cycle.

Lemma 1 *The cost of an α-flow Hamiltonian cycle C is at most $1/\alpha$ times the optimum.*

Proof: Let P_{ij} be the set of the directed paths realizing the α-flow for arc $i \to j \in C$. For $P \in P_{ij}$, let α_P be the flow on P which is contributing to the total α-flow from i to j. Then

$$Cost(C) = \sum_{(i,j)\in C} c_{ij} \;=\; \frac{1}{\alpha}\sum_{(i,j)\in C} \alpha \cdot c_{ij} = \frac{1}{\alpha}\sum_{(i,j)\in C}\sum_{P\in P_{ij}} \alpha_P \cdot c_{ij} \;\leq\; \frac{1}{\alpha}\sum_{(i,j)\in C}\sum_{P\in P_{ij}} \alpha_P \cdot Cost(P)$$

$$\leq \frac{1}{\alpha}Cost(LP_{ATSP}) \;\leq\; \frac{1}{\alpha}Cost(IP_{ATSP})$$

The first inequality follows from the triangle inequality. The second inequality follows from the fact that we might be using a part of the fractional solution values for realizing the α-flows without any overlap. $\square$

Experimental results

Lemma 1 implies that the existence of an α-flow Hamiltonian cycle in the fractional solution would give a $1/\alpha$-approximation algorithm for ATSP. Does this kind of cycle always exist for some constant α? In our computational experiments we always could get a 1/2-flow Hamiltonian cycle. The experiments were conducted on randomly created instances and on some instances from the public library TSPLIB. We already discussed in details the solution of one of the randomly created instances (figure 1). In that example we were able to find a 2/3-flow Hamiltonian cycle. However, for many other networks including the bigger ones from TSPLIB the best we could find was a 1/2-flow Hamiltonian cycle.

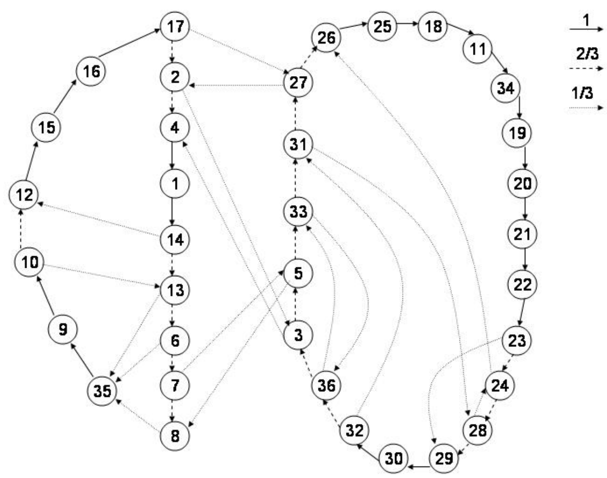

Figure 3: Fractional solution to TSPLIB instance

The fractional solution of one of these instances (data file *ftv35.atsp* from TSPLIB) is given in figure 3. In this solution, there are two high-value cycles which cover the whole node set. To get a high-flow Hamiltonian cycle, we need to concatenate these two cycles. Here is one way to do that: (*i*) Start from node 4 of the left cycle; (*ii*) travel all the nodes of the left cycle till node 2; (*iii*) switch to node 3 of the right cycle; (*iv*) travel all the nodes of the right cycle till node 36; (*v*) switch back to node 4 of the left cycle to complete the tour. Steps (*iii*) and (*v*) can be realized using directed paths to make the flow values between the cycles 1/2. 1/2-flows are realized the following way at the switching points. From 2 to 3: send 1/3 directly through the arc $2 \to 3$; send 1/6 through the path $2 \to 4 \to 1 \to 14 \to 12 \to 15 \to 16 \to 17 \to 27 \to ... \to 36 \to 3$. From 36 to 4: send 1/3 through the path $36 \to 3 \to 4$; send 1/6 through the path $36 \to 33 \to 31 \to 27 \to 2 \to 4$.

