An Approximation Algorithm for the Maximum Traveling Salesman Problem

Refael Hassin and Shlomi Rubinstein[1]

Abstract

We develop a polynomial time approximation algorithm for the maximum traveling salesman problem. It guarantees a solution value of at least r times the optimal one for any given $r < \frac{5}{7}$.

Keywords: Analysis of algorithms, maximum traveling salesman.

1 Introduction

Let $G = (V, E)$ be a complete (undirected) graph with node set V and edge set E. For $e \in E$ let $w(e) \geq 0$ be its weight. For $E' \subseteq E$ we denote $w(E') = \sum_{e \in E'} w(e)$. For a random subset $E' \subseteq E$, $w(E')$ denotes the expected value. The *maximum traveling salesman problem* is to compute a Hamiltonian circuit (a *tour*) with maximum total edge weight. We denote the weight of an optimal tour by *opt*. The problem is Max-SNP-hard [3] and therefore cannot have a polynomial time approximation scheme unless P=NP. Several polynomial algorithms with a constant performance guarantee are known for it [7, 8, 9, 10], a polynomial approximation scheme is known for a geometric version [2], while polynomially solvable cases are described in [3, 5, 6].

Fisher, Nemhauser and Wolsey [7] showed that the *greedy* (see also [9]), the *best neighbor*, and the *2-interchange* algorithms produce tours whose weights are at least 0.5*opt*. The *2-matching* algorithm of Fisher, Nemhauser and Wolsey [7] has a performance guarantee of $\frac{2}{3}$. Kosaraju, Park and Stein [10] improved this algorithm and claimed a bound of $\frac{5}{7}$, however there is a flaw in their proof. The correct bound is $\frac{19}{27}$ [4]. [For the directed version of the problem the algorithm in [10] still gives a bound of approximately 0.6. Our algorithms can be modified for the directed case, but the resulting bound is lower than 0.6.]

This paper contains a polynomial algorithm that computes a tour of weight at least r *opt* for any given $r < \frac{5}{7}$.

2 The algorithm

Algorithm *Max_TSP* is given in Figure 1. It constructs two tours and selects the one with greater weight. A *2-matching* (also called a *cycle cover*) is a subgraph with all vertex degrees equal to 2. As in [7], we start by computing a maximum weight cycle cover, $\mathcal{C}$. Since the maximum cycle cover problem is a relaxation of the maximum traveling salesman problem, $w(\mathcal{C}) \geq opt$. $\mathcal{C}$ consists of vertex disjoint cycles $C_1, ..., C_r$ satisfying $|C_i| \geq 3$ $i = 1, ..., r$.

The first tour is constructed by Algorithm $A1$ (see Figure 2). It uses a parameter $\epsilon > 0$ whose role is to balance between the performance guarantee and the time complexity. It treats differently *short cycles*, such that $|C_i| \leq \epsilon^{-1}$, and *long cycles*. For each short cycle it computes a maximum Hamiltonian path on its vertices. For each long cycle it deletes an edge of minimum length. The resulting path cover is extended to a tour T_1.

The second tour is also constructed from $\mathcal{C}$. We start by deleting edges from $\mathcal{C}$ according to *Delete* described in Figure 3. The result is a collection $\mathcal{P}$ of subpaths of $\mathcal{C}$ such that the following two lemmas hold:

[1]Department of Statistics and Operations Research, School of Mathematical Sciences, Tel-Aviv University, Tel-Aviv 69978, Israel. {hassin,shlom}@math.tau.ac.il

```
Max_TSP
   input
   1. A complete undirected graph G = (V, E) with weights w_ij i, j ∈ V.
   2. A constant ε > 0.
   returns A tour T.
   begin
   Compute a maximum cycle cover C = {C_1, ..., C_r}.
   T_1 := A1(G, C, ε).
   T_2 := A2(G, C).
   if w(T_1) ≥ w(T_2)
      then
           return T_1.
      else
           return T_2.
      end if
   end Max_TSP
```

Figure 1: Algorithm Max_TSP

Lemma 1 *For every edge in $\mathcal{C}$, the probability that it will be deleted by Procedure Delete is $\frac{1}{3}$.*

Proof: Consider a cycle $C_i \in \mathcal{C}$ with $l = 3k + p$ edges for $p \in \{0, 1, 2\}$. Then k edges are deleted with probability 1 and an additional edge is deleted with probability $\frac{p}{3}$. Each edge has the same probability to be deleted so that the probability for each edge is $\frac{1}{3}$. ∎

Lemma 2 *For each vertex in V, the probability that its degree in $\mathcal{P}$ is 1 is $\frac{2}{3}$. For a pair of vertices on distinct cycles of $\mathcal{C}$, the probability that both have degree 1 in $\mathcal{P}$ is $\frac{4}{9}$.*

Proof: Consider a cycle $C_i \in \mathcal{C}$ and denote $|C_i| = l$. Since each vertex of C_i has equal probability to have degree 1 in $\mathcal{P}$, it is sufficient to show that the expected number of such vertices is $\frac{2}{3}l$.

If $l \bmod 3 = 0$ then C_i breaks into 2-edge paths so that exactly $\frac{2}{3}l$ vertices have degree 1.

