Chapter 12

THE MAXIMUM TSP

Alexander Barvinok [*]
Department of Mathematics
University of Michigan
Ann Arbor, MI 48109-1109, USA
barvinok@math.lsa.umich.edu

Edward Kh. Gimadi [†]
Sobolev Institute of Mathematics
Novosibirsk, Russia
gimadi@math.nsc.ru

$\boxed{\text{Anatoliy I. Serdyukov}}$ [‡]
Sobolev Institute of Mathematics
Novosibirsk, Russia

1. Introduction

The Maximum Traveling Salesman Problem (MAX TSP), also known informally as the "taxicab ripoff problem", is stated as follows:

Given an $n \times n$ real matrix $c = (c_{ij})$, called a *weight matrix*, find a Hamiltonian cycle $i_1 \mapsto i_2 \mapsto \ldots \mapsto i_n \mapsto i_1$, for which the maximum value of $c_{i_1 i_2} + c_{i_2 i_3} + \ldots + c_{i_{n-1} i_n} + c_{i_n i_1}$ is attained. Here $(i_1, \ldots, i_n)$ is a permutation of the set $\{1, \ldots, n\}$.

Of course, in this general setting, the Maximum Traveling Salesman Problem is equivalent to the Minimum Traveling Salesman Problem, since the maximum weight Hamiltonian cycle with the weight matrix c corresponds to the minimum weight Hamiltonian cycle with the weight

[*]Partially supported by NSF Grant DMS 9734138
[†]Partially supported by RFFI grant 99-01-00601
[‡]Partially supported by RFFI grant 99-01-00601

G. Gutin et al. (eds.), *The Traveling Salesman Problem and Its Variations*

matrix $-c$. What makes the MAX TSP special is that there are some interesting and natural special cases of weights c_{ij}, not preserved by the sign reversal, where much more can be said about the problem than in the general case. Besides, methods developed for the Maximum Traveling Salesman Problem appear useful for other combinatorial and geometric problems.

In most cases, in particular in problems P1–P3 and P5–P6 below, we assume that the weight matrix is *non-negative*: $c_{ij} \geq 0$ for $i, j = 1, \dots, n$. This is the main condition that breaks the symmetry, since the corresponding condition of non-positive weights for the Minimum Traveling Salesman Problem is rarely, if ever, imposed and considered unnatural. It also makes the MAX TSP somewhat easier than the MIN TSP. We will discuss the following special cases.

P1. The symmetric problem. We have $c_{ij} = c_{ji}$ for all i, j.

P2. The semimetric problem. We have $c_{ij} + c_{jk} \geq c_{ik}$ for all triples (i, j, k).

P3. The metric problem. We have $c_{ij} = c_{ji}$ for all pairs (i, j) and $c_{ij} + c_{jk} \geq c_{ik}$ for all triples (i, j, k).

P4. The problem with warehouses. In this case, the matrix c has some special structure. We are given $r \times n$ matrices $u = (u_{ij})$ and $v = (v_{ij})$ such that

$$c_{ij} = \max_{k=1,\dots,r} (u_{ki} + v_{kj}) \quad \text{for all} \quad i, j = 1, \dots, n. \tag{1}$$

The number r is assumed to be small (fixed) and n is allowed to vary. The smallest r for which representation (1) exists is called in [86] the *combinatorial rank* of c. The problem has the following interpretation [88]: suppose that together with n cities, there are r *warehouses*. Before visiting each city, the Traveling Salesman has to pick up goods at any of the r warehouses. The cost of going from the i-th city to the k-th warehouse is u_{ki} and the cost of going from the k-th warehouse to the j-th city is v_{kj}. If the Traveling Salesman wants to maximize the cost of the tour visiting each city only once (with no restrictions regarding visiting warehouses imposed), he should use the cost matrix c determined by the above equations. We do not assume c_{ij} to be non-negative and the analysis of the MAX TSP in this case mirrors that of the MIN TSP with "max" replaced by "min" throughout.

P5. The problem in a normed space. Let us fix a norm $\|\cdot\|$ in Euclidean space $\mathbb{R}^d$. More precisely, we require $\|x\|$ to be a non-negative real number for any $x \in \mathbb{R}^d$, that $\|\lambda x\| = \lambda \|x\|$ for all

$x \in \mathbb{R}^d$ and all $\lambda \geq 0$ and that $\|x + y\| \leq \|x\| + \|y\|$ for all $x, y \in \mathbb{R}^d$. We do not require that $\|-x\| = \|x\|$. It is convenient, although not necessary, to assume that $\|x\| = 0$ implies $x = 0$. Although the symmetry condition may be violated, we call $\|\cdot\|$ a norm anyway, hoping that this will not lead to a confusion. In this case, the weight matrix c has the following structure: we are given n points $p_1, \ldots, p_n$ in $\mathbb{R}^d$ such that $c_{ij} = \|p_j - p_i\|$ for all i and j.

P6. The problem in a polyhedral norm. This is a particular case of P5 with the additional assumption that the unit ball $\mathbf{B} = \{x \in \mathbb{R}^d : \|x\| \leq 1\}$ is a polyhedron, that is the intersection of finitely many halfspaces.

Some relations between P1–P6 are seen instantly: it is immediate that P3 is the intersection of P1 and P2, that P6 is a particular case of P5 and that P5 is a particular case of P2. It is also clear that P5 can be viewed as a "limit" of P6. Moreover, for any particular instance of MAX TSP with n points in a general norm, there is an equivalent instance in a polyhedral norm with the unit ball $\mathbf{B}$ having $O(n^2)$ facets (the facets of $\mathbf{B}$ account for the directions $p_i - p_j$, where $p_1, \ldots, p_n$ are given points). This idea is from [285], where it is attributed to J. Mitchell. Although not very difficult, it may seem a little surprising that P6 is a particular case of P4, with the warehouses in P4 corresponding to the facets of $\mathbf{B}$.

In this chapter, we describe what is known (to us) about special cases P1–P6. We also describe some results on probabilistic analysis of the problem when weights c_{ij} are sampled at random from some distribution.

Before we proceed, we discuss an interesting application of the MAX TSP.

1.1. An application: the shortest superstring problem

Given strings $s_1, \ldots, s_n$ over some alphabet, find the shortest string s (superstring) containing each s_i as a substring. This problem, known as the shortest superstring problem has applications to DNA sequencing and data compression (see [515]).

In [114], the following reduction to the MAX TSP was suggested. Let c_{ij} be the length of the overlap of s_i and s_j, that is the length of the longest string v such that $s_i = uv$ and $s_j = vw$ for some strings u and w. It turns out that an approximate solution to the MAX TSP with the weight matrix c gives rise to an approximate solution to the shortest superstring problem. More precisely, let us fix a real number ρ in the interval $[1/2, 1]$. Given a Hamiltonian cycle whose weight approximates the maximum weight of a Hamiltonian cycle within a factor of ρ, one can produce a superstring s whose length approximates the minimum length of a superstring within a factor of $(4 - 2\rho)$.

