When is the Assignment Bound Tight for the Asymmetric Traveling-Salesman Problem ?

Alan Frieze*
Carnegie-Mellon University

Richard M. Karp†
University of California, Berkeley and
International Computer Science Institute, Berkeley, Calif.

Bruce Reed
University of Bonn

Abstract

We consider the probabilistic relationship between the value of a random asymmetric traveling salesman problem $ATSP(M)$ and the value of its assignment relaxation $AP(M)$. We assume here that the costs are given by an $n \times n$ matrix M whose entries are independently and identically distributed. We focus on the relationship between $Pr(ATSP(M) = AP(M))$ and the probability p_n that any particular entry is zero. If $np_n \to \infty$ with n then we prove that $ATSP(M) = AP(M)$ with probability 1-o(1). This is shown to be best possible in the sense that if $np(n) \to c$, $c > 0$ and constant, then $Pr(ATSP(M) = AP(M)) < 1 - \phi(c)$ for some positive function ϕ. Finally, if $np_n \to 0$ then $Pr(ATSP(M) = AP(M)) \to 0$.

1 Introduction

The *Assignment Problem (AP)* is the problem of finding a minimum-weight perfect matching in an edge-weighted bipartite graph. An instance of the AP can be specified by an $n \times n$ matrix $M = (m_{ij})$; here m_{ij} represents the weight of the edge between x_i and y_j, where $X = \{x_1, x_2, \ldots, x_n\}$ is the set of "left vertices" in the bipartite graph, and $Y = \{y_1, y_2, \ldots, y_n\}$ is the set of "right vertices."The AP can be stated in terms of the matrix M as follows: find a permutation σ of $\{1, 2, \ldots, n\}$ that minimizes $\sum_{i=1}^{n} m_{i,\sigma(i)}$.Let $AP(M)$ be the optimal value of the instance of the AP specified by M.

The *Asymmetric Traveling-Salesman Problem (ATSP)* is the problem of finding a Hamiltonian circuit of minimum weight in an edge-weighted directed graph. An instance of the ATSP can be specified by an $n \times n$ matrix $M = (m_{ij})$ in which m_{ij} denotes the weight of edge $< i, j >$. The ATSP can be stated in terms of the matrix M as follows: find a cyclic permutation π of $\{1, 2, \ldots, n\}$

*Research supported by NSF grant CCR-9024935
†Research supported by NSF Grant CCR-9017380

that minimizes $\sum_{i=1}^{n} m_{i,\pi(i)}$; here a cyclic permutation is one whose cycle structure consists of a single cycle. Let $ATSP(M)$ be the optimal value of the instance of the ATSP specified by M.

It is evident from the parallelism between the above two definitions that $AP(M) \leq ATSP(M)$. The ATSP is NP-hard, whereas the AP is solvable in time $O(n^3)$. Several authors (for a recent survey see [BaTo]) have investigated whether the AP can be used effectively in a branch-and-bound method to solve the ATSP.

The most striking evidence of the power of this approach is given by the recent work of Miller and Pekny. Among many other computational results, they obtained optimal solutions to random instances with up to 500,000 cities, in which the m_{ij} were drawn independently from the integers in the range $[0, n]$. Miller and Pekny noticed that $AP(M)$ was often equal to $ATSP(M)$, and they exploited this observation by developing a special method to search for a cyclic permutation among the optimal solutions to the AP.

Motivated by the computational experience of Miller and Pekny, we have investigated the following question: when the m_{ij} are drawn independently from a common distribution (over, say, the nonnegative reals), what is the probability that $AP(M) = ATSP(M)$? The answer depends on the probability that an entry is zero. We show that, if the expected number of zeros in a row of M tends to infinity as $n \to \infty$ then the probability that $AP(M) = ATSP(M)$ tends to 1, and we give an $O(n^3)$-time algorithm for finding an optimal solution to the ATSP with high probability; on the other hand, if the underlying distribution is uniform over the range of integers $[0..\lfloor c_n n \rfloor]$, where c_n tends to infinity with n, then the probability that $AP(M) = ATSP(M)$ tends to 0. Finally, we show that if the underlying distribution is uniform over the range of integers $[0..\lfloor cn \rfloor]$ where c is a positive constant, then the probability that $AP(M) = ATSP(M)$ does not tend to 1. We conjecture that for distributions of this type, the probability that $AP(M) = ATSP(M)$ tends to some positive constant less than 1 which depends on c.