Note that we keep at least 1/2 on the arcs of the left and right cycles; so the result is a 1/2-flow Hamiltonian cycle.

Below we summarize the results of our experiments. The experiments were run on 12 TSPLIB instances. In addition, we randomly created 50 instances (30 nodes). A 1/2-flow Hamiltonian cycle was found for all those instances. We divide instances into several groups when reporting the results. The groups are (A) an arc-missing high-flow cycle (as in figure 1); (B) two high-value cycles (as in figure 3); (C) LP-relaxation returns an integer solution; (D) LP-relaxation returns an α-value closed trail where $\alpha \geq 1/2$. In this last case each node might be visited more than once but a α-flow Hamiltonian cycle can be obtained by taking shortcuts when necessary. Particularly, the output of many instances was a half-integral solution. These instances belong to group (D) because every arc with value 1 can be split into parallel arcs with value 1/2 each. The results are summarized in table 1.

Table 1: Summary of experimental results

groups	TSPLIB instances	random instances
A	ftv44, ftv47, ry48p	6 instances
B	ftv35, ftv38	15 instances
C	br17, ftv33, ft53	14 instances
D	ftv55, ftv64, ftv70, p43	15 instances

Finally, we note that a flow-based LP formulation of ATSP was implemented for our experiments (see [10] for a detailed discussion and references). This formulation has polynomial number of constraints and variables. The constraints here directly express the fact that there should be a directed path between any pair of nodes. We choose an arbitrary node s and call it *root*. In order the network to be strongly connected, there should be directed paths from any node to s and from s to any node. The IP formulation which accomplishes this is given by (5)-(11).

$$\min \quad \sum_{e\in E} c_e x_e \tag{5}$$

$$\text{s.t.} \quad \sum_{e\in\delta^+(u)} f_e^{(s,t)} - \sum_{e\in\delta^-(u)} f_e^{(s,t)} = \lambda_u \qquad \text{for each } u \in V, t \in V (t \neq s) \tag{6}$$

$$\sum_{e\in\delta^+(u)} f_e^{(t,s)} - \sum_{e\in\delta^-(u)} f_e^{(t,s)} = -\lambda_u \qquad \text{for each } u \in V, t \in V (t \neq s) \tag{7}$$

$$x_e \geq f_e^{(s,t)} \qquad \text{for each } e \in E, t \in V (t \neq s) \tag{8}$$

$$x_e \geq f_e^{(t,s)} \qquad \text{for each } e \in E, t \in V (t \neq s) \tag{9}$$

$$\sum_{i:i\to j\in E} x_{ij} = \sum_{k:j\to k\in E} x_{jk}, \qquad \text{for each } j \in V \tag{10}$$

$$f_e^{(s,t)}, f_e^{(t,s)},\ x_e\ \mathit{binary} \qquad \text{for each } e \in E, t \in V (t \neq s). \tag{11}$$

The 0-1 variables x_e represent whether or not arc e is included in the solution; 0-1 variables $f_e^{(s,t)}$ ($f_e^{(t,s)}$) represent whether arc e is on the directed path $s \rightsquigarrow t$ ($s \rightsquigarrow t$) in the solution. λ_u is 1 if $u = s$; -1 if $u = t$; 0 otherwise. The conservation-of-flow constraints (6) and (7) guarantee that there are directed paths between any two nodes in terms of f variables.

The existence of high-flow Hamiltonian cycles

Below we show that for groups (A) and (B) discussed in the experimental results we can get a high-flow Hamiltonian cycle without finding the particular flow that realizes it.

Lemma 2 *Let x be a feasible solution to (LP_{ATSP}). Suppose C is a Hamiltonian cycle such that we could find x-flows of value at least α through disjoint paths for any C-arc except $s \to t$. Then C is an α-flow Hamiltonian cycle.*

Proof: Let $P = C - s \to t$ be the directed path on which the disjoint x-flows are at least α. Let y be the residual flow after subtracting α from the flows of all P-arcs. We need to show that the y-flow from s to t is at least α. Based on the *min cut - max flow theorem* it is enough to show that the minimum (s,t)-cut has capacity at least α. Let $(S, \bar{S})$ be any (s,t)-cut (see figure 4). Let $\delta^+(S)$ and $\delta^-(S)$ be the sets of arcs correspondingly leaving and entering S.