2. Hardness Results

In [659] it is shown that the MIN TSP is MAX SNP-hard even when restricted to matrices with weights $c_{ij} \in \{1, 2\}$. MAX SNP-hardness means that unless P=NP, there is a constant $\epsilon > 0$ for which there is no polynomial time algorithm approximating the minimum weight of a Hamiltonian cycle up to within relative error ϵ. It follows that P1–P3 are MAX SNP-hard. In P4, as long as the number r of warehouses is allowed to vary, the problem is as hard as the general TSP since any $n \times n$ matrix c_{ij} can be written as

$$c_{ij} = \max_{k=1,\dots,n} (u_{ki} + v_{kj}),$$

where $v_{ij} = c_{ij}$, $u_{ii} = 0$ and $u_{ij} = -\infty$ for $i \neq j$.

However, we see in Section 8 that the problem becomes polynomially solvable when r is fixed. We also note that it is an NP-hard problem to find the combinatorial rank of a given matrix c although it is possible to check in polynomial time whether the combinatorial rank does not exceed 2 and find the corresponding representation (1), see [174].

Fekete in [285] shows that P5 with $\|x\| = \sqrt{\xi_1^2 + \ldots + \xi_d^2}$ for $x = (\xi_1, \ldots, \xi_d)$ being the standard Euclidean norm is NP-hard provided $d \geq 3$ (the idea is to reduce the problem of finding a Hamiltonian cycle in a grid graph to the MAX TSP in 3-dimensional Euclidean space by using a clever drawing of the graph on the unit sphere). As we noted in Section 1, any instance of the MAX TSP with n points in a normed space can be reduced to an instance of the MAX TSP in a polyhedral norm for which the unit ball $\mathbf{B}$ has $O(n^2)$ facets. Using this observation, Fekete further shows that P6 is an NP-hard problem provided the number of facets of the unit ball $\mathbf{B}$ is allowed to vary. As we discuss in Section 8, the problem is polynomially solvable if the number of facets of $\mathbf{B}$ is fixed. The status of the MAX TSP in $\mathbb{R}^2$ with the standard Euclidean norm is not known. Note that in the case of Euclidean norm there is a standard difficulty of comparing lengths (which are sums of square roots of rational numbers), so it is not clear whether the decision version of the MAX TSP (given a rational number ρ, is there a Hamiltonian cycle whose length exceeds ρ) is in NP.

3. Preliminaries: Factors and Matchings

If G is an undirected graph, one can view a Hamiltonian cycle in G as a *connected* subset $E' \subseteq E$ of edges such that every vertex $v \in V$ is incident to exactly two edges in E'. If we do not insist that E' is connected, we come to the notion of a 2-factor.

We recall that a set $E' \subseteq E$ of edges of an undirected graph $G = (V, E)$ is called a 2-matching if every vertex of G is incident to at most two edges from E'. A set $E' \subseteq E$ of edges of an undirected graph $G = (V, E)$ is called a 2-factor if every vertex of G is incident to exactly two edges from E' (see Appendix A). More generally, suppose that to every vertex v of G a non-negative integer $f(v)$ is assigned. A set $E' \subseteq E$ is called an *f-factor* if every vertex v of G is incident to exactly $f(v)$ edges from E'. If $f(v) \in \{0, 1\}$ for every v, an f-factor is called a matching and if $f(v) = 1$ for all v, the matching is called perfect (see Appendix A).

As is known, (see, for example, Chapter 9 of [570] and Appendix A), a maximum weight (perfect) matching in a given weighted graph can be constructed in $O(|V|^3)$ time. Finding a maximum weight f-factor in a given weighted graph G can be reduced to finding a maximum weight perfect matching in some associated graph G'' (see Chapter 10 of [570]) with $|V''| = O(|E|)$ and can be solved in polynomial time as well. Various methods of finding an approximate solution to the MAX TSP (MIN TSP) are based on finding a 2-factor of the maximum (minimum) weight and then transforming it to a Hamiltonian cycle.

Using Edmonds' technique of "blossoms", Hartvigsen in [435] obtained an $O(n^3)$ algorithm to construct a maximum weight 2-matching in a given weighted graph with n vertices. A maximum weight 2-factor can be found in $O(n^3)$ time as well by Hartvigsen's algorithm after some initial conditioning of the weights. All weights on the edges of the graph should be increased by a large constant so that every maximum weight 2-matching is necessarily a 2-factor.

One can observe that a 2-factor is just a union of vertex-disjoint cycles $S_1, \dots, S_l$ such that the union $S_1 \cup \dots \cup S_l$ contains every vertex of the graph. As we try to approximate an optimal Hamiltonian cycle by an optimal 2-factor, it seems natural to try to search for an optimal 2-factor satisfying some additional restrictions that would make it somewhat closer to a Hamiltonian cycle.

A 2-factor is called *k-restricted* if none of the cycles $S_1, \dots, S_l$ contains k or fewer vertices. Hence a 2-factor is 2-restricted and a Hamiltonian cycle is k-restricted for any $k < |V|$.

The problem of finding a maximum weight 4-restricted 2-factor is NP-hard [812], whereas the status of the problem for a maximum weight

3-restricted 2-factor is not known. However, there is a polynomial time algorithm for finding an (unweighted) 3-restricted 2-factor in a given graph [435] and a 4-restricted 2-factor in a given bipartite graph, if there is any [436]. In some cases, see, for example, [659], knowing how to find some k-restricted 2-factor allows one to get a better solution to the weighted problem. The problem of deciding if there is a 5-restricted 2-factor is NP-complete (this result belongs to Papadimitriou, see [223] and [436]) and the status of the problem for 4-restricted 2-factors in general (not necessarily bipartite) graphs is not known.

Now we briefly describe the situation of a directed graph.

Let $G = (V, A)$ be a directed graph. We recall (see Appendix A) that collection of vertex disjoint cycles $S_1, \ldots, S_l$ is called 1-factor (or, sometimes, *cycle cover*) if every vertex of G belongs to one of the cycles.

Given a weighted directed graph with n vertices, one can find the 1-factor of the maximum (minimum) weight in $O(n^3)$ time (see, for example, Chapter 11 of [656] and Appendix A).

Recall that a partial tour is a set of edges (arcs) of an undirected (directed) graph which can be appended by including some other edges (arcs) of the graph to a Hamiltonian cycle in the graph (see Appendix A). As we remarked earlier, in almost all cases we assume that the weights on the edges are non-negative. A frequently used approach to the Maximum TSP is to construct a partial tour of a sufficiently large weight and then append it to a Hamiltonian cycle. When the weights are non-negative, adding new edges may only increase the total weight.

4. MAX TSP with General Non-Negative Weights

In this section, we assume that the weight matrix is non-negative but otherwise does not have any special properties. In other words, we are looking for a maximum weight Hamiltonian cycle in a complete weighted directed graph $G = (V, A)$ with $|V| = n$ vertices and $|A| = n(n-1)$ arcs (note, that the arcs (i, j) and (j, i) are different). The following notion will be used throughout the paper.

Definition 1 *Suppose that we have an algorithm for a class of problems with non-negative weights. We say that the algorithm has a performance ratio ρ (where ρ is a positive real number) if it produces a Hamiltonian cycle whose weight is at least ρ times the maximum weight of a Hamiltonian cycle.*

For general non-negative weights the following performance ratios can be obtained in polynomial time: 1/2 [308], 4/7 [521] and 38/63 [515], the best currently known.