The results of this paper are closely related to some earlier results of Karp [K], Karp and Steele [KS] and Dyer and Frieze [DF]. Here the $m_{i,j}$ are drawn independently from the uniform distribution over [0,1]. Karp showed that $ATSP(M)/AP(M) = 1-o(1)$ (whp) (we use the notation (whp) as shorthand for "with probability tending to1 as n tends to infinity.") Later, Karp and Steele and then Dyer and Frieze strengthened this result in several ways. For example the latter paper shows that the error term is $o((\log n)^4/n)$.

2 The Theorems

Theorem 1 *Let $\{X_n\}$ be a sequence of random variables over the nonnegative reals. Let $p_n = Pr[X_n = 0]$ and let $w(n) = np_n$. Let $M = M(n)$ be an $n \times n$ matrix whose entries are drawn independently from the same distribution as X_n. If $w(n) \to \infty$ as $n \to \infty$ then $AP(M) = ATSP(M)$ (whp).*

(Examination of the proof of Theorem 1 reveals that the distribution of non-zero's can be more complicated than actually stated. Indeed one can allow the costs to be generated as follows: start with an *arbitrary* real non-negative $n \times n$ matrix M. Randomly permute its rows and columns. Then for each $i, j \in [n]$ replace $M_{i,j}$ with zero, with probability p_n. There is also the proviso that the probability of two identical columns should tend to zero with n.)

Frieze [Fr] has shown that, if $w(n) = \ln n + t(n)$, where $t(n)$ tends to infinity, then $ATSP(M) = 0$

(whp), and so $AP(M) = ATSP(M)$ (whp). Thus, we restrict attention to the case where $w(n) = O(\ln n)$. The case where X_n has the uniform distribution over the range of integers $[0..N(n)]$ is particularly relevant to the Miller-Pekny computations. In this case, Theorem 1 tells us that $AP(M) = ATSP(M)$ (whp) provided that $N(n) = o(n)$.

Theorem 2 *Let $M = M(n)$ be an $n \times n$ matrix whose entries are drawn independently from the uniform distribution over $\{0, 1, ..., \lfloor cn \rfloor\}$ where c is a positive constant. Then, the probabilty that $AP(M) \neq ATSP(M)$ does not tend to zero as n tends to infinity.*

Theorem 3 *Let $M = M(n)$ be an $n \times n$ matrix whose entries are drawn independently from the uniform distribution over $\{0, 1, ..., \lfloor c_n n \rfloor\}$ where c_n tends to infinity with n. Then, the probability that $AP(M) \neq ATSP(M)$ tends to 1 as n tends to infinity.*

3 Proof of Theorem 1

We begin with some conventions and definitions. When n is understood from context we abbreviate p_n by p, $w(n)$ by w and $M(n)$ by M; also, "permutation" will mean "permutation of $\{1, 2, \ldots, n\}$".

Let H be the weighted bipartite graph with vertex set $X \cup Y$, where $X = \{x_1, x_2, \ldots, x_n\}$ and $Y = \{y_1, y_2, \ldots, y_n\}$, and with an edge of weight m_{ij} between x_i and y_j. Let G be the complete digraph on vertex set $\{1, 2, \ldots, n\}$,in which each edge $< i, j >$ has weight m_{ij}. A *cycle cover* is a subgraph of G in which each of the n vertices has in-degree 1 and out-degree 1. The AP can be stated in any of the following equivalent forms:

- Find a perfect matching of minimum weight in H;
- find a cycle cover of minimum weight in G;
- find a permutation σ to minimize $\sum_{i=1}^{n} m_{i,\sigma(i)}$.

Let the indicator variable z_{ij} be 1 if $m_{ij} = 0$ and 0 otherwise. Then the z_{ij} are independent, and each z_{ij} is equal to 1 with probability p. Emulating a useful trick due to Walkup [Wal1] we view the z_{ij} as being generated in the following way. Let h be defined by the equation $1 - p = (1 - h)^5$ and let z_{ij}^k, for $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n$ and $k = 1, 2, 3, 4, 5$, be independent indicator variables, each of which is equal to 1 with probability h. Let $z_{ij} = \max_{k=1}^{5} z_{ij}^k$. Then the z_{ij} are independent, and each is equal to 1 with probability p. For $k = 1, 2$, let H_k be the bipartite graph with vertex set $X \cup Y$, and with an edge between x_i and y_j if and only if $z_{ij}^k = 1$. For $k = 3, 4, 5$, let G_k be the digraph with vertex set $\{1, 2, \ldots, n\}$ and an edge from i to j if and only if $z_{ij}^k = 1$. The edges of G_3, G_4 and G_5, respectively, will be called *out-edges*, *in-edges* and *patch edges*. Each type of edge will play a special role in the construction of a Hamiltonian circuit of weight $AP(M)$. It will be important that the random graphs H_1 and H_2, and the random digraphs G_3, G_4 and G_5, are completely independent. Also, let $s(n) = nh(n)$; $s(n)$ is the expected degree of a vertex in H_1 or H_2, and the expected out-degree of a vertex in G_3, G_4 or G_5. Clearly, $s(n) \geq \frac{w(n)}{5}$, and thus $s(n)$ tends to infinity if $w(n)$ does.