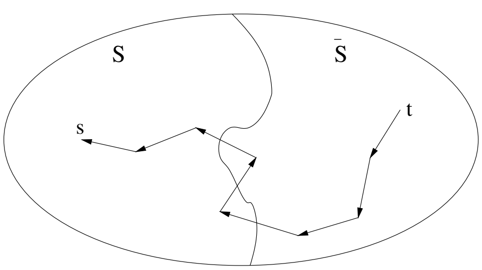

Figure 4: (s,t)-cut in Lemma 2

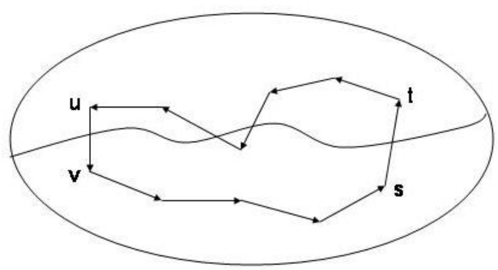

Figure 5: (s,t)-cut in Lemma 3

We can get a new valid constraint for (LP_{ATSP}) by taking the sum of balance constraints (3) over all $j \in S$. All those x_{ij} such that $i, j \in S$ will be both in left and right hand sides of the new constraint, and so can be cancelled. Thus, any feasible solution satisfies the following equality:

$$\sum_{e \in \delta^+(S)} x_e = \sum_{e \in \delta^-(S)} x_e \tag{12}$$

Note that by subtracting α from x-flow of each $\delta^+(S) \cap P$-arc, we subtract α more from $\sum_{e\in\delta^+(S)} x_e$ than from $\sum_{e\in\delta^-(S)} x_e$. Similarly, by subtracting α from x-flow of each $\delta^-(S) \cap P$-arc, we subtract α more from $\sum_{e\in\delta^-(S)} x_e$ than from $\sum_{e\in\delta^+(S)} x_e$. Summarizing,

$$\sum_{e\in\delta^+(S)} y_e - \sum_{e\in\delta^-(S)} y_e = \sum_{e\in\delta^+(S)} x_e - \sum_{e\in\delta^-(S)} x_e - \alpha \cdot |\delta^+(S) \cap P| + \alpha \cdot |\delta^-(S) \cap P| = \alpha$$

The last equality is based on (12) and that $|\delta^-(S) \cap P| = |\delta^+(S) \cap P| + 1$. Thus, the capacity of $(S, \bar{S})$ in terms of y-flow is $\sum_{e\in\delta^+(S)} y_e = \alpha + \sum_{e\in\delta^-(S)} y_e \geq \alpha$ □

Note that the case of figure 1 is the special case of Lemma 2 when the *x-value* on each C-arc except $s \to t$ is at least α. We needed a more general statement in the lemma for exploring the next case.

Lemma 3 *Suppose C is a Hamiltonian cycle such that every arc except $s \to t$ and $u \to v$ has x-value at least $\alpha \geq 0.5$; if we subtract α from all those high-value arcs, $s \to t$ would still have flow value $\geq 1 - \alpha$. Then*

1) After subtracting 0.5 from the high-value arcs, $s \to t$ still has flow value ≥ 0.5;

2) C is a 0.5-flow Hamiltonian cycle.

Proof: Note that if we show (1) then (2) simply follows from Lemma 2. (1) is proved using *min cut - max flow* arguments as in Lemma 2. Subtract 0.5 from the high-value C-arcs. Take any (s,t)-cut (see figure 5). If no high-value C-arc crosses the cut then its capacity should be ≥ 1 (based on the cut consraint (2)). If a high-value C-arc crosses the cut then we have a residual capacity $\geq \alpha - 0.5$ from that arc; in addition $s \to t$ still has flow value $\geq 1 - \alpha$. Then the total capacity of the cut is $\geq (1 - \alpha) + (\alpha - 0.5) = 0.5$. Based on the *min cut - max flow theorem*, the maximum flow from s to t is ≥ 0.5. □

Lemma 4 *Suppose C_1 and C_2 are two node-disjoint cycles that cover the whole node set. All the arcs on C_1 and C_2 have x-values at least $\alpha \geq 0.5$. There is an arc from $s \in C_1$ to $t \in C_2$ with x-value at least $1 - \alpha$. Then it is possible to concatinate C_1 and C_2 to get a 0.5-flow Hamiltonian cycle.*

Proof: Suppose $s \to u \in C_1$, $v \to t \in C_2$. Then $C = s \to t \overset{C_2}{\rightsquigarrow} v \to u \overset{C_1}{\rightsquigarrow} s$ is a 0.5-flow Hamiltonian cycle based on Lemma 3. □

Based on Lemmas 2, 4 and our experimental results we have the following conjecture.

Conjecture 1 *In any basic solution of (LP_{ATSP}) there is a $1/2$-flow Hamiltonian cycle.*

If this conjecture turns out to be true then based on Lemma 1 the following procedure would lead to a 2-approximation algorithm: (*i*) Solve (LP_{ATSP}); (*ii*) Find a 1/2-flow Hamiltonian cycle C; (*iii*) Output C as a solution to (IP_{ATSP}).

Summarizing, here are some interesting open questions about (LP_{ATSP}). (1) Is there a high-value (high-flow) cycle in any basic solution and does it lead to a rounding algorithm? (2) Is there a high-flow Hamiltonian cycle in any basic solution, and if yes how can it be found efficiently? These questions could also be raised for related network design problems where it is important to find cycles.

References

[1] N. Christofides. *Worst-case analysis of a new heuristic for the traveling salesman problem.* Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, PA, 1976.

[2] A. Frieze, G. Galbiati, and F. Maffioli. *On the worst-case performance of some algorithms for the asymmetric traveling salesman problem.* Networks 12, pages 23–39, 1982.

[3] M. Grotschel, L. Lovasz, and A. Schrijver. *Geometric algorithms and combinatorial optimization.* Springer-Verlag, 1988.

[4] K. Jain. *A factor 2 approximation algorithm for the generalized Steiner network problem.* Combinatorica 21, pages 39–60, 2001.

[5] H. Kaplan, M. Lewenstein, N. Shafrir, and M. Sviridenko. *Approximation algorithms for Asymmetric TSP by decomposing directed regular multigraphs.* In Proceedings of the 44th Annual Symposium on the Foundation of Computer Science, pages 56–67, 2003.

[6] J. Kleinberg and D. Williamson. *A new $O(\log n)$-approximation algorithm for ATSP.* In Lecture Notes on Approximation Algorithms, Fall 1998. http://www.almaden.ibm.com/cs/people/dpw/Courses/cornell.ps, pages 125–127. IBM Research Report RC 21409, February 1999.

[7] E. Lawler, J. Lenstra, A. Rinnooy Kan, and D. Shmoys. *The Traveling Salesman Problem.* John Wiley, 1985.

[8] V. Melkonian, E. Tardos, *Algorithms for a network design problem with crossing supermodular demands.* Networks 43(4), pages 256–265, 2004.

[9] C. Papadimitriou and S. Vempala. *On the approximability of the traveling salesman problem.* Proceedings of the 32nd ACM Symposium on the theory of computing (STOC '00), Portland, 2000.

[10] W. Pulleyblank. *Polyhedral combinatorics.* In Optimization, Volume 1 of Handbooks in Operations Research, pages 371-446, Elsevier, 1989.

[11] S. Vempala and M. Yannakakis. *A convex relaxation for the Asymmetric TSP.* Proceedings of the 10th ACM-SIAM Symposium on Discrete Algorithms, Baltimore, 1999.