A 5/8 Approximation Algorithm for the Asymmetric Maximum TSP

Moshe Lewenstein* Maxim Sviridenko*

Abstract

The Asymmetric Maximum Travelling Salesperson problem, also known as the Taxicab Ripoff problem, is the problem of finding a maximally weighted tour in a complete asymmetric graph with non-negative weights.

We propose a polynomial time approximation algorithm for the problem with a $\frac{5}{8}$ approximation guarantee. This (1) improves upon the approximation factors of previous results and (2) presents a simpler solution to the previously fairly involved algorithms. Our solution uses a simple LP formulation. Previous solutions were combinatorial. We make use of the LP in a novel manner and strengthen the Path-Coloring method originally proposed in [13].

1 Introduction

Problem Formulation In the maximum asymmetric travelling salesman problem, Max TSP for short, we are given a complete weighted directed graph $G = (V, E, w)$ with nonnegative edge weights $w_{uv} \geq 0, (u,v) \in E$ and need to find a closed tour of maximum weight visiting all vertices exactly once.

Motivation The minimization variant of this problem is one of the most studied and well-known optimization problems and has many applications [12]. The Max TSP is also well-studied and a well-motivated optimization problem. We shall shortly mention several applications.

Since Max TSP is an NP-hard optimization problem it is desirable to design approximation algorithms for this problem with good performance guarantees. We say that an algorithm for a maximization problem has performance guarantee $\rho \leq 1$ if it always delivers a solution with value at least ρ times the value of the optimal solution. (For a minimization problem $\rho \geq 1$ and the algorithm should always deliver a solution with value at most ρ times the optimal value.)

The following applications motivate Max TSP.

1. Shortest superstring problem: Given a collection of strings $s_1, ..., s_n$ we seek the shortest possible string S such that every string in the collection is a substring of S, i.e. $\forall i, S = S' s_i S''$. This

*IBM T.J. Watson Research Center, Yorktown Heights, 10598; {moshe, sviri}@watson.ibm.com

problem arises in DNA sequencing and has applications to data compression as well. Kosaraju, Park and Stein [13] noted that there is an implicit reduction from shortest superstring to Max TSP in the proof of [5]. An ever tighter reduction is given by Breslauer, Jiang and Jiang [6]. Specifically, they show that a ρ approximation factor for Max TSP implies a $(3.5 - 1.5) \cdot \rho$ approximation for shortest superstring.

2. Maximal compression problem: This problem is the sibling of the shortest superstring problem. Given a collection of strings $s_1, ..., s_n$ we seek a string S such that (1) every string in the collection is a substring of S and (2) maximizes $\sum_i |s_i| - |S|$. The shortest superstring's optimal solution is equal to the optimal solution of this problem, but the approximate solutions can differ greatly approximation-wise. This problem which arises in various data compression problems was analyzed by Tarhio and Ukkonen [18] and by Turner [19]. In this setting the vertices represent strings and an edges between two "strings" is weighted with the amount of maximum overlap between these strings. The optimal compression is equivalent to the weight of the maximal hamiltonian path. With the creation of a special vertex representing the start and end of the hamiltonian cycle, the maximal compression problem is equivalent to Max TSP on this graph. Hence a ρ approximation for Max TSP implies a ρ approximation for the maximal compression problem.

3. Minimum asymmetric {1,2}-TSP: Introduced by Papadimitriou and Yannakakis [16], this problem is that of the classical minimum TSP in the asymmetric case when the edge weights are 1 or 2. The problem can easily be transformed into the maximum variant, where the weights of 2 are replaced by 0. It is easily verifiable that a ρ factor approximation for Max TSP implies a $2 - \rho$ approximation algorithm for minimum asymmetric {1,2}-TSP.

Previous Results: As we already noted the Max TSP is an NP-hard optimization problem. Moreover, Papadimitriou and Yannakakis [16] proved that this problem is MAX SNP-hard and therefore, we cannot obtain a polynomial time approximation scheme for this problem unless P=NP. This result was refined by Engebretsen [7]; his result implies that there is no $(\frac{2803}{2804} + \varepsilon)$ approximation algorithm for Max TSP for any $\varepsilon > 0$ unless $P = NP$ (note that he proved this result for minimum asymmetric $\{1, 2\}$-TSP). Recently, Engebretsen and Karpinsky [8] improved the negative result to $319/320 + \varepsilon$. The first polynomial time algorithm with proven performance guarantee for this problem is due to Fisher, Nemhauser and Wolsey [10]. They noticed that the algorithm which finds a minimum cycle cover on the input graph, deletes the cheapest edge in each cycle and patches all cycles together has performance guarantee 1/2. Other approximation algorithms which improve the approximation factors appear in [14, 13, 2]. The best known result to date was an 8/13 factor [2].

In the case when all weights are either 0 or 1 Vishwanathan [20] obtained a 7/12 approximation and Bläser and Siebert [4] obtained a 2/3 approximation algorithm exploiting an approach due to Papadimitriou and Yannakakis [16]. Many other variants of approximation algorithms for Max TSP appear in the literature. A good survey on Max TSP, also containing many results that appeared in the Russian literature, is [1].

Our results: In this paper we present a new algorithm which achieved a 5/8 approximation factor for Maximum Asymmetric TSP. The algorithm uses a simple LP formulation of a cycle cover which does not contain 2-cycles. This is a very weak variant of the full classical *subtour elimination* constraints. We solve this LP and transform it into a collection of cycle covers with useful properties. We then remove edges to satisfy a generalized version of the Path-coloring Lemma due to Kosaraju, Park and Stein [13]. This yields many hamiltonian cycles of which one must contain weight at least 5/8 of the maximal weight hamiltonian cycle.