The rest of this section describes the idea of the algorithm from [515]. First, a maximum weight 1-factor F is constructed in $O(n^3)$ time. Then three different ways to patch the cycles into a Hamiltonian cycle are considered. Cycles with 2 vertices, also called *2-cycles*, play a special role.

The first way consists of deleting a minimum weight arc from each of the cycles of F and extending the obtained paths into a tour H_1 (this is the idea of the algorithm in [308]).

In the second way, a weighted undirected graph G' is produced from the given complete digraph G and a Hamiltonian cycle H_2 is constructed using the maximum weight matching in G' (cf. also Section 5).

In the third way, cycles of F with many vertices are broken into pieces and an auxiliary directed graph G'' is constructed by contracting the pieces. A Hamiltonian cycle H_3 is obtained by using a maximum weight matching in G''. Let W be the weight of F, let bW be the total weight of the heavier arcs in the 2-cycles of F and let cW be the weight of the lighter arcs in the 2-cycles of F.

It turns out that H_1 provides a $(2/3 + (b - 2c)/3)$ performance guarantee, H_2 provides a $(7/12 - (b - 2c)/12)$ performance guarantee and H_3 provides a $(2/3 + 4(b - 2c)/15)$ performance guarantee. It follows then that the best of H_1, H_2 and H_3 provides a 38/63 performance guarantee.

Although the original paper uses all three cycles H_1, H_2 and H_3, it appears that H_2 and H_3 alone are sufficient to obtain a 38/63 performance guarantee.

In the next three sections, we are going to discuss approximation algorithms for problems P1-P3.

5. The Symmetric MAX TSP

We consider the MAX TSP with non-negative weights c_{ij} subject to the symmetry condition of P1. In other words, we are looking for a maximum weight Hamiltonian cycle in a weighted complete undirected graph $G = (V, E)$ with $|V| = n$ vertices and $|E| = \binom{n}{2}$ edges.

In [308] a polynomial time algorithm with the performance ratio of 2/3 was suggested. The algorithm finds the maximum weight 2-factor and modifies it to a Hamiltonian cycle, cf. Section 3. In [520] a better performance ratio of 13/18 was achieved and in [438] a polynomial time algorithm with the performance ratio 5/7 was constructed. Below we sketch an algorithm due to Serdyukov [751], which achieves the performance ratio of 3/4 and has $O(n^3)$ complexity.

5.1. Sketch of the Algorithm from [751]

First, we describe the algorithm when the number n of vertices is even. This is also the simplest case to analyze.

Part I, n is even. Let us construct a maximum weight 2-factor F and a maximum weight perfect matching M (cf. Section 3). The 2-factor F is a union $F = S_1 \cup \ldots \cup S_l$ of vertex disjoint cycles, each of which contains at least 3 edges of the graph. The idea is to construct two partial tours T_1 and T_2, choose the one of the largest weight and arbitrarily extend it to a Hamiltonian cycle. We start with $T_1 = M$ and $T_2 = \emptyset$. Now we process cycles $S_1, \ldots, S_l$ one by one, so that precisely one edge e_i of each cycle S_i is assigned to T_1 and the remaining edges of S_i are assigned to T_2. Clearly, T_2 will always be a partial tour.

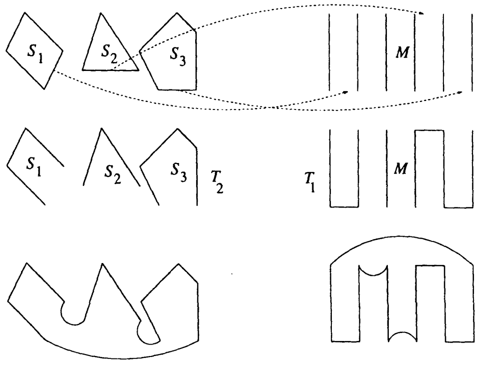

Figure 12.1. Constructing a Hamiltonian cycle when n is even

Now, we observe that we can choose e_i in such a way that T_1 remains a partial tour. Indeed, we can choose e_1 arbitrarily. Let us assume that $e_1, \ldots, e_{i-1}$, $i \geq 2$ are chosen, hence T_1 is a union of vertex-disjoint paths. Let us pick two adjacent candidate edges e'_i and e''_i of S_i. The common vertex of e'_i and e''_i is an end-vertex of a path in T_1, hence we choose $e_i \in \{e'_i, e''_i\}$ so that e_i is not incident to the other end-vertex of that path.

When all cycles $S_1, \ldots, S_l$ are processed, we choose T to be the one of T_1 and T_2 with the larger weight $c(T)$ (either of the two if $c(T_1) = c(T_2)$). Finally, we extend T to a Hamiltonian cycle.

We observe that after the cycle S_i has been processed, the weight $c(T_1) + c(T_2)$ increases by $c(S_i)$ and hence in the end we have $c(T_1) + c(T_2) = c(M) + c(F)$. Thus $c(T) \geq \dfrac{c(F) + c(M)}{2}$ and the 3/4 performance grantee follows from the observation that $c(F) \geq c(H)$ and $c(M) \geq c(H)/2$ for every Hamiltonian cycle H.

The case of an odd number n of vertices requires some adjustments and its relies on some of the arguments and constructions in Part I.

Part II, n is odd. This time, there is no perfect matching M but one can find a maximum weight almost 1-factor, which covers all but one vertex v of the graph and which we also call M. As above, we construct a maximum weight 2-factor $F = S_1 \cup \ldots \cup S_l$. Let us assume that v is a vertex of S_1 and let e be an edge which is incident to v, is not an edge of F and has the maximum weight among all such edges. Depending on whether the other end-vertex of e belongs to S_1 or to some other cycle S_2, we construct a partial tour T_2 by either removing two non-adjacent edges of S_1 adjacent to e or by removing an edge adjacent to e in S_1 and another edge adjacent to e in S_2 and patching S_1 and S_2 via e.

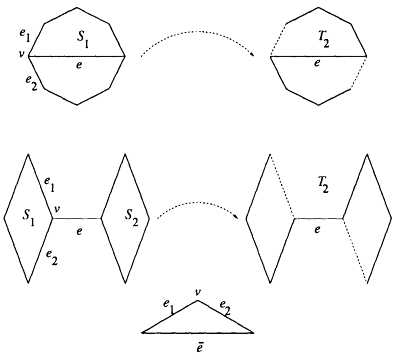

Figure 12.2. Constructing a partial tour T_2 when n is odd

The two removed edges are added to M thus forming a partial tour T_1. Next, we process the remaining cycles of F (that is, the cycles S_i for $i > 1$ in the first case and the cycles S_i for $i > 2$ in the second case) as in Part I and produce a partial tour $\tilde{T}$. The weight of $\tilde{T}$ is compared to the weight of a partial tour $\overline{T}$ in a modified weighted graph $\overline{G}$ on $n - 1$ vertices constructed as follows. Let e_1 and e_2 be the edges of S_1 that are incident to v. We remove vertex v and all edges incident to it from G and preserve the weights on all the remaining edges of G, except for a single edge $\overline{e}$ for which e_1, e_2 and e form a triangle. The weight of $\overline{e}$ is modified: $c(\overline{e}) := c(e_1) + c(e_2)$. Since the new graph $\overline{G}$ has an even number of vertices, we construct a partial tour $\overline{T}$ as in Part 1. If $\overline{T}$ happens to contain $\overline{e}$, we modify it by removing $\overline{e}$ and inserting e_1 and

e_2 instead. Hence we obtain a partial tour $\overline{T}$ in G. From the partial tours $\widetilde{T}$ and $\overline{T}$ we choose the one of the largest weight and extend it to a Hamiltonian cycle H.