The construction of the desired Hamiltonian circuit proceeds in the following stages:

- (Identification of "troublesome vertices.") By considering the edges of $H_1 \cup H_2$ identify a set $A \subset X$ and a set $B \subset Y$. The cardinality of $A \cup B$ is small (whp). The set $A \cup B$ contains the vertices of exceptionally small degree plus certain other vertices that are likely to be incident with edges of nonzero weight in an optimal assignment. At the same time construct a matching in H which is of minimum weight, subject to the condition that it covers the vertices in $A \cup B$ and no other vertices.
- Consider the subgraph of $H_1 \cup H_2$ induced by $(X \setminus A) \cup (Y \setminus B)$. This bipartite graph has a perfect matching (whp). Combining that perfect matching with the matching constructed in the previous step, obtain an optimal assignment for H in which every non-zero-weight edge is incident with a vertex in $A \cup B$.
- The optimal assignment just constructed has the properties of a random permutation.
- Using the out-edges and in-edges, attempt to convert the original optimal assignment into a permutation with no short cycles. This process succeeds (whp).
- Using the patch edges, patch the long cycles together into a single cycle, thus solving the $ATSP$. The patching process succeeds (whp).

The overall strategy of the proof is to construct an optimal assignment while keeping the in-edges, out-edges and patch edges (except those incident with $A \cup B$) in reserve for use in converting the optimal assignment to a tour. In the following sections we describe the algorithm in greater detail and give the proofs of the main assertions.

4 Identification of the Sets A and B

Consider the directed bipartite graph D with vertex set $X \cup Y$. The edges of D are those of H_1 directed from X to Y plus those of H_2 directed from Y to X. The expected out-degree of a vertex in D is $s(n)$, which we abbreviate by s. Let $d(v)$ be the out-degree of vertex v, let $N(v)$ be the set of out-neighbors of v and, for any set of vertices S, let $N(S)$ be the set of vertices adjacent from vertices in S.

We give an iterative construction for identifying a small set $A \cup B$ of vertices that are likely to be incident with edges of nonzero weight in an optimal assignment.

Let $W_{-1} = \{x \in X \cup Y | d(x) \leq s/2\}$. Let F_0 be a minimum weight matching in H which covers the vertices of W_{-1}. Let W_0 denote the set of vertices covered by F_0. Define a maximal sequence $(W_0, F_0), (W_1, F_1), \ldots, (W_r, F_r) = (W, F)$ where (W_i, F_i) is obtained from (W_{i-1}, F_{i-1}) as follows: suppose there exists $x \notin W_{i-1}$ such that $|N(x) \cap W_{i-1}| \geq s/4$. F_i is then a minimum weight matching in H which covers W_{i-1} and x and, necessarily, one other vertex y. We then take $W_i = W_{i-1} \cup \{x, y\}$. (F_i is obtained by constructing a least cost augmenting path from x w.r.t. F_{i-1}.)

Lemma 1 $|W| < 3ne^{-\frac{s}{5}}$ *(whp).*

Proof The proof follows from two simple claims. They can easily be justified by the first moment method and the calculations are omitted.

CLAIM 1: $|W_{-1}| \leq ne^{-s/5}$ (whp).

CLAIM 2: $S \subseteq X \cup Y, |S| \leq 3ne^{-s/5}$ implies that (whp), in $H_1 \cup H_2$, S contains fewer than $2|S|$ edges.

Assume the conditions of the above two claims. Then $|W_0| \leq 2|W_{-1}| \leq 2ne^{-s/5}$. Now $|W_i| = |W_0| + 2i$ and W_i contains at least $is/4$ edges. If $|W| \geq 3ne^{-s/5}$ then r, the number of pairs of verttices adjoined to W_0 in constructing the set W, is greater than or equal to $r_0 = \lfloor ne^{-s/5}/2 \rfloor$. But then W_{r_0} has at most $3ne^{-s/5}$ vertices and contains at least $r_0 s/4$ edges, contradicting CLAIM 2. ■

We then take $A = W \cap X$ and $B = W \cap Y$. The sub-process of constructing A involves counting the edges directed into $Y \setminus B$ from each vertex $x \in X \setminus A$, but does not depend at all on which particular vertices in $Y \setminus B$ are adjacent to x. Thus, $N(x) \cap (Y \setminus B)$ is of cardinality at least $s/4$, and it is a random set, in the sense that the probability that it is equal to a given subset of $Y \setminus B$ depends only on the cardinality of that subset; moreover, there is no dependency among the distinct sets $N(x) \cap (Y \setminus B)$ as x ranges over $X \setminus A$. Similar statements can be made about the sets $N(y) \cap (X \setminus A)$, for $y \in Y \setminus B$. There are also no dependencies between these two collections of sets. Furthermore, in constructing W using the augmenting path approach, we did not need to consider the cost of any edge with both its endpoints in $H - W$. Thus, each such edge is still an in, out, or patch edge with probability h.