Our result immediately implies a better result for the maximal compression problem. It also implies an approximation algorithm for the minimum $\{1,2\}$-TSP with performance guarantee 11/8. Recently, a better performance guarantee of 4/3 was shown for this special case [4]. Another implication is a simple $(41/16 = 2.5 + 1/16)$ approximation algorithm for the shortest superstring problem. The best known approximation algorithm for this problem has performance guarantee 2.5 [17], yet is quite complicated.

2 Algorithm and Analysis

In this section we present our $\frac{5}{8}$ approximation algorithm. We will assume that the input graph G has an even number of vertices. For the case of an odd number of vertices we show a reduction to the even number case in Section 3. This assumption is made in order to be able to use perfect matchings in our algorithm (more specifically, maximum matchings in a complete graph).

We begin with some preliminary notation. For a vertex set V let $K(V)$ denote the edge set $K(V) = (V \times V) \setminus \{(v,v) \mid v \in V\}$. We will consider weighted directed graphs $G = (V, K(V), w)$, where $w : K(V) \to \mathcal{Q}_{\geq 0}$. With w_{uv} we shorthand $w(u,v)$. The optimal solution to the Max TSP problem will be denoted *opt*.

A *cycle cover* C for a graph is a set of cycles such that each vertex participates in exactly one cycle. The weight of a cycle cover, denoted $w(C)$, is the sum of the weights of its edges. A directed multigraph is said to be *d-regular* if the indegree, and the outdegree, of each vertex is equal to d.

A set of edges E is called *k-path-colorable* if the set of edges can be partitioned into k collections of vertex disjoint paths. The collections $\langle E_1, ..., E_k \rangle$ are said to *color-partition* E. Let E be a set of edges that is 1-path-colorable. We say that an edge $e \notin E$ is *path-compatible* with E if $\{e\} \cup E$ is 1-path-colorable.

2.1 Generic Algorithm and Previous Solutions

The following generic algorithm first appeared in [10] and was also used in [13, 2, 20].

GENERIC ALGORITHM

Step 1. Compute a maximum-weight cycle cover C for G.
Step 2. Use C to obtain a set P of vertex-disjoint paths.
Step 3. Construct a tour by patching together the paths of P.

In [10] the algorithm uses a simple method. To obtain the set P in step 2 the lightest edge in each cycle was discarded, yielding a $\frac{1}{2}$ approximation factor. In [13, 2, 20] better approximations were achieved by slightly adapting the graph and finding a maximum matching M along with the maximum cycle cover. The main idea is to discard some fraction of the edges to obtain a 2-path-colorable set of edges.

If $C \cup M$ was 2-path-colorable with color-partition $\langle E_1, E_2 \rangle$ then choosing the set of edges with the larger weight would yield a solution $\geq \frac{3}{4}$ *w(opt)* since $w(M) + w(C) \geq 1.5w(opt)$.

Since edges need to be discarded the focus of the algorithms is on the fraction discarded. The results in [13, 2] are achieved by a tradeoff between several applications (three and four respectively) of the algorithms with different discarding schemes. All discarding schemes rely on the following graph coloring property.

Lemma 1 ([13, 3]) *(KPS Lemma) Let G be a directed multigraph such that (1) each vertex has indegree at most 2, outdegree at most 2, and total degree at most 3, and (2) the graph does not contain any 2-cycles. Then the edges of G are 2-path-colorable.*

It needs to be pointed out that in [13] only the formulation of the Lemma is contained. A partial proof appears in [2] and a full proof appears in [3].

For our solution we will need a more general version of the KPS-Lemma allowing us to color certain graphs with 2-cycles. The 2-cycles that remain undesirable are those that are on a cycle other than this 2-cycle. Namely, for a cycle $\langle v_1, ..., v_k, v_1 \rangle$ in a directed multigraph G a *back-edge* is an edge (v_i, v_{i-1}), where $i-1$ is k if i is 1. Note that for the case $k = 2$, (v_2, v_1) of the cycle is not called a back-edge, yet a different (v_2, v_1) (which can happen in a multigraph) is called a back-edge, see the lefthand figure in Fig. 1. The following lemma is the desired generalization.

Lemma 2 *(Path-Coloring Lemma) Let $G = (V, E)$ be a directed multigraph such that (1) each vertex has indegree at most 2, outdegree at most 2, and total degree at most 3, and (2) there are no back-edges on any cycle in G. Then E is 2-path-colorable.*

Proof: The Lemma will follow from the KPS Lemma if we eliminate all 2-cycles from G. To this end one 2-cycle will be eliminated while maintaining the properties of the Lemma, not creating any new 2-cycles, and assuring that the eliminated 2-cycle can be 2-path-colored in compatibility with the rest of the graph. Hence, one can eliminate all 2-cycles and the result will follow.

Let $(u, v), (v, u) \in E$ be a 2-cycle in G. Since total degree of u is at most 3 there is at most one other edge, e_1, incident on u, either (x, u) or (u, x) for some vertex x. Vertex x cannot be equal to v because of property (2). Likewise, there is at most one edge, e_2, (v, w) or (w, v). There are four cases.

Case 1. u and v both have total degree 2: this 2-cycle is isolated and is 2-path-colorable independent of $G \setminus \{u, v\}$.

Case 2. Only one of u and v has total degree 3. This case is a special case of case 4.

In the next two cases both u and v have total degree 3.