It turns out that $c(H) \geq \frac{3}{4}c(H^*)$, where H^* is a maximum weight Hamiltonian cycle in G. Suppose that H^* contains both e_1 and e_2. Replacing e_1 and e_2 by $\overline{e}$, we obtain a Hamiltonian cycle in $\overline{G}$ and hence by Part 1 we have $c(H) \geq c(\overline{T}) \geq \frac{3}{4}c(H^*)$. Suppose that H^* contains at most one edge from the set $\{e_1, e_2\}$ and let e^* be an edge of the minimum weight in H^*. Since H^* contains at least one edge incident to v and not in the cycle S_1, we must have $c(e) \geq c(e^*)$. Then the required estimate follows from the inequalities

$$c(F) \geq c(H^*) \quad \text{and} \quad c(M) \geq \frac{c(H^*) - c(e^*)}{2}$$

and

$$c(H) \geq c(\widetilde{T}) \geq \frac{c(F) + c(M) + c(e)}{2}.$$

The latter inequality is obtained as in Part I.

Hassin and Rubinstein in [439] combined the ideas of their earlier paper [438] with those of Serdyukov to obtain a randomized algorithm, which, for any fixed $\rho < 25/33$, achieves an expected performance ratio of at least ρ in $O(n^3)$ time.

5.2. The Idea of the Algorithm from [439]

The algorithm constructs three "candidate" Hamiltonian cycles H_1, H_2 and H_3 and chooses the one with the largest weight. As in Section 5.1, a maximum weight 2-factor $F = S_1 \cup \ldots \cup S_l$ is constructed. Given a $\rho < 25/33$, a number $\epsilon > 0$ is computed (ϵ approaches 0 when ρ approaches 25/33) and the cycles $S_1, \ldots, S_l$ are assigned to be "long" or "short", depending on whether the number of vertices in the cycle is greater or less than ϵ^{-1}. Then, for each short cycle, a maximum weight Hamiltonian path on its vertices is constructed (using dynamic programming, one can do it in $O(n^2 2^{1/\epsilon})$ time). From each long cycle a minimum weight edge is excluded. All paths obtained (from both short and long cycles) are extended into a Hamiltonian cycle H_3.

The cycles H_1 and H_2 are obtained by extending partial tours T_1 and T_2, which are constructed in a somewhat similar way as in Section 5.1. As in Section 5.1, a maximum weight perfect matching M is constructed. Furthermore, a maximum weight perfect matching M' is constructed on the edges with the endpoints in different cycles $S_1, \ldots, S_l$. Then, from each cycle S_i roughly half of the edges are assigned to T_1 (randomization is involved in choosing which half) and the rest is assigned to T_2. After that, edges of M' are used to patch T_2 (randomization is used here as well).

The gain in the performance ratio compared to the "pure" algorithm from Section 5.1 comes from the observation that if the optimal Hamiltonian cycle follows cycles of F closely, then H_3 alone provides a good approximation. If, however, the optimal cycle does not follow F closely, it has to use sufficiently many edges from M'. The algorithm is randomized and it is proved that the expected weight of the produced cycle is at least ρ times the maximum weight of a Hamiltonian cycle.

One can show that for any prescribed probability $p < 1$ of success, after running the algorithm independently $O\big(n/(1-p)\big)$ times, one obtains the desired cycle with the probability at least p.

We note also that the complexity bound includes an $O(n^2 2^{1/\epsilon})$ term which is dominated by $O(n^3)$ for any fixed $\epsilon > 0$ but quickly takes over if ϵ is allowed to approach 0, that is, if we want the performance ratio to approach 25/33.

6. The Semimetric MAX TSP

In this section, we discuss Problem P2. Note, that we do not assume the symmetry condition of P1. Again, it is convenient to restate the problem of as the problem of finding a maximum weight Hamiltonian cycle in a (this time directed) weighted complete graph $G = (V, A)$ with $|V| = n$ vertices and $|A| = n(n-1)$ arcs.

The following result is crucial.

Theorem 2 *Let $G = (V, A)$ be a directed graph with $|V| = n$ vertices and non-negative weights $c(a)$ on its arcs. Let $F = S_1 \cup \ldots \cup S_l$ be a 1-factor of G. Let m_i be the number of vertices in S_i, so $m_1 + \ldots + m_l = n$. Let $m = \min\{m_i : 1 \le i \le l\}$. Then there exist Hamiltonian cycles H_1 and H_2 in G whose weights satisfy the inequalities*

$$c(H_1) \ge \left(1 - \frac{1}{n}\right)^{l-1} c(F) \tag{2}$$

and

$$c(H_2) \ge \left(1 - \frac{1}{2m}\right) c(F). \tag{3}$$

Moreover, given a 1-factor F of G, Hamiltonian cycles H_1 and H_2 can be constructed in $O(mn)$ time.

Inequality (2) was obtained in [754] and inequality (3) was obtained in [517].

Sketch of Proof. To construct H_1, we repeatedly patch two cycles of the factor into one, thus reducing the number of cycles by 1 until we reach a Hamiltonian cycle. In doing so, we arrange current cycles S_i in a non-decreasing order of the weight per vertex ratios $c(S_i)/m_i$ and always patch the first two cycles, say S_1 and S_2.

Let us fix an arc of S_2, say $(1, 2)$. We delete arc $(1, 2)$ of S_2 and a certain arc (x, y) of S_1 and then add arcs $(1, y)$ and $(x, 2)$, thus obtaining a new cycle $S_1 * S_2$.

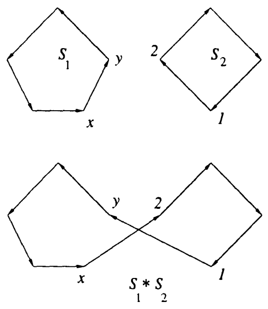

Figure 12.3. Patching cycles

Using the triangle inequality $c(1, 2) \leq c(1, z) + c(z, 2)$ for every vertex z of S_1, one can deduce that the average weight $c(S_1 * S_2)$ over all possible choices of arcs (x, y) of S_1 is at least as big as $c(S_1) + c(S_2) - c(S_1)/m_1$. Hence if we choose (x, y) so as to maximize the weight of $c(S_1 * S_2)$, we get

$$c(S_1 * S_2) \geq c(S_1) + c(S_2) - c(S_1)/m_1,$$

from which it follows that the weight of the 1-factor multiplies by at least $(1 - 1/n)$ after each patch (we recall that S_1 has the lowest weight per vertex ratio, so $c(S_1)/m_1 \leq c(F)/n$).