5 Construction of an Optimal Assignment

Lemma 2 *The subgraph of $H_1 \cup H_2$ induced by $(X \setminus A) \cup (Y \setminus B)$ has a perfect matching (whp).*

Proof Recall that a random k-out bipartite graph on the vertex set $X \cup Y$, where X and Y are disjoint n-element sets, is constructed by having each vertex in X choose k random neighbors in Y, and each vertex in Y choose k random neighbors in X. The proof follows immediately from Walkup's result [Wal2] that a random k-out bipartite graph has a perfect matching (whp) for any $k \geq 2$. ■

Thus, we can obtain an optimal assignment (whp) by combining an optimal matching covering $A \cup B$ with a perfect matching in the subgraph of $H_1 \cup H_2$ induced by $(X \setminus A) \cup (Y \setminus B)$.

6 Structure of the Optimal Assignment

In this section we show that, if M is a random instance of the AP, then, with suitable implementation, the construction of an optimal assignment based on Lemma 2 yields a random permutation. Define the *equivalence class* of a matrix M as the set of all matrices obtained by permuting the columns of M. A typical member of this equivalence class, corresponding to the permutation π, is the matrix M^π defined by $m^\pi_{i,\pi(j)} = m_{ij}$. Except for a negligible fraction of the matrices in U_n^N (namely, those with two equal columns), the equivalence class of M consists of $n!$ distinct and equiprobable matrices. Let σ be the optimal assignment for M obtained by the algorithm described above. Then $\pi\sigma$ is an optimal assignment for M^π. Moreover, the algorithm for constructing the optimal assignment can be implemented so that

- if σ is the optimal assignment constructed for M, then $\pi\sigma$ is the optimal assignment constructed for M^π;
- A, the set of troublesome rows for M, is also the set of troublesome rows for M^π.

One way to ensure this is to permute the columns of M into lexicographic order, find the set of troublesome rows and an optimal assignment in the resulting matrix, and then permute the columns back. For any fixed σ, as π ranges over all permutations of $\{1, 2, \ldots, n\}$, $\pi\sigma$ also ranges over all permutations of $\{1, 2, \ldots, n\}$. Since all the matrices in $[M]$ are equally likely, we have established that the permutation produced by the optimal assignment algorithm described above is equally likely to be any permutation.

We note some facts about random permutations. Let σ be drawn at random from the set of permutations of $\{1, 2, \ldots, n\}$. Then

- σ has at most $2 \ln n$ cycles (whp);
- For all k,there are fewer than $w(n)^k$ cycles of length k (whp);
- There are at most $\frac{n}{w^{1/3}}$ vertices on cycles of length at most $\frac{n}{\sqrt{w}}$ (whp).

Now let σ be the optimal assignment selected by our algorithm. Then

Lemma 3 *No cycle of σ has more than one-tenth of its vertices in W (whp).*

Proof Conditioning on the event that M lies in a particular equivalence class, σ is equally likely to be any permutation, while A is a fixed set of very small cardinality (whp). A straightforward calculation shows that no cycle has more than one-twentieth of its vertices in A (whp). Indeed, given $|A| = a \leq 3ne^{-s/5}$, the expected number of cycles containing so many members of A is at most

$$\begin{aligned}\sum_{k=1}^{n} \binom{n-a}{k-\lceil k/20 \rceil} \binom{a}{\lceil k/20 \rceil} (k-1)! n^{-k} &\leq \sum_{k=1}^{n} 2^k \left(\frac{a}{n}\right)^{k/20} \\ &= o(1).\end{aligned}$$

A similar argument applies to B. ■

7 Elimination of Small Cycles

Call a cycle in a permutation *small* if it contains fewer than $\frac{n}{\sqrt{w}}$ vertices. We now show how the out-edges and in-edges are used to convert the original optimal assignment into an optimal assignment in which no cycle is small. Our procedure is to take each small cycle of the original optimal assignment σ in turn and try to remove it without creating any new small cycles. During its execution our algorithm will designate a vertex as *dirty* when its out-edges or in-edges have been observed, so that they may no longer be considered random. The initial set of dirty vertices

is the set W defined above. A vertex that is not dirty will be called *clean*. If i is clean then, independently for each j, $< i, j >$ is an out-edge with probability $\frac{s}{n}$ and $< j, i >$ is an in-edge with probability $\frac{s}{n}$. Throughout the computation, we will maintain the property that at least nine-tenths of the vertices in any remaining short cycle are clean.