In the case $l \bmod 3 = 1$ there are two possibilities:

(i) If e_1 is deleted (with probability $\frac{1}{3}$) then we are left with $\frac{l-4}{3}$ 2-edge paths and two 1-edge paths with total of $\frac{2}{3}(l-4)+4$ vertices of degree 1.

(ii) If e_1 is not deleted (with probability $\frac{2}{3}$) then we are left with $\frac{l-4}{3}$ 2-edge paths and one 3-edge path so that the number of vertices with degree 1 is $\frac{2}{3}(l-4)+2$.

The expected number of degree 1 vertices is therefore $\frac{1}{3}(\frac{2}{3}(l-4)+4) + \frac{2}{3}(\frac{2}{3}(l-4)+2) = \frac{2}{3}l$. The case $l \bmod 3 = 2$ is proved similarly.

For vertices on distinct cycles, the events that a vertex has degree 1 in $\mathcal{P}$ are independent and therefore the second part of the lemma follows from the first one. ∎

The construction of the second tour is done by Algorithm $A2$ presented in Figure 4.

Lemma 3 *For every edge $e \in M$, the probability that it will be deleted by the deletion step of Algorithm $A2$ is at most $\frac{1}{4}$.*

Proof: Consider $e = (u, v) \in M$ where $u \in C_i$. Assume that *Delete* has been applied to all the cycles C_j $i \neq j$ resulting in a set $\mathcal{P}'$ of paths such that v is an end of a path (as implied by the assumption $e \in M$). Starting from u traverse e to v, follow the path whose end vertex is v, continue from its other end along the edge of M' incident with it, and continue alternating between paths of $\mathcal{P}'$ and edges of M' incident with their ends. This process may end in two ways. One is that it visits nodes of cycles other than C_i and finally it encounters an edge of M' which is not in M, that is, its other end is internal to a path in $\mathcal{P}'$. In this case e doesn't belong to a cycle of $M \cup \mathcal{P}$ no matter how *Delete* will break C_i into paths.

A1

input

1. A complete undirected graph $G=(V,E)$ *with weights* w_{ij} $i,j \in V$*.*

2. A cycle cover $\mathcal{C}$*.*

3. A constant $\epsilon > 0$*.*

returns *A tour* T_1*.*

begin

for $i = 1, ..., r$*:*

  if $|C_i| \le \epsilon^{-1}$

    then

      Compute a maximum Hamiltonian path H_i *in the subgraph induced by the vertices of* C_i*.*

    else

      Let e_i *be a minimum weight edge of* C_i*.*

      $H_i := C_i \setminus \{e_i\}$*.*

    end if

  end for

Connect $H_1, ..., H_r$ *in some arbitrary order to form a tour* T_1*.*

return T_1*.*

end *A1*

Figure 2: Algorithm $A1$

The other possibility is that we reach a vertex $u' \in C_i$ through an edge of M'. Call a cycle C_i^* of $M \cup \mathcal{P}$ a *k-cycle* if it contains k edges of M. We are interested in the cases where e belongs to a 2- or 3-cycle of $M \cup \mathcal{P}$. This is possible if u' is reached after using 2 or 3 edges from M' and the number of edges on C_i separating u and u' equals to the number of edges in a path created by *Delete*. Let p_0 be the probability (for a fixed $\mathcal{P}'$) that the pattern chosen for C_i will be such that u and u' are the two ends of a path. In all other cases either e is not on any cycle in $M \cup \mathcal{P}$ or it is on a k-cycle for $k \ge 4$. Let p_∞ be the probability that u is an end of a path while u' isn't. In this case $e \in M$ but it is not on any cycle in $M \cup \mathcal{P}$. We will prove that $p_0 \le p_\infty$ and this implies that the probability that e is deleted by Algorithm $A2$ is at most $\frac{1}{2}p_0 + \frac{1}{4}p_4 + 0p_\infty \le \frac{1}{4}$, where p_4 is the probability that e is contained in a k-cycle of $M \cup \mathcal{P}$ such that $k \ge 4$. Note that p_0, p_4 and p_∞ above are conditioned on $e \in M$. We simplify the presentation and prove $p_0 \le p_\infty$ for the uncoditional probabilites. The same relation will be implied with respect to the conditional probabilities since the change only involves division by a constant.

We prove the above property for all cases except for when $|C_i| = 4$. We will then analyze the remaining case in which e is incident with two cycles of $\mathcal{C}$ with exactly four edges each.

• $|C_i| = 3k$ $k \in \{1, 2, ...\}$. In this case *Delete* breaks C_i into 2-paths and p_0 is the probability that one of them has ends u and u'. This is possible only if exactly two edges separate u and u' on C_i and in this case with equal probability u will be an end of a path while u' will be a center of another path. Thus $p_0 = p_\infty$.

• $|C_i| = 3k+1$ $k \in \{2, 3, ...\}$. In this case *Delete* may result in two cases:

1. $k-1$ 2-paths and one 3-path.
2. $k-1$ 2-paths and two 1-paths.

Suppose first that u and u' are adjacent in C_i. p_∞ is the probability that u' will be internal in a 2- or 3-path whose one end is u. In case 1 (that occurs with probability $\frac{2}{3}$), $p_0 = 0$. [This isn't true when $k = 1$ and this is why