To construct H_2, we find two families $P_1, \ldots, P_m$ and $Q_1, \ldots, Q_m$ of Hamiltonian cycles such that

$$\sum_{i=1}^{m} c(P_i) + \sum_{i=1}^{m} c(Q_i) \geq (2m - 1) \sum_{i=1}^{l} c(S_i). \tag{4}$$

Then H_2 is a maximum weight cycle from the collections $P_1, \ldots, P_m$ and $Q_1, \ldots, Q_m$. Each cycle P_i or Q_j uses all but one arc from every cycle S_i and some additional arcs bundling the cycles together. Let us choose a cyclical order $S_1, \ldots, S_l, S_1$ on the cycles S_i and a cyclical order on the vertices of each cycle S_i compatible with the direction of the cycles. To construct P_j, from each cycle S_i we remove the arc coming into the j-th vertex and connect the $(j-1)$-st vertex of the next cycle with the j-th vertex of S_i.

To construct Q_j, we delete the arc coming into the j-th vertex of S_1 and then for each next cycle we remove the arc coming into the vertex whose index drops by 1 each time we pass to a new cycle. New arcs connecting each cycle with the next one are added to make a Hamiltonian cycle.

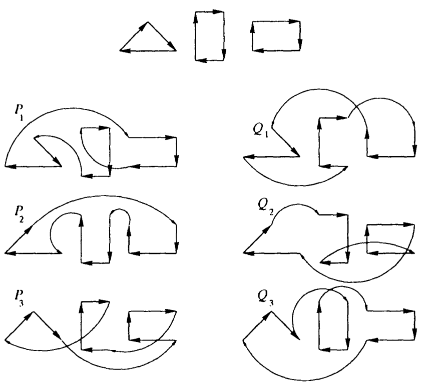

Figure 12.4. Constructing P_i and Q_j

Inequality (4) is obtained by the repeated application of the triangle inequality.

Since we can construct a maximum weight 1-factor with $m \geq 2$ in $O(n^3)$ time, by (3) we get a polynomial time algorithm with the performance ratio 3/4 [517]. The construction of H_1 and inequality (2) will be later used in Section 9.

7. The Metric MAX TSP

When both the triangle inequality of P2 and the symmetry condition of P1 are satisfied, the performance ratio can be improved further. Indeed, constructing a maximum weight 2-factor in $O(n^3)$ time, (see Appendix A), we have $m \geq 3$ in inequality (3) of Theorem 2. Hence we get a polynomial time algorithm with the performance ratio of 5/6 [517]. The same performance ratio is obtained in [521]. As in (3), the performance ratio of the Kovalev and Kotov algorithm is expressed in terms of the minimum number m of vertices in a cycle of the underlying maximum weight 2-factor. It turns out to be equal to $(m+2)/(m+3)$, which agrees with (3) for $m = 3$ but is somewhat weaker for $m > 3$.

As this survey was nearing completion, we learned that recently Hassin and Rubinstein in [440] designed a randomized algorithm for the metric MAX TSP with $O(n^3)$ complexity and a 7/8 expected performance ratio.

8. TSP with Warehouses

In this section, we discuss Problem P4. Hence we assume that we are given $r \times n$ matrices u and v with the weight matrix c represented in the form (1). As we already mentioned in Section 1, by switching "max" to "min" we get corresponding results for the MIN TSP and we do not need to assume that c is non-negative.

Our main result is a polynomial time algorithm from [90] for the case when r is fixed. Note that in [90], the matrix c is assumed to be symmetric, and the corresponding problem is referred to as the *Tunneling* TSP, as warehouses are replaced by tunnels connecting cities (we also note that if the Traveling Salesman travels by air, the role of the warehouses is naturally played by airline hubs). The symmetry condition is not crucial (the earlier version of the paper treats the general case) although it leads to some improvement of the complexity estimates. Here we discuss the general, not necessarily symmetric case.

8.1. Sketch of the Algorithm from [90]

Let us consider a graph $G = (V, E)$ with $n + r$ nodes. The set V of nodes is the union $V = C \cup W$ of $n = |C|$ nodes, called *cities* and $r = |W|$ nodes called *warehouses.* Each city is connected to each warehouse by two arcs, one, from the city to the warehouse, colored red and the other, from the warehouse to the city, colored blue. The weight of the red arc connecting the i-th city and the k-th warehouse is u_{ki} whereas the weight of the blue arc is v_{ki}. We are looking for a closed walk in G of the maximum total weight, which visits every city exactly once. The problem reduces to finding a maximum weight subset $E' \subseteq E$ which satisfies the following three conditions:

(8.1.1) every city is incident to exactly two arcs in E', one red and one blue;

(8.1.2) for every warehouse, the number of incident red edges in E' is equal to the number of incident blue edges in E';

(8.1.3) the subgraph of G induced by E' is connected.

Then an Euler tour through E' will produce the desired solution. Since the set E' must be connected, all visited warehouses must be connected in E'. The idea is to enumerate all possible *minimal ways* to connect warehouses, that is, we enumerate all minimal connected subgraphs of G containing a subset of W. Since such a connected subgraph is a tree and contains at most $2r-2$ edges, the direct enumeration of all possible minimal connected subgraphs $F \subseteq E$ takes $O(n^{2r-2})$ time, which is polynomial in n if r is fixed. Hence we enumerate all possible connected subgraphs F and for each such an F, we find a subset $E'_F \subseteq E$ of the maximum weight which contains $E(F)$ and satisfies (8.1.1)–(8.1.2). The problem of finding E'_F reduces to solving of $O(n^r)$ maximum weight f-factor problems, since the only conditions we have to satisfy on the edges of $E'_F \setminus E(F)$ are the degree constraints on the cities and warehouses enforced by (8.1.1)–(8.1.2), cf. Section 3. Finally, we choose E' of the maximum weight among all E'_F.

The resulting complexity of the algorithm is $O(n^{3r+1})$, some ways to improve it are discussed in [90] and [87].

The following simple observation turns out to be quite useful (we use it in Section 9 comparing two different approaches to the MAX TSP in a space with a polyhedral norm).

Lemma 3 *Suppose that the combinatorial rank of the weight matrix c is r. Then there exists a maximum weight 1-factor F of the complete directed graph on the vertex set containing not more than r cycles. Moreover, if c is given in the form (1) the 1-factor F can be constructed in $O(n^3)$ time.*

Sketch of Proof. We have

$$c_{ij} = \max_{k=1,\ldots,r} (u_{ki} + v_{kj})$$

for some $r \times n$ matrices u and v. For each pair (i,j) of cities let us identify a warehouse $k = k(i,j)$ such that $c_{ij} = u_{ki} + v_{kj}$.

Suppose that a maximum weight 1-factor contains more than r cycles. Then there will be two arcs (s,t) and (p,q) of different cycles that correspond to the same warehouse k. Let us patch the cycles by deleting (s,t) and (p,q) and inserting (s,q) and (p,t). The weight of the factor can only increase since

$$c_{sq} + c_{pt} \geq u_{ks} + v_{kq} + u_{kp} + v_{kt} = c_{st} + c_{pq}. \tag{5}$$

Some other special classes of matrices c for which the MAX TSP is polynomially solvable (in particular, via the "pyramidal tour" approach) are surveyed in Chapter 11 and [142]. In [112] it is shown that if c is an $n \times n$ non-negative matrix and $c_{ij} = 0$ for all i, j with $|i - j| \neq 1$, then a maximum weight tour can be found in $O(n)$ time.