We now describe the *rotation-closure algorithm* that is used to eliminate one small cycle. Let C be a small cycle of length k in the current optimal assignment. Let $\hat{k} = \min(\lceil \frac{9k}{10} \rceil, \lfloor \ln \ln n \rfloor)$. Choose $\hat{k}$ clean vertices on C. We make up to $\hat{k}$ separate attempts to remove C. The ith attempt consists of an Out Phase and an In Phase. Let v_i be the ith of the $\hat{k}$ clean vertices selected from C, and let u_i be the predecessor of v_i on C.

7.1 The Out Phase

Define a *near-cycle-cover* as a digraph θ consisting of a directed path P_θ ending at a clean vertex plus a set of vertex-disjoint directed cycles covering the vertices not in P_θ. We obtain an initial near-cycle-cover by deleting edge $< u_i, v_i >$ from the current optimal assignment, thus converting the small cycle C into a path from v_i to u_i. We then attempt to obtain many near-cycle-covers by a rotation process. The state of this process is described by a rooted tree whose nodes are near-cycle-covers, with the original near-cycle-cover at the root. Consider a typical node θ consisting of a path P_θ directed from a_θ to b_θ plus a cycle cover of the remaining vertices. We obtain descendants of θ by looking at out-edges directed from b_θ. Consider an edge that is directed from b_θ to a vertex y whose predecessor x is clean. Such an edge is *successful* if either y lies on a large cycle or y lies on P_θ and the subpaths of P_θ from a_θ to x and from y to b_θ are both of length at least $\frac{n}{\sqrt{w}}$. In such cases a descendant of θ is created by deleting $< x, y >$ and inserting $< b_\theta, y >$. Once node θ has been examined, b_θ is permanently marked dirty. The tree of near-cycle-covers is grown in a breadth-first manner until the number of leaves reaches $m = \sqrt{n \ln n}$.

We shall show later that the number of vertices marked dirty throughout the entire algorithm is $o(n)$ (whp). Assuming this, noting that each path P_θ ends in a clean vertex b_θ, and assuming that the number of vertices on short cycles is less than $\frac{n}{w^{\frac{1}{3}}}$ (this is true (whp)) the number of descendants of node θ is a random variable whose distribution is $BINOMIAL(n - o(n), s/n)$, and the random variables associated with distinct nodes are independent. Suppose that level t of the rooted tree describing the Out Phase has a vertices. Then, applying a Chernoff bound on the tails of the binomial distribution, the number of nodes at level $t+1$ lies between $\frac{as}{2}$ and $2as$, with probability greater than or equal to $1 - e^{-\frac{as}{10}}$. Hence the probability that the Out Phase fails to produce m leaves is (quite conservatively) at most

$$\sum_{k=1}^{\infty} e^{-ks/10} \leq e^{-s/20}$$

.

7.2 The In Phase

The tree produced by an Out Phase has m terminal nodes. Each of these is a near-cycle-cover in which the directed path begins at v_i. Let the jth terminal node be denoted G_j, and let the directed path in G_j run from v_i to x_j. During the In Phase we grow rooted trees independently

from all the G_j, $j = 1, 2, \ldots, m$. The process is like the Out Phase, except that, in computing the descendants of a node θ, we fan backwards along in-edges, rather than forwards along out-edges. For example, if a node θ with a path P_θ from a_θ to x_j is encountered, then we look for in-edges of the form $< x, a_\theta >$ such that x does not lie on a short cycle and y, the successor of x in G_θ, is clean. We then create a descendant by deleting $< x, y >$ and inserting $< x, a_\theta >$, provided that this substitution does not create a path or cycle of length less than $\frac{n}{\sqrt{w}}$. Once the descendants of θ have been computed, the node a_θ is permanently marked dirty.