Case 3. One of $\{(u,v),(v,u)\}$ is path-compatible with both e_1 and e_2. Assume, w.l.o.g. that $e_1 = (x,u)$ and that $e_2 = (v,w)$. Set $V' = V \setminus \{u,v\}$. The subgraph induced by V', G', has exactly four edges fewer than G. Add edge (x,w) to G'. It is straightforward to check that the properties are maintained. Moreover, by property (2), no new 2-cycle is created. For a 2-path-coloring of G, let $\langle E_1, E_2 \rangle$ be a 2-path-coloring of G' such that $(x,w) \in E_1$. Then $\langle E_1 \cup \{(x,u),(u,v),(v,w)\} \setminus \{x,w\}, E_2 \cup \{(v,u)\}\rangle$ can be easily verified to be a 2-path-coloring of G.

Case 4. Each of (u,v) and (v,u) is path-compatible with exactly one of e_1 and e_2. Contract u and v into one vertex z discarding edges (u,v) and (v,u). All properties are maintained and a new 2-cycle is not created since z has either indegree 0 or outdegree 0. A 2-path-coloring in the contracted graph can be extended by assigning (u,v) and (v,u) the color of their path-compatible edge e_1 or e_2. □

2.2 LP for Cycle Cover and Algorithm Outline

In our solution we maintain the generic structure of the algorithm but propose a novel way of obtaining the vertex-disjoint paths. Rather than finding a cycle cover using combinatorial methods we formulate an LP for cycle cover and add an additional constraint. In the IP corresponding to this LP the constraint forbids two-cycles.

This LP is then solved and used to construct a collection of cycle covers. To each cycle cover in the collection we add a maximum matching, as in [13, 2]. We then transform this collection of graphs, by discarding and moving edges, into a collection of graphs that are 2-path-colorable. The collection of vertex-disjoint paths with largest weight is chosen from amongst all graphs. Our discarding scheme is different from previous schemes, in particular we will be transferring edges between graphs in the collection.

The LP is formulated as following.

LP FOR CYCLE COVER WITH TWO-CYCLE CONSTRAINT

Max $\sum_{(u,v)\in K(V)} w_{uv} x_{uv}$ subject to

$\sum_u x_{uv} = 1,$	$\forall v$	(indegree constraints)
$\sum_v x_{uv} = 1,$	$\forall u$	(outdegree constraints)
$x_{uv} + x_{vu} \le 1,$	$\forall u \ne v$	(two-cycle constraints)
$x_{uv} \ge 0,$	$\forall u \ne v$	(non-negativity constraints)

Let $\{x^*_{uv}\}_{uv \in K(V)}$ be a solution for the LP. Set D to be the minimal integer such that for all $(u,v) \in K(V)$, $D \cdot x^*_{uv}$ is integral. Denote by $k \cdot (u,v)$ the multiset containing k copies of the edge (u,v). Define the weighted multigraph $D \cdot G = (V, \hat{E}, w)$ where $\hat{E} = \{(D \cdot x^*_{uv}) \cdot (u,v) \mid (u,v) \in K(V)\}$. Note that D may be exponential in the graph size which creates a problem. In this section we assume that D is polynomial in the size of the graph and show a polynomial time implementation for general D in Section 4.

Note that it follows from the degree constraints of the LP that $D \cdot G$ is D-regular. Hence, we can apply the following lemma to the multigraph $D \cdot G$.

Lemma 3
1. Let G be a $d-$regular multigraph. The edges of G can be partitioned into d cycle covers.
2. If the indegree and the outdegree of any vertex in the multigraph G is at most d then edges of G can be partitioned into d collections of vertex disjoint cycles and paths.

This lemma is well-known and is a straightforward consequence of König's Theorem on the edge colorings of bipartite multigraphs [9].

Another ingredient necessary for our algorithm is a maximum matching in a directed graph. To find a maximum matching the directed graph can be converted to an undirected graph by setting the weight of edge (u, v) to be the maximum of $\{w_{uv}, w_{vu}\}$. A maximum matching in the directed graph now has a 1-1 correspondence with one in the undirected graph.

We now present the algorithm.

<u>MAX TSP ALGORITHM</u>

Step 1. Solve the LP.
Step 2. Set G' to be $D \cdot G$.
Step 3. Split G' into D cycle covers $C_1, ..., C_D$.
Step 4. Find a maximum matching M in G.
Step 5. Transform $C_1 \cup M, ..., C_D \cup M$ into D graphs $G_1, ..., G_D$ such that each G_i is 2-path-colorable by Lemma 2 with color-partition $\langle P_i, P_i' \rangle$.
Step 6. Choose the collection with largest weight from collections $\{P_1, P_1', ..., P_D, P_D'\}$.
Step 7. Patch the chosen collection to a Hamiltonian tour.

2.3 Transformation into D 2-Path-Colorable Graphs

Define the graphs G_i to be $C_i \cup M$. The graphs need to be transformed so that each will be 2-path-colorable. To achieve this we will be moving edges between the different graphs removing edges when necessary. Our goal is to do this in a manner that will allow us to bound the fraction of the overall weight that is removed when edges are deleted.

In this section we will do this is in such a manner that the overall weight of the removed edges is at most half the weight of M on average for each of the graphs G_i. So, on average for each G_i the weight will be bounded from below by $w(C) + \frac{1}{2}w(M) \geq w(opt) + \frac{1}{4}w(opt) = \frac{5}{4}w(opt)$. Hence, taking the graph with heaviest weight and coloring the edges with a 2-path-coloring yields a $\frac{5}{8}$ approximation for MAX TSP.