9. MAX TSP in a Space with a Polyhedral Norm

Let $\mathbb{R}^d$ be Euclidean space endowed with the standard scalar product $\langle \cdot, \cdot \rangle$. Let us fix some non-zero vectors $a_1, \dots, a_r \in \mathbb{R}^d$ and let

$$\mathbf{B} = \left\{ x \in \mathbb{R}^d : \langle a_i, x \rangle \leq 1 : i = 1, \dots, r \right\}. \tag{6}$$

Hence $\mathbf{B}$ is a full-dimensional polyhedron, which contains the origin in its interior. We do not assume that $\mathbf{B}$ is symmetric about the origin but it is convenient, although not necessary, to assume that $\mathbf{B}$ is bounded. Given $\mathbf{B}$, let us define the Minkowski functional $\| \cdot \|$ by

$$\|x\| = \min\{\lambda \geq 0 : x \in \lambda B\}.$$

Hence $\| \cdot \|$ defines a distance function $d(x, y) = \|y - x\|$.

Given n points $p_1, \dots, p_n \in \mathbb{R}^d$, let us consider their distance matrix $c = (c_{ij})$, $c_{ij} = \|p_j - p_i\|$. One can observe that the combinatorial rank of c (see Section 1) is bounded above by the number r of inequalities defining $\mathbf{B}$ in (6). Moreover, one can represent c in the form (1) provided the vectors $a_1, \dots, a_r$ in (6) are given. Indeed,

$$\begin{aligned} c_{ij} = \|p_j - p_i\| &= \min\left\{ \lambda \geq 0 : p_j - p_i \in \lambda \mathbf{B} \right\} \\ &= \min\left\{ \lambda \geq 0 : \langle a_k, p_j - p_i \rangle \leq \lambda : \quad k = 1, \dots, r \right\} \\ &= \max\left\{ \langle a_k, p_j - p_i \rangle : k = 1, \dots, r \right\} = \max_{k=1,\dots,r} (u_{ki} + v_{kj}), \end{aligned}$$

where $u_{ki} = -\langle a_k, p_i \rangle$ and $v_{kj} = \langle a_k, p_j \rangle$.

Geometrically, the combinatorial rank of the matrix of pairwise distances between points in a polyhedral norm does not exceed the number of facets of the unit ball. This observation is from [86] and, as mentioned there, is joint with S. Onn and A. Gerards. In the context of Section 8, facets of the unit ball $\mathbf{B}$ correspond to warehouses.

Note that, if $\mathbf{B}$ is unbounded, we have to write

$$c_{ij} = \max\left\{ 0, \langle a_k, p_j - p_i \rangle : k = 1, \dots, r \right\}$$

and the combinatorial rank of c may increase by 1.

Using results of Section 8, we get a polynomial time algorithm for solving the MAX TSP problem in polyhedral norm, provided the number of facets r of the unit ball is fixed [90]. When the unit ball $\mathbf{B}$ is symmetric about the origin and hence $\|\cdot\|$ is a genuine norm, the complexity $O(n^{r-2}\ln n)$ can be achieved, see [90] and [87]. Particularly interesting cases are those of the L^1 norm in the plane $\mathbb{R}^2$ (the so called "Manhattan norm" alluding to the other name of the MAX TSP, the "taxicab ripoff" problem) and the L^∞ norm in the plane, for both of which Fekete in [285] obtained an algorithm of linear complexity $O(n)$.

A different approach was suggested by Serdyukov in [755]. The algorithm in [755] is based on the following observation.

Lemma 4 *Let $\mathbf{B} \subset \mathbb{R}^d$ be a bounded polyhedron, given by (6) and let $\|\cdot\|$ be the corresponding Minkowski functional. For points $p_i, p_j \in \mathbb{R}^d$, let $c_{ij} = \|p_j - p_i\|$. Then*

$$c_{12} + c_{23} \geq c_{13} \tag{7}$$

for any three points p_1, p_2 and p_3.

Suppose that for $k = 1, \ldots, r$, the set $G_k = \{x \in \mathbf{B} : \langle a_k, x\rangle = 1\}$ is a facet of $\mathbf{B}$ and let Γ_k be the cone with the vertex at the origin spanned by G_k. Suppose further, that p_1, p_2 and p_3, p_4 are points such that for some k, one has $p_2 - p_1 \in \Gamma_k$ and $p_4 - p_3 \in \Gamma_k$. Then

$$c_{12} + c_{34} \leq c_{14} + c_{32}. \tag{8}$$

Indeed, (7) is standard and follows from the convexity of $\mathbf{B}$. Inequality (8) is, essentially, inequality (5) of Section 8, since if $p_j - p_i \in \Gamma_k$ then $\|p_j - p_i\| = \langle a_k, p_j - p_i\rangle$ and the arc (p_i, p_j) corresponds to the k-th warehouse.

9.1. Sketch of the Algorithm from [755]

Given points $p_1, \ldots, p_n \in \mathbb{R}^d$ and the weight matrix $c_{ij} = \|p_j - p_i\|$, we construct a maximum weight 1-factor $F = S_1 \cup \ldots \cup S_l$. The idea is to show that the number l of cycles can be reduced to at most $\lfloor r/2 \rfloor$, where r is the number of facets of $\mathbf{B}$.

Indeed, suppose that $l > \lfloor r/2 \rfloor$. Then there are two arcs (x_1, x_2) and (y_1, y_2) which belong to different cycles of F and such that $x_2 - x_1, y_2 - y_1 \in \Gamma_k$ for some cone Γ_k spanned by a facet of $\mathbf{B}$ (note that the cones Γ_k cover the whole space $\mathbb{R}^d$ and that each cycle has arcs from at least *two* cones Γ_k). Now we patch the cycles by deleting (x_1, x_2) and (y_1, y_2) and inserting (x_1, y_2) and (y_1, x_2). Lemma 4 implies that the weight of the 1-factor can not become smaller.

Repeating this procedure, we obtain a maximum weight 1-factor with at most $\lfloor r/2 \rfloor$ cycles. Up to this point, the construction goes as in Lemma 3 of Section 8, only cutting the required number of cycles by half because of a special geometric feature of the problem.

Now we construct a Hamiltonian cycles H_1 as in Theorem 2, Section 6 (note that by inequality (7), the weight function satisfies the triangle inequality).

Clearly, the complexity of the algorithm is $O(n^3)$, dominated by the complexity of constructing a maximum weight 1-factor. As follows by (2), Section 6, the weight of the constructed Hamiltonian cycle is at least $(1 - 1/n)^{\lfloor r/2 \rfloor - 1}$ times the maximum weight of a Hamiltonian cycle.

It is interesting to compare algorithms of [90] and [755]. While algorithm from [755] does not solve the problem exactly except in the case of the norm in the plane $\mathbb{R}^2$ whose unit ball is a triangle, its relative error is asymptotically 0 for $n \longrightarrow +\infty$ as long as $r = o(n)$ and its complexity is linear in the number r of facets of the unit ball $\mathbf{B}$. On the other hand, the algorithm from [90] gives an exact solution, but its complexity is exponential in the number of facets of $\mathbf{B}$.