Suppose that $S_1, S_2, \ldots, S_m$ are the near-cycle-covers produced by the Out Phase. We describe a two-stage process for producing the near-cycle-covers of the In Phase which is equivalent to that described in the previous paragraph. In the first stage, imagine that we grow the trees ensuring only that edges from clean vertices are used and that y is not on a small cycle. Thus we allow new small cycles to be produced. Each tree is grown to a depth ℓ where $(w/2)^\ell \approx m$. The following is true (whp): for each tree, and each depth $t < \ell$, the ratio between the number of nodes of depth $t+1$ and the number of nodes of depth t lies between $w/2$ and $2w$. The parameter ℓ is chosen so that even if each non-leaf has only $w/2$ descendants, the tree will still have at least m leaves. Let Σ_j denote the set of initial vertices of the paths of the near-cycle-covers created from S_j in this way. We show that $\Sigma_1 = \Sigma_2 = \ldots = \Sigma_m$. In fact let $\Sigma_j^{(t)}$ denote the set of start vertices of the paths at level t in the j'th tree. Clearly $\Sigma_j^{(0)} = \{v_i\}$ for all j and so assume inductively that we have $\Sigma_j^{(t)} = \Sigma_1^{(t)}$ for some $t \geq 0$. But then the clean vertices $\Sigma_j^{(t+1)}$ whose in-edges are directed into $\Sigma_j^{(t)}$ are the same as the clean vertices $\Sigma_1^{(t+1)}$ whose in-edges are directed into $\Sigma_1^{(t)}$. This completes the inductive step. We see also that from this construction the number of vertices marked dirty by this stage is at most $(2w)^\ell$ and then it is easy to see that the total number of dirty vertices produced is $O(n^{.5+o(1)})$. It follows from our analysis of the Out Phase that if $\mathcal{E}$ denotes the event "we fail to produce m trees in the first stage", then

$$Pr(\mathcal{E}) \leq e^{-w/20}. \tag{1}$$

We do not have to multiply the right-hand-side of (1) by m because the construction above succeeds for all S_j if and only if it succeeds for S_1. In the second stage we prune the m trees we have produced by deleting any edge which involved the construction of a small cycle. For $j = 1, 2, \ldots, m$, let T_j be the pruned tree grown from root G_j during the two stages. Thus T_j is what we would get from G_j if we had followed the procedure described in the second paragraph of this section.

Let us call T_j *good* if it has m leaves, and *bad* otherwise. Since the number of vertices marked dirty during the entire computation is $o(n)$, by the same analysis that was applied to the Out Phase, the probability that an individual T_j is bad is less than or equal to $e^{-\frac{w}{20}}$. Thus the expected number of bad trees is at most $me^{-\frac{w}{20}}$ and, by Markov's inequality, the probability that the number of bad trees is at least $m/2$ is bounded above by $2e^{-\frac{w}{20}}$. Thus,

$$\begin{aligned} Pr\left(\exists \geq \frac{m}{2} \text{ good trees}\right) &\geq Pr\left(\mathcal{E}\right) - Pr\left(\exists \geq \frac{m}{2} \text{ bad trees}\right) \\ &\geq 1 - 3e^{-w/20}. \end{aligned}$$

Assuming that The Out Phase succeeds in creating a rooted tree with m leaves, and that at least half of the m trees T_j created during the In Phase are good, we now have at least $m/2$ sets,

each consisting of m near-cycle-covers. In the set associated with T_j, each near-cycle-cover consists of a path ending at x_j, together with a cycle cover of the remaining vertices. The m paths have distinct starting points. Since the out-edges from x_j are unconditioned, the probability that none of the $\frac{m^2}{2}$ paths is closed by an out-edge is bounded above by $(1-\frac{s}{n})^{\frac{m^2}{2}} \leq n^{-\frac{s}{2}}$. Thus, with probability at least $1-n^{-\frac{s}{2}}$, one of these paths can be closed with an out-edge, and doing so creates an optimal assignment with one short cycle less than the optimal assignment that existed at the beginning of the Out Phase.

Thus the probability that the tth attempt at removing C fails given that the first $t-1$ attempts have also failed can be bounded by, say, e^{-Aw} for some absolute constant $A > 0$. Since we make $\hat{k}$ attempts, the probability that we fail to remove all short cycles is at most

$$\sum_{k=1}^{\lfloor \ln \ln n \rfloor} w^k e^{-\hat{k}Aw} + 2 \ln n e^{-Aw \ln \ln n} = o(1).$$

8 The Patching Process

At the start of the patching process we have an optimal assignment without small cycles, and the patch edges are unobserved, and thus unconditioned, except for those incident with the set of vertices $W = \{i | x_i \in A\} \cup \{j | y_j \in B\}$. Suppose we start with cycles $C_1, C_2, \ldots, C_r$, each of which contains at least $\frac{n}{\sqrt{w}}$ vertices. We describe a procedure that attempts to patch these cycles together to form a tour. The basic operation of patching together two cycles C and C' is as follows. Suppose cycle C contains an edge $< a_1, a_2 >$ and cycle C' contains an edge $< b_1, b_2 >$. If $< a_1, b_2 >$ and $< b_1, a_2 >$ are both patch edges then we can combine C and C' into a single cycle by deleting $< a_1, a_2 >$ and $b_1, b_2 >$ and inserting $< a_1, b_2 >$ and $< b_1, a_2 >$.