At the start each G_i contains a cycle cover and a matching. Therefore property (1) of the Path-Coloring Lemma is satisfied. The transformation which we will shortly describe needs to ensure that there are no back-edges on any cycle, i.e. property (2) of the Path-Coloring Lemma. There are 3 possible configurations where G_i might contain a cycle, not necessarily from the cycle cover, with a back-edge upon it.

Configuration 1: A 2-cycle from C_i with a parallel edge from the matching, see $\mathcal{F}_1$ in Figure 1.

Configuration 2: An edge (u,v) is on some cycle in C_i of size ≥ 3 and there is an edge $m=(v,u)\in M$, see Figure 1.

Configuration 3: A 2-cycle $\{(u,v),(v,u)\}$ appears in C_i, where, say, (v,u) is the back-edge. In this case since (u,v) is on a "cycle" there must be edges (x,u) and (v,w) from the matching M, where $x\neq w$ and $x,w\notin\{u,v\}$. In fact, there may be a chain of interchanging matching edges and 2-cycles, see Figure 2. It is important to note that to comply with property (2) of the Path-Coloring Lemma it is sufficient to break chains at any location on the chain.
Formally, $u_1,u_2,...,u_{2k}$ is a *chain* in G_i if (1) $(u_{2j-1},u_{2j})\in M$ for $1\leq j\leq k$ and (2) $(u_{2j},u_{2j+1}),(u_{2j+1},u_{2j})\in C_i$ for $1\leq j\leq k$ and (3) $k\geq 2$. A *maximal chain* in G_i is a chain that cannot be extended to a larger chain.

It is quite easy to verify, using a case-based analysis of a pair $\{u,v\}$, that these three bad configurations are the only ones violating property (2) of the Path-coloring Lemma in a graph that is the union of a matching and a cycle cover.

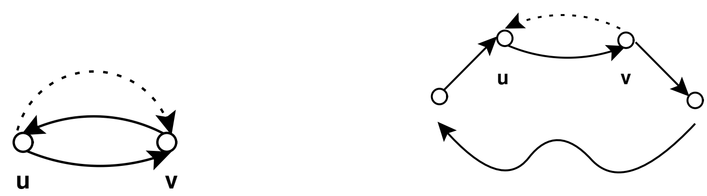

Figure 1: Dashed edges are from M and solid edges from the cycle cover. On the left, configuration 1, an $\mathcal{F}_1$, and, on the right, configuration 2.

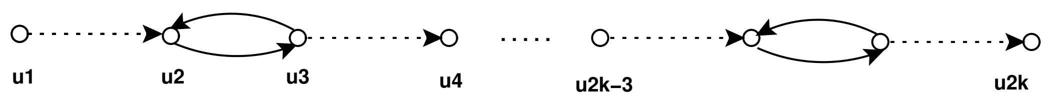

Figure 2: Configuration 3 - alternating edges from M and cycles from the cycle cover. Possibly, $(u_{2k-1},u_{2k})=(u_1,u_2)$.

2.3.1 Removing Configurations 1 and their counterparts Configurations 3

To remove the undesired configurations we will need to delete some of the edges from the graphs. However, to obtain a lower bound on the fraction of edges we remove we prove the following.

Lemma 4 (Balancing Lemma) *Let $m=(u,v)\in M$. If there are f graphs in which m is contained in an $\mathcal{F}_1$ then there are at least f other graphs in which the cycle cover does not contain both (u,v) and (v,u).*

Proof: By the two-cycle constraint $x^*_{uv}+x^*_{vu}\leq 1$. Hence, $D\cdot(x^*_{uv}+x^*_{vu})\leq D$. Since the edge (u,v) appears in $D\cdot x^*_{uv}$ covers and the edge (v,u) appears in $D\cdot x^*_{vu}$ covers the number of edges (u,v) and (v,u) in all cycle covers combined is $\leq D$.

By definition, for each G_i containing an $\mathcal{F}_1$ on vertices $\{u, v\}$ both (u, v) and (v, u) appear in C_i. If there are f graphs containing an $\mathcal{F}_1$ on vertices $\{u, v\}$ then these f graphs contain $2f$ cycle cover edges between u and v. Hence, there are at most $D - 2f$ **other** graphs G_j containing exactly one cycle cover edge (u, v) or (v, u). Since there are D graphs overall, f containing an $\mathcal{F}_1$ on vertices $\{u, v\}$ and at most $D - 2f$ graphs containing at least one edge from a cycle cover between u and v, it follows that there are at least f graphs containing no edge from a cycle cover between u and v. □

Consider an edge $m = (u, v) \in M$. m appears in each of the D graphs G_i. It follows from the Balancing Lemma that each m which appears in an $\mathcal{F}_1$, say in G_i, can be paired uniquely with an m, say in G_j, where (u, v) and (v, u) do not appear in C_j. An *m-pair* is a triplet $\langle m, i, j \rangle$, where $m \in M$ is on an $\mathcal{F}_1$ in G_i and its pair is in G_j.

We discard the edge m from the $\mathcal{F}_1$ in G_i, thus, eliminating Configuration 1. If we can guarantee that its pair, the m in G_j, is not removed then we only remove one out of two edges to eliminate the Configurations of type 1. However, since the m in G_j may participate in a Configuration 3 we need a slightly more sophisticated method which eliminates both undesired configurations together. Note that the m in G_j cannot be of type Configuration 2.