10. MAX TSP in a Normed Space

Given a norm in $\mathbb{R}^d$, we can always approximate it by a polyhedral norm within an arbitrarily small relative error. Hence each of the two approaches described in the previous section leads to a fully polynomial approximation scheme for the MAX TSP in a finite-dimensional vector space with a fixed norm, see [86].

Let us consider the case of the Euclidean norm $\|x\| = \sqrt{\xi_1^2 + \ldots + \xi_d^2}$ for $x = (\xi_1, \ldots, \xi_d)$ in some more detail. Assuming that d is fixed, to approximate the norm $\|\cdot\|$ by a polyhedral norm within relative error ϵ, we need to approximate the unit ball $\mathbf{B}$ in $\mathbb{R}^d$ by a polyhedron $\mathbf{P}$ such that

$$\mathbf{P} \subset \mathbf{B} \subset (1 + \epsilon)\mathbf{P}.$$

Such a polyhedron $\mathbf{P}$ will have $O(\epsilon^{-d})$ facets, so the complexity of the algorithm from [90] will be $n^{O(\epsilon^{-d})}$. However, since the complexity of algorithm of Section 9.1 depends linearly on the number of facets of $\mathbf{P}$, by Serdyukov's approach, we get an algorithm of $O(n^3 + \epsilon^{-d})$ complexity and relative error $\epsilon + O(\epsilon^{-d}/n)$. Thus for any *fixed* $\epsilon > 0$ the algorithm of [755] achieves asymptotically the relative error ϵ with the complexity $O(n^3)$ as the number n of points grows to infinity.

Using a different approach, in the case of Euclidean norm, Serdyukov in [753] proposed a method for which the relative error is asymptotically 0, as n grows. The approach is based on the following geometric result.

Lemma 5 *Let $\{I_1, \ldots, I_l\}$ be a set of l straight line intervals in $\mathbb{R}^d$. Then the smallest angle between a pair of intervals from $\{I_1, \ldots, I_l\}$ is bounded from above by a constant $\alpha(d,l)$ such that $\lim_{l \longrightarrow +\infty} \alpha(d,l) = 0$. Moreover,*

$$\cos \alpha(d,l) \geq 1 - \gamma(d) l^{\frac{-2}{d-1}}$$

for some constant $\gamma(d)$ independent on the number of intervals.

Intuitively, if we have many intervals in a space of low dimension, some of them will be nearly parallel. We denote by $\alpha(I_i, I_j)$ the angle between I_i and I_j.

The proof of Lemma 5 can be obtained as follows. Without loss of generality, we may assume that the intervals $I_1, \ldots, I_l$ are diameters of the unit sphere $S^{d-1} \subset \mathbb{R}^d$ (that is, each interval is symmetric about the origin and has length 2). Representing a diameter by its endpoints on the sphere, we observe that for every interval I, the set of all diameters J of S^{d-1} such that $\alpha(I, J) \leq \beta$ is represented by a set $C(I, \beta)$, which is a pair of antipodal spherical caps centered at the endpoints of I and of radius β in the intrinsic (geodesic) metric of the sphere. Using well-known formulas for spherical volumes, we estimate the smallest β for which some two sets $C(I_i, \beta)$ and $C(I_j, \beta)$ are bound to intersect. Then $\alpha(I_i, I_j) \leq 2\beta$.

10.1. Sketch of the Algorithm from [753]

Step 1. Given n points $p_1, \ldots, p_n$ in $\mathbb{R}^d$ with the weight matrix $c_{ij} = \|p_j - p_i\|$, we construct a maximum weight matching. Hence we get a set M of $m = \lfloor n/2 \rfloor$ straight line intervals in $\mathbb{R}^d$.

Step 2. Let us choose a number $l \leq m/2$ (to be specified later). Let us choose some l intervals in M of the smallest weight and call them *light*. All other intervals in M we call *heavy*. Let $I_1, \ldots, I_m$ be an ordering of the intervals from M. We call a subsequence $R = (I_i, I_{i+1}, \ldots, I_{i+k})$, $k \geq 0$, a *run* if R is a maximal subsequence with the properties that all intervals in R are heavy and the angle between each pair I_j, I_{j+1} of consecutive intervals in R does not exceed $\alpha(d,l)$, cf. Lemma 5. In particular, a subsequence consisting of a single heavy interval may constitute a run. Our goal is to construct an ordering $I_1, \ldots, I_m$ with not more $l-1$ runs. To achieve that, we start with an arbitrary ordering $I_1, \ldots, I_m$ and identify all runs. We observe that if the number of runs exceeds $l-1$, by Lemma 5, we can find a run R_1 with the rightmost interval I_i and a run

R_2 with the rightmost interval I_j such that $\alpha(I_i, I_j) \leq \alpha(d, l)$. Then, by a proper reordering of the intervals, we merge R_1 and R_2 into a larger run R. After repeating this procedure not more than, say, m times, we get an ordering with not more than $l - 1$ runs.

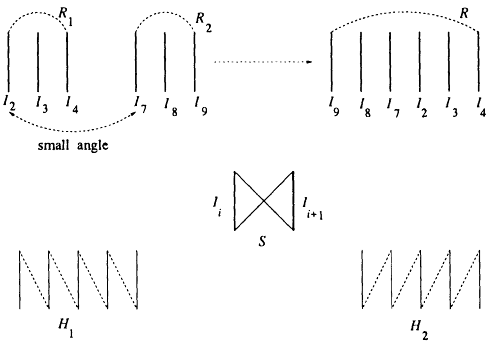

Figure 12.5. Patching runs into Hamiltonian cycles

Step 3. For every pair of consecutive intervals I_i and I_{i+1}, we find a pair of edges patching them into a maximum weight 4-cycle S. Let E be the set of all newly constructed edges. Using edges from E we consecutively patch the intervals $I_1, \ldots, I_m$ into a Hamiltonian cycle H. The idea is to choose H to be the best of the two main types of patching H_1 and H_2, see Figure 4. More precisely, in [753], Serdyukov constructs four cycles, two having the H_1 type and two having the H_2 type, to take care of "boundary effects", and chooses the best of the four.

The algorithm achieves $O(n^3)$ complexity, again dominated by the complexity of constructing a maximum weight matching. It turns out that if l is chosen to be $\lfloor n^{\frac{d-1}{d+1}} \rfloor$, then the algorithm achieves $1 - \beta(d) n^{\frac{-2}{d+1}}$ performance ratio, where $\beta(d)$ is a constant depending on the dimension alone. We observe that as long as d is fixed, the performance ratio approaches 1 as the number n of points increases.