We attempt to create a tour by repeatedly patching cycles together in this way. We describe a generic step in which, having patched $C_1, C_2, \ldots, C_{s-1}$ together to form a cycle C, we try to patch C_s and C together. There are at most $3ne^{-\frac{s}{5}}$ vertices in W on $C_s \cup C$. Independently, for each pair consisting of an unconditioned vertex on C_s and an unconditioned vertex on C, a patch edge is present with probability s/n. Thus, the probability that there is no pair of edges that will patch C_s and C together is bounded above by $(1-(\frac{s(n)}{n})^2)^{(|C|-3ne^{-\frac{s}{5}})(|C_s|-3ne^{-\frac{s}{5}})}$. This is less than or equal to $e^{-w+o(1)}$.Hence the probability that the patching process fails is bounded above by $\sqrt{w}e^{-w+o(1)} = o(1)$.

We have now completed the entire proof of Theorem 1.

9 The Proofs of Theorems 2 and 3

In proving Theorems 2 and 3, we shall describe the generation of our matrix M, the corresponding bipartite graph H, and the directed graph G in a slightly different manner.

We first generate a random $n \times n$ matrix N where each entry is drawn uniformly from $\{0, 1, ..., \lfloor c_n n \rfloor\}$ and these choices are independent (note that in order to combine Theorems 2 and 3 we insist only that c_n is either constant or goes to infinity with n). We then randomly choose a permutation Π of $\{1, ..., n\}$ with each permutation equally likely. We obtain M by setting

$m_{i\pi(j)} = n_{ij}$ (where n_{ij} is the entry in the ith row and j^{th} column of N). Proceeding in this fashion rather than choosing the elements of M directly makes certain assertions about the independence of events obvious. We let H' be the bipartite graph which corresponds to N in the same way that H corresponds to M. We note that if we choose some optimal matching in H' then the corresponding matching in M will have the cycle cover of a random permutation.

We now need some definitions. So, consider an arbitrary matrix N and corresponding bipartite subgraph H'. By a *forced edge* of H' we mean an edge which is in every optimal matching in H'. By an *active edge* we mean an edge which is in some optimal matching of H' and has weight at least one. The first step in proving Theorems 2 and 3 is to note that if a particular weighting has a lot of forced edges then probably it will not satisfy $AP(M) = ATSP(M)$. In particular if all the edges are forced then the probability that $ATSP(M) = AP(M)$ is $\frac{1}{n}$.

The precise result we will need is that if for some weighting the corresponding H' has s forced edges then the probability that the corresponding cycle cover has a non-hamiltonian cycle made up only of forced edges - a *forced cycle* - is the same as it would be if we took a random cycle cover and then chose s edges at random and called them forced. This follows from the manner in which we generate M. It is convenient now to give a lower bound for the probability $\pi_{t,n}$ that a random cycle cover has a cycle of length at most t, (more precise esitmates are available, see for example Bollobás [B]). We will use $\pi_{t,n} \geq 1 - \frac{1}{t+1}$. To see this use induction on

$$\pi_{t,n} = \left(2t - 1 + \sum_{i=t+1}^{n-t-1} \pi_{t,i}\right)/n,$$

which is a consequence of the fact that the size of the cycle containing 1 is uniformly distributed. The following lemma then follows easily.

Lemma 4 *For any weighting of N such that H' has s forced edges, the probability that the corresponding weighting of M has a forced cycle of size at most $n-1$ and hence $AP(M) \neq ATSP(M)$ is at least*

$$\max\left\{\left(1 - \frac{1}{t+1}\right)\left(\frac{s(s-1)\dots(s-t+1)}{n(n-1)\dots(n-t+1)}\right) \mid t = 1, 2, \dots, n-1\right\}.$$

The second step in the proof is to note that most weightings will have many active edges because many vertices of G will not be the tail of any arc of cost zero. In fact since the probability that x is such a vertex is $(1 - \frac{1}{\lfloor c_n n \rfloor})^n$, we obtain:

Lemma 5 *The expected number of active edges for a random weighting is at least*

$$(1 + o(1))ne^{-1/c_n}$$

.

The key to the proof, is the following lemma which links these two results.