Prior to removing the edges of M that appear in $\mathcal{F}_1$s, we identify all maximal chains, i.e. Configuration 3, in all the graphs. Remember, breaking a chain at any point eliminates the problem of violating property (2) of the Path-Coloring-Lemma. One way to break the chain is to remove an edge on the chain. Another possibility is to reverse one of the edges from M on the chain. One must be careful not to reverse all edges from M on the chain, because otherwise, a reverse cycle might be created once again violating property (2) of the Path-Coloring-Lemma. Yet, reversing at least one edge and making sure that at least one edge does not get reversed is sufficient to "break" the chain.

Consider a pair $\langle m, i, j \rangle$ where m is on such a maximal chain in C_j. We check whether any edge has been reversed on this maximal chain. If one has, then there is no Configuration 3 "problem" in G_j and we simply remove m from G_i to eliminate the Configuration 1 from G_i. If no edge has been reversed on the maximal chain, then we reverse it, i.e. $m = (u, v)$ is removed from G_j and (v, u) is added to G_j. In parallel, we remove (v, u) from G_i. Hence, overall we have removed one copy of m. Both Configurations have been fixed.

2.3.2 Removing Configurations 3

The only type of Configuration 3 necessary to fix, after removing edges of type Configuration 1, are maximal chains that have not been paired with Configuration 1 edges. Removing the cheapest edge from M appearing on a chain breaks the chain as desired. Moreover, the weight removed is less than half the weight of the edges from M on this maximal chain, upper bounding the removal cost.

2.3.3 Removing Configurations 2

For those back-edges violating property (2) of the Path-Coloring-Lemma because of type Configuration 2, the following lemma provides a removing scheme. The scheme removes at most $\frac{1}{2}$ of the weight of the violating back-edge.

Lemma 5 *Let $P = (x,u),(u,v),(v,w)$ be a portion of a cycle from C_i in G_i and let $(v,u) \in M$. Let W be the overall weight of the edges in G_i. Then by (1) removing (u,v) or (2) removing (v,u) and adding another (u,v), the overall weight will be at least $W - \frac{1}{2}w_{vu}$.*

Proof: If $w_{uv} \leq \frac{1}{2}w_{vu}$ then (1) removing edge (u,v) gives the desired. Otherwise, $w_{uv} \geq \frac{1}{2}w_{vu}$ and (2) yields the result. □

Both (1) and (2) remove the undesired 2-cycle. Hence, this fixes the violation of property (2) in the Path-Coloring-Lemma.

2.4 Putting it all together

Step 5 of our algorithm works as follows.

<u>TRANSFORMATION TO BACK EDGE FREE CYCLES</u>

Step 5a. Identify edges $m = (u,v) \in M$ that appear on an $\mathcal{F}_1$ in each of the graphs.
Step 5b. Pair, uniquely, each such edge m in G_i with an m in some G_j such that $(u,v),(v,u) \notin C_j$.
Step 5c. Identify all maximal chains of Configuration 3.
Step 5d-a. For each m on an $\mathcal{F}_1$; if its pair is not on a maximal chain, remove it.
Step 5d-b. If its pair is on a maximal chain, then if $\exists e \in M$ on the maximal chain such that e has been reversed, remove m from the $\mathcal{F}_1$.
Step 5d-c. Otherwise, reverse m on the maximal chain, i.e. remove $m = (u,v)$ from G_j and replace it with (v,u). On G_i remove (v,u).
Step 5e. For each maximal chain where no edge has been reversed, delete cheapest edge from M on the chain.
Step 5f. To each appearance of $m = (u,v) \in M$ in each graph G_i such that (v,u) appears in C_i but (u,v) does not, apply Lemma 5.

Theorem 1 *The algorithm produces a Hamiltonian path of weight $\geq \frac{5}{8}w(opt)$, where $w(opt)$ is the weight of the maximal TSP.*

Proof: The Hamiltonian cycle in G corresponds to a feasible solution of the LP. The optimal solution to the LP is $\{x^*_{uv}\}_{(u,v)\in K(V)}$. Therefore, $w(opt) \leq \sum_{(u,v)\in K(V)} w_{uv}x^*_{uv}$. Hence, $D \cdot w(opt) \leq D \cdot \sum_{u,v\in K(V)} w_{uv}x^*_{uv}$. It follows from Lemma 3 that after step 3, $\sum_{i=1}^{D} w(C_i) = D \cdot \sum_{u,v\in K(V)} w_{uv}x^*_{uv}$. Obviously, $w(M) \geq \frac{1}{2} \cdot w(opt)$. Since M was added to each of the graphs

G_i, the weight of the graphs before the transformation was $\sum_{i=1}^{D} w(C_i) + D \cdot w(M)$. From the discussion in section 2.3 the overall weight deleted from all the graphs together is $\leq \frac{1}{2} \cdot D \cdot w(M)$. Hence the overall weight of the graphs after step 5 is at least $\sum_{i=1}^{D} w(C_i) + \frac{1}{2} \cdot D \cdot w(M) = D \cdot \sum_{u,v \in K(V)} w_{uv} x^*_{uv} + \frac{1}{2} \cdot D \cdot w(M) \geq D \cdot w(opt) + \frac{1}{4} \cdot D \cdot w(opt) = \frac{5}{4} \cdot D \cdot w(opt)$. Since each of the D graphs is two-path-colorable it follows that there are $2D$ path collections to choose from. Choosing the heaviest yields one of weight $\frac{\frac{5}{4} \cdot D \cdot w(opt)}{2D} = \frac{5}{8} \cdot w(opt)$. Completing the path collection to a Hamiltonian tour cannot decrease the weight. □

3 Algorithm for graphs with odd number of vertices.

If the input graph G has an odd number of vertices then the maximum matching M in G does not necessarily satisfy the inequality $w(M) \geq w(opt)/2$. In this case we show how to reduce the problem to the problem on a graph with an even number of vertices. Assume that we have guessed two consecutive edges $(v_1, v_2), (v_2, v_3)$ of some optimal tour for Max TSP in G; this is done by enumerating over all directed paths of length 2. Find the maximum matching M_1 in the graph $G \setminus \{v_1, v_2, v_3\}$ and maximum matching M_2 in the graph $G \setminus \{v_2\}$. We claim that $w(M_1) + w(M_2) + w_{v_1 v_2} + w_{v_2 v_3} \geq w(opt)$ and therefore $\max\{w(M_1) + w_{v_1 v_2} + w_{v_2 v_3}, w(M_2)\} \geq w(opt)/2$. The reason that $w(M_1) + w(M_2) + w_{v_1 v_2} + w_{v_2 v_3} \geq w(opt)$ is that the optimal tour can be partitioned into two edge disjoint graphs: one is a matching in $G \setminus \{v_1, v_2, v_3\}$ plus two edges $(v_1, v_2), (v_2, v_3)$ and another is a matching in $G \setminus \{v_2\}$.

If $w(M_2) \geq w(M_1) + w_{v_1 v_2} + w_{v_2 v_3}$ then we can apply the previous algorithm without modifications since in this case $w(M_2) \geq w(opt)/2$ and M_2 is a matching.

If $w(M_2) < w(M_1) + w_{v_1 v_2} + w_{v_2 v_3}$ then we delete vertex v_3 from G and redefine the weight of the edge (v_1, v_3) to be $w_{v_1 v_2} + w_{v_2 v_3}$. After that we apply our algorithm to the optimal fractional solution of the LP defined on the new instance of the problem and matching $M_1 \cup \{(v_1, v_3)\}$. If P is the heaviest collection of paths delivered by the algorithm then we add vertex v_2 back to the graph G. If P contains edge (v_1, v_3) then we delete it and add two edges (v_1, v_2) and (v_2, v_3) instead. Since the optimal value of the LP on the new instance of the problem is an upper bound on the $w(opt)$ we obtain that $w(P) \geq w(opt)/2 + (w(M_1) + w_{v_1 v_2} + w_{v_2 v_3})/4 \geq 5/8 w(opt)$

4 Polynomial time implementation of the algorithm

The algorithm described in the previous section might obviously run in exponential time. The reason is that the number D (defined as the minimum number such that all numbers $D x^*_{uv}$ are integers) can be exponentially big and therefore, the algorithm will produce an exponential number of cycle covers. The simple way to deal with this problem is to round each number x^*_{uv} down to the nearest multiple of ε/n^2 where $\varepsilon > 0$ is some precision parameter such that $\varepsilon = 1/s$ for some integer s. Let $\bar{x}_{uv}$ be the value of variable x_{uv} obtained after rounding x^*_{uv}. Let $\bar{D}$ be a minimum number such that all numbers $\bar{D}\bar{x}_{uv}$ are integers. Clearly, $\bar{D} \leq sn^2$. Consider the multigraph $\bar{D} \cdot G$, i.e. the multigraph having $\bar{D}\bar{x}_{uv}$ copies of edge (u, v). Since we rounded x^*_{ij} down the indegree and outdegree of any vertex in $\bar{D} \cdot G$ are at most $\bar{D}$, moreover there are at most $\bar{D}$ edges between any pair of vertices u and v by the two-cycle constraint. By applying Lemma 3 we can partition $\bar{D} \cdot G$ into $\bar{D}$ collections of vertex disjoint cycles and paths. Adding the matching to each of these collections and eliminating all three types of "bad" configurations as in the previous section we obtain $\bar{D}$ graphs satisfying conditions of Lemma 2. Therefore, applying Lemma 2 to each of these graphs we get $2\bar{D}$ partial tours which can be completed to the Hamiltonian cycle. We now estimate the value of the best Hamiltonian path obtained this way. If W is a maximum weight of the edge in the input graph G then clearly, $W \leq w(opt)$. The total weight of the graph $\bar{D} \cdot G$ is

$$\bar{D} \sum_{u,v \in V} w_{uv} \bar{x}_{uv} \geq \bar{D} \sum_{u,v \in V} w_{uv}(x^*_{uv} - \varepsilon/n^2) \geq \bar{D} \sum_{u,v \in V} w_{uv} x^*_{uv} - \bar{D}\varepsilon W.$$

Therefore, the Hamiltonian cycle delivered by the algorithm has weight at least $\frac{5}{8}w(opt) - \frac{\varepsilon W}{2} \geq (\frac{5}{8} - \frac{\varepsilon}{2})w(opt)$.

We now describe a more complicated procedure of implementing our algorithm in polynomial time without losing ε in the performance guarantee. A matrix is called *doubly stochastic* if all of its entries are nonnegative and the sum of all the elements in any row or column is exactly one. A doubly stochastic matrix with integer entries is called a *permutation matrix.*

Lemma 6 (Birhoff-von Neumann [15]) *Any doubly stochastic $n \times n$ matrix can be represented as a convex combination of at most n^2 permutation matrices and such representation can be found in polynomial time.*

This lemma can be derived by recursive application of Hall's Theorem [9]. The actual representation can be found by at most n^2 applications of the algorithm for finding a perfect matching in bipartite graph.

The optimal solution of the LP $X = \{x^*_{uv}\}_{u,v \in K(V)}$ is a doubly stochastic matrix by indegree, outdegree and nonnegativity constraints. Applying Lemma 6 we obtain $X = \sum_{i=1}^{n^2} \lambda_i \Pi_i$ where Π_i are permutation matrices, $\lambda_i \geq 0$ are coefficients in the convex combination, i.e. $\sum_{i=1}^{n^2} \lambda_i = 1$. Each permutation matrix Π_i corresponds to a cycle cover C_i in the original graph G. Moreover if we consider the multigraph $D \cdot G$ where D is the minimal number such that $\lambda_i D$ are integers for all $i = 1, \ldots, n^2$ (such a D is not necessary polynomially bounded) then Π_i represents $\lambda_i D$ identical cycle covers in the cycle cover decomposition of $D \cdot G$ guaranteed by Lemma 3. We now implement our algorithm working with cycle covers $C_i, i = 1, \ldots, n^2$ instead of working with D cycle covers explicitly.

In the first stage we add the matching M to each of the cycle covers $C_i, i = 1, \ldots, n^2$. If all graphs $G_i = C_i \cup M, i = 1, \ldots, n^2$ don't contain "bad" configurations (Figures 1,2) then by the Lemma 2 we can find a 2-path-coloring of each $C_i \cup M$. Let P_i and P'_i be collections of paths corresponding to a 2-path-coloring of $C_i \cup M$. Then the weight of the heaviest collection P of paths is at least $3/4w(opt)$ since $w(P) \geq \sum_{i=1}^{n^2} \lambda_i(w(P_i) + w(P'_i))/2 = \sum_{i=1}^{n^2} \lambda_i(w(C_i) + w(M))/2 \geq 3/4w(opt)$. So, what we need to show is that we can remove all "bad" configurations, as before, by deleting at most half of edges from M on average.

Removing Configurations 1 and their counterparts Configurations 3 We now prove a generalization of the Balancing Lemma used in the previous section.

Lemma 7 (Generalized Balancing Lemma) *Let $m = (u, v) \in M$ and let S_2 be a subset of the set $\{1, \ldots, n^2\}$ such that m is contained in an $\mathcal{F}_1$ in each graph $C_i \cup M$ for $i \in S_2$. Then there is a set $S_0 \subset \{1, \ldots, n^2\}$ such that (1) for each $i \in S_0$ the cycle cover C_i in the graph $G_i = C_i \cup M$ does not contain (u, v) and (v, u) and (2) $\sum_{i \in S_0} \lambda_i \geq \sum_{i \in S_2} \lambda_i$.*

Proof: Let $S_0 \subseteq \{1, \ldots, n^2\}$ be the set of cycle covers which do not contain (u, v) and (v, u) and let $S_1 = \{1, \ldots, n^2\} \setminus (S_2 \cup S_0)$, i.e. S_1 is the set of cycle covers C_i having exactly one edge ((u, v) or (v, u)) between vertices u and v. By the two-cycle constraint $x^*_{uv} + x^*_{vu} \leq 1$. Hence, $\sum_{i \in S_2} 2\lambda_i + \sum_{i \in S_1} \lambda_i \leq 1$. However, $\sum_{i \in S_2 \cup S_1 \cup S_0} \lambda_i = 1$ and therefore, $\sum_{i \in S_0} \lambda_i \geq \sum_{i \in S_2} \lambda_i$. □

Consider the edge $m = (u, v) \in M$. Let $\bar{S}_0 \subseteq S_0$ be any minimal subset of S_0 such that $\sum_{i \in \bar{S}_0} \lambda_i \geq \sum_{i \in S_2} \lambda_i$. If $\sum_{i \in \bar{S}_0} \lambda_i > \sum_{i \in S_2} \lambda_i$ then we take any $t \in \bar{S}_0$ and define two copies t' and t'' of the cycle cover C_t having weights in a convex combination $\lambda_{t''} = \sum_{i \in \bar{S}_0} \lambda_i - \sum_{i \in S_2} \lambda_i$ and $\lambda_{t'} = \lambda_t - \lambda_{t''}$. Redefine $\bar{S}_0 = \bar{S}_0 \cup \{t'\} \setminus \{t\}$ and $S_0 = S_0 \cup \{t', t''\} \setminus \{t\}$. By construction we have $\sum_{i \in \bar{S}_0} \lambda_i = \sum_{i \in S_2} \lambda_i$. We now partition the set $\bar{S}_0$ into two sets S'_0 and S''_0. The set S'_0 consists of those graphs G_i from $\bar{S}_0$ where edge m is on a maximal chain (Configuration 3) and there were no edges reversed on this maximal chain on prior steps. Let $S''_0 = \bar{S}_0 \setminus S'_0$. After that we partition the set S_2 into two sets S'_2 and S''_2 in such a way that $\sum_{i \in S'_0} \lambda_i = \sum_{i \in S'_2} \lambda_i$ and $\sum_{i \in S''_0} \lambda_i = \sum_{i \in S''_2} \lambda_i$. If there is no such S'_2 then we make two copies j' and j'' of some graph $G_j, j \in S_2$ and define weights $\lambda_{j'}$ and $\lambda_{j''}$ such that $\lambda_j = \lambda_{j'} + \lambda_{j''}$. Put one copy to S'_2 and another to S''_2. We pair set S'_0 with set S'_2 and set S''_0 with set S''_2. For the first pair of sets we reverse edge (u, v) in graphs $G_i, i \in S'_0$ and delete (v, u) in S'_2 as we did in Section 2.3.2. For the second pair we just delete edge (u, v) from all $G_i, i \in S''_2$. We repeat this step for all other edges $(u, v) \in M$ which belong to some "bad" configuration $\mathcal{F}_1$. Each such step increases the number of cycle covers in the convex combination by at most two, moreover we delete at most half of any matching edge on average. Therefore, in the end of this exclusion step we don't have configurations $\mathcal{F}_1$ and we have at most $3n^2$ cycle covers in a convex combination and corresponding graphs G_i.

Removing Configurations 3 We do exactly what we did in Section 2.3.2. We just remove the cheapest edge in any maximal chain that has not been paired with Configuration 1 edges.

Removing Configurations 2 Applying Lemma 5 we remove at most half of any matching edge participating in such configuration.

5 Conclusion

An obvious open question is to close the gap between the best known positive result 5/8 and negative result 319/320 for Maximum Asymmetric TSP. On one hand, we don't even know the gap between the value of an optimal fractional solution of the linear programming relaxation considered in this paper and the optimal solution of MAX ATSP. The worst instance we know has gap 2/3. On the other hand, our LP is rather weak, we could add many valid inequalities known for TSP [11]. The most promising set of constraints is

$$\sum_{u,v \in S} x_{uv} \leq |S| - 1, \text{ for all } S \subset V.$$

These constraints are known as *subtour elimination* inequalities and they guarantee that the graph $D \cdot G$ considered in the Section 2 is strongly D-edge-connected. Unfortunately, we don't know how to use this property in the rounding procedure. However, we point out that it does eliminate the example showing a 2/3 gap for the LP with weaker constraints.

6 Acknowledgements

The authors would like to thank A. Ageev, R. Hassin and A. Yeo for helpful discussions on the subject of the paper.

References

[1] A. I. Barvinok, E.Kh. Gimadi and A.I. Serdyukov, The Maximum Traveling Salesman Problem, to appear in the book "The Traveling Salesman Problem and Its Applications", edited by G. Gutin and A. Punnan.

[2] Markus Bläser. An $\frac{8}{13}$ Approximation Algorithm for the Asymmetric Max-TSP. *Proc. of Symposium on Discrete Algorithms*, 2002, 64-73.

[3] Markus Bläser. An $\frac{8}{13}$ Approximation Algorithm for the Asymmetric Max-TSP. Technical Report SIIM-TR-A-01-21, Institute fr Informatik und Mathematik, Universtitt Lbeck, Germany, 2001.

[4] M. Bläser and B. Siebert, Computing Cycle Covers without Short Cycles, In Proceedings of ESA 2001, LNCS 2161, pp. 368–380.

[5] A. Blum, T. Jiang, M. Li, J. Tromp and M. Yannakakis, Linear Approximation of Shortest Superstring. *J. of the ACM*, 31(4), 1994, 630-647.

[6] D. Breslauer, T. Jiang and Z. Jiang, Rotations of periodic strings and short superstrings, *Journal of Algorithms* 24(2):340–353, 1997.

[7] L. Engebretsen, An explicit lower bound for TSP with distances one and two. STACS 99 (Trier), 373–382, *Lecture Notes in Comput. Sci.*, 1563, Springer, Berlin, 1999.

[8] L. Engebretsen and M. Karpinski, Approximation hardness of TSP with bounded metrics, In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, Proceedings of 28th International Colloquium on Automata, Languages and Programming, Lecture Notes in Computer Science, v. 2076, pp. 201-212. Springer-Verlag, Crete, Greece, 2001.

[9] S. Fiorini and R. Wilson, Edge-colourings of graphs. Research Notes in Mathematics, No. 16. Pitman, London; distributed by Fearon-Pitman Publishers, Inc., Belmont, Calif., 1977.

[10] M.L. Fisher, L. Nemhauser and L.A. Wolsey. An Analysis of Approximations for Finding a Maximum Weight Hamiltonian Circuit. *Networks*, 12(1):799-809, 1979.

[11] M. X. Goemans, Worst-case comparison of valid inequalities for the TSP, *Math. Programming* 69(2A):335–349, 1995.

[12] The traveling salesman problem. A guided tour of combinatorial optimization. Edited by E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys. Wiley-Interscience Series in Discrete Mathematics. A Wiley-Interscience Publication. John Wiley & Sons, Ltd., Chichester, 1985.

[13] S.R. Kosaraju, J.K. Park and C. Stein. Long Tours and Short Superstrings. *Proc. 35th Annual Symposium on Foundations of Computer Science*, 166-177, 1994.

[14] M. M. Koval'ov and V. M. Kotov, Estimate of the error of series of approximate algorithms. (Russian) Vestnik Beloruss. Gos. Univ. Ser. I Fiz. Mat. Mekh. 1986, no. 3, 44–48,

[15] E. L. Lawler, Combinatorial optimization: networks and matroids. Holt, Rinehart and Winston, New York-Montreal, Que.-London, 1976, page 189.

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

[17] Z. Sweedyk, $2\frac{1}{2}$-approximation algorithm for shortest superstring, *SIAM J. Comput.* 29(3):954–986, 2000.

[18] J. Tarhio, E. Ukkonen. A Greedy Approximation Algorithm for Constructing Shortest Common Superstrings. *Theoretical Computer Science*, 57:131-145, 1988.

[19] J. S. Turner. Approximation algorithms for the shortest common superstring problem. *Information and Computation*, 83:1–20, 1989.

[20] S. Vishwanathan, An approximation algorithm for the asymmetric travelling salesman problem with distances one and two. *Inform. Process. Lett.* 44(6):297–302, 1992.