The analysis is based on the following observations. The total weight of heavy intervals is at least $c(M)(1 - l/m)$, that is, asymptotically equal to the weight $c(M)$ of the matching M. Hence it suffices to analyze how does weight $c(H)$ relate to the total weight of heavy intervals in M. If the intervals I_i and I_{i+1} had been parallel, we would have had $c(S) \geq 2c(I_1) + 2c(I_2)$ for the optimal patching S of I_1 and I_2 into a 4-cycle. If I_i and I_{i+1} are nearly parallel, $c(S)$ is guaranteed to be almost twice the weight $c(I_1) + c(I_2)$ and an exact bound can be obtained in terms of the angle $\alpha(I_1, I_j)$. Let us consider the ordering $I_1, \ldots, I_m$ obtained after Step 2 is performed. Let $R = (I_i, I_{i+1}, \ldots, I_k)$ be a run and let P_i be the path obtained from R in H_i, $i = 1, 2$. It follows that the average $\frac{c(P_1) + c(P_2)}{2}$ is guaranteed to be almost twice the total weight of the intervals in R. Since the number of runs is small, "boundary effects" are asymptotically negligible and we conclude that the average weight of the H_1 and H_2 patchings is almost as large as $2c(M)$. Since for any Hamiltonian cycle H^* we have $2c(M) \geq c(H^*)$ (for n even) and $2c(M) \geq c(H^*)(1 - 1/n)$ (for n odd), we conclude that H should be asymptotically optimal.

Exploiting geometric duality between minimum stars and maximum matchings in [286] and [287] Fekete et al. obtained a practical near-linear heuristic in the case of Euclidean norm in $\mathbb{R}^2$ that has a worst-case guarantee of $2/\sqrt{3} \approx 1.15$. It is interesting to note [784] that the standard linear programming relaxation of a maximum matching problem for an even number of points has always integral optima in the case of a norm in $\mathbb{R}^2$ and hence can be solved by a network flow algorithm, whereas the general maximum matching algorithm of $O(n^3)$ complexity becomes impractical for large n.

11. Probabilistic Analysis of Heuristics

In this section, we assume that instances of the MAX TSP are sampled at random from some probability space. We discuss how some obvious heuristics, "farthest neighbor" (FN) and "subtour patching" (SP), typically behave. Such heuristics produce a Hamiltonian cycle, whose weight often approximates the maximum weight of a Hamiltonian cycle reasonably well.

11.1. Asymptotic Optimality

Let $\mathcal{K}_n$ be a probability space of MAX TSP instances with n cities. We say that an algorithm (heuristic) has an (ϵ_n, δ_n) performance guarantee on problems from $\mathcal{K}_n$ if the probability that the weight of the produced Hamiltonian cycle approximates the maximum weight within relative error ϵ_n is at least $1 - \delta_n$. If $\mathcal{K}_n$, $n = 1, 2, \ldots$, is a series of probability spaces of MAX TSP instances of increasing sizes, we say that the algorithm is asymptotically optimal if one can choose performance guarantees (ϵ_n, δ_n) is such a way that $\epsilon_n \longrightarrow 0$ and $\delta_n \longrightarrow 0$ as $n \longrightarrow +\infty$.

11.2. The Farthest Neighbor Heuristic

Given a weight matrix $c = (c_{ij})$, we start with any vertex, go to the farthest, then to the farthest remaining and so on. The following result is obtained in [362] and [364].

Theorem 6 *Suppose that the weights c_{ij} are sampled independently at random from some distribution in the interval $[a_n, b_n]$ with $b_n > a_n > 0$. Let $P(x) = \mathbf{P}\{\xi < x\}$ be the distribution function of a random variable $\xi = (c_{ij} - a_n)/(b_n - a_n)$.*
Let

$$\psi(n) = \int_0^{1-1/n} \frac{dx}{1 - P(x)}.$$

Suppose that at least one of the following holds:
1) We have $\psi(n) \longrightarrow +\infty$ and $\psi(n)/n \longrightarrow 0$ as $n \longrightarrow +\infty$;
2) We have $P(x) > x$ for all x and $\psi(n) = o(n)$;
3) We have $P(x) \leq x$ for all x.
Then FN is asymptotically optimal.

It follows that FN is asymptotically optimal for any convex distribution and for the uniform distribution without any additional assumptions. This is in contrast to the "Nearest Neighbor" heuristic for the MIN TSP, see [365] and [366] and Chapter 6.

The proof of Theorem 6 goes along the following lines: we estimate the expected weight of the constructed Hamiltonian cycle, use Chebyshev's inequality to show that a typical weight is sufficiently close to the expectation and use a trivial upper bound nb_n on the largest weight of a Hamiltonian cycle (it turns out that the expected weight is sufficiently close to nb_n).

In the case of the uniform distribution $P(x) = x$, Vohra in [811] proved that for any $\delta_n > 0$ one can choose $\epsilon_n = O(\sqrt{n^{-1}\ln(1/\delta_n)})$ so that (ϵ_n, δ_n) is a performance guarantee for the heuristic. One can get a better bound $\epsilon_n = O(n^{-1}\ln(1/\delta_n))$ by using large deviation inequalities. Letting $\delta_n = n^{-\lambda/2}$, where λ is any positive constant, we have

$$\epsilon_n = \frac{\lambda \ln n}{n}, \quad \delta_n = n^{-\lambda/2},$$

which implies asymptotically optimality of the farthest city heuristic. Moreover, this result is valid for a wider class of distributions where $P(x) \leq x$.

11.3. The Subtour Patching Heuristic

Given a complete directed graph on n vertices, suppose that the arc weights c_{ij} are chosen independently at random from some distribution.

The subtour patching heuristics starts with constructing a maximum weight 1-factor and then patches the subtours into a tour. The procedure turns out to be quite efficient under some mild assumption about the distribution. Essentially, we do not even need the weights c_{ij} to be independent, the following two conditions will be sufficient:

(11.3.1) The resulting distribution on the optimal 1-factors is uniform;

(11.3.2) Individual weights c_{ij} are not very large compared to sums of several weights.

It is well known known (see, for example, Sections 7–9 of [510]), a random permutation of n elements with probability very close to 1 will have $O(\ln n)$ cycles. Therefore, (11.3.1) implies that the resulting maximum weight subtour cover with the overwhelming probability will have $O(\ln n)$ subtours. The condition (11.3.2) implies that the patching procedure does not "skew" the weight of the tour too much.

For example, in [363] it was shown that if (in the directed case) the columns of the distance matrix form a sequence of *symmetrically dependent* random variables (we recall that random variables $X_1, \ldots, X_m$ are called symmetrically dependent if the distribution of the vector $(X_{\sigma(1)}, \ldots, X_{\sigma(m)})$ does not depend on the permutation σ of the set $\{1, \ldots, m\}$), then the MAX TSP can be solved in $O(n^3)$ time with the performance estimates

$$\epsilon_n = 2c^* \ln n / F^*_{AP}, \quad \delta_n = (e/n)^{0.38},$$

where c^* is the maximum entry of the distance matrix and F^*_{AP} is the optimal value of the objective function in the maximum assignment problem. Hence for $0 < a_n \le c_{ij} \le b_n$ with $b_n/a_n = o(n/\ln n)$ the subtour patching algorithm is asymptotically optimal.

Results similar to the ones described in Sections 11.2 and 11.3 can be obtained for discrete distributions, e.g. for FN see [364]. We note also that Dyer, Frieze and McDiarmid [263] presented an asymptotically optimal linear-time partitioning heuristic when the points are chosen uniformly in the unit square. They also computed the expected value of the longest Hamiltonian cycle through n random points which turned out to be approximately equal to $0.7652n$.