Lemma 6 *The expected number of forced edges in a random weighting is at least the expected number of active edges.*

Combining Lemmas 5 and 6, we obtain that the expected number of forced edges is at least $(1+o(1))ne^{-1/c_n}$. Theorem 2 then follows immediately from Lemma 4, on taking $t=2$, i.e.

$$\begin{aligned} Pr(AP(M) \neq ATSP(M)) &\geq Pr(\text{forced cycle of length at most 2}) \\ &\geq E\left(\frac{2s(s-1)}{3n(n-1)}\right) \\ &\geq \frac{2E(s)(E(s)-1)}{3n(n-1)} \qquad \text{by Jensen's Inequality} \\ &= (1-o(1))\frac{2e^{-2/c}}{3} \end{aligned}$$

Again combining Lemmas 4, 5 and 6 we see that if c_n tends to infinity with n then Theorem 3 follows from

$$\begin{aligned} Pr(AP(M) \neq ATSP(M)) &\geq Pr(\text{forced cycle of length at most t}) \\ &\geq \left(1-\frac{1}{t+1}\right)\left(1-\frac{2}{c_n}\right)^t\left(1-O\left(\frac{t^2}{n}\right)\right) \\ &= 1-o(1) \end{aligned}$$

if $t \to \infty, t = o(c_n + \sqrt{n})$.

One can tighten Theorem 2 slightly by insisting that the solution to $AP(M)$ contains no 1-cycles. Thus let $D(M)$ denote problem $AP(M)$ with the added constraint that the permutation should contain no 1-cycles i.e.be a *derangement*. If the solution to $AP(M)$ is a derangement then it also solves $D(M)$ and the probability of this tends to e^{-1}. Since forced edges occur independently of the cycle structure we can see that

$$Pr(D(M) \neq ATSP(M)) \geq (1-o(1))\left(\frac{2}{3}-(1-e^{-1})\right)e^{-2/c}.$$

The question of whether or not Theorem 3 can be similarly strengthened remains open. The answer is almost certainly yes, but how to prove it?

It remains only to prove Lemma 6.

To prove Lemma 6, we give an injective mapping from the (weighting, active edge) pairs to the (weighting, forced edge) pairs. This implies the result. Indeed, let $m = \lfloor c_n n \rfloor + 1$, $N = n^2$ and Ω_e (resp. Ω'_e) denote the set of weightings in which e is an active (resp. forced) edge. Then

$$\begin{aligned} E(s) &= m^{-N}\sum_e |\Omega_e| \\ &\geq m^{-N}\sum_e |\Omega'_e| \qquad \text{as will be shown} \\ &= E(\text{number of active edges}). \end{aligned}$$

It only remains to show that $|\Omega_e| \geq |\Omega'_e|$ for all edges e. Now, given an active edge e in a weighting W, we obtain a new weighting W' by reducing the weight on e by 1 and leaving all other weights

the same. We note that the cost of an optimal matching with respect to W' is one less than the cost of an optimal matching with respect to W and any optimal matching with respect to W' must use e. In our mapping, we map (W, e) to (W', e). Clearly, this gives the desired injection. This completes the proof of Lemma 6 and the two theorems. We note that our injection is almost a bijection because adding one to a forced edge yields an active edge in a new matching unless the forced edge has weight $\lfloor c_n n \rfloor$, surely a rare occurrence.

References

[BaTo] E.Balas and P.Toth, *Branch and bound methods*, in The traveling salesman problem: a guided tour of combinatorial optimization, E.L.Lawler, J.K.Lenstra, A.H.G.Rinnooy Kan and D.B.Shmoys Eds. (1985).

[B] B.Bollobás, Random Graphs, Academic press, London 1985.

[DF] M.E.Dyer and A.M.Frieze, *On patching algorithms for random asymmetric travelling salesman problems*, Mathematical Programming 46 (1990) 361-378.

[Fr] A.M.Frieze, *An algorithm for finding hamilton cycles in random digraphs* , Journal of Algorithms 9 (1988) 181-204.

[K] R.M.Karp, *A patching algorithm for the non-symmetric traveling salesman problem*, SIAM Journal on Computing 8 (1979) 561-573.

[KS] R.M.karp and J.M.Steele, *Probabilistic analysis of heuristics* in The traveling salesman problem: a guided tour of combinatorial optimization, E.L.Lawler, J.K.Lenstra, A.H.G.Rinnooy Kan and D.B.Shmoys Eds. (1985).

[MiPe] D.L.Miller and J.F.Pekny, *Exact solution of large asymmetric traveling salesman problems*, Science 251 (1991) 754-762.

[Wal1] D.W.Walkup, *On the expected value of a random assignment problem*, SIAM Journal of Computing 8 (1979) 440-445.

[Wal2] D.W.Walkup, *Matchings in random regular bipartite graphs*, Discrete Mathematics 31 (1980) 59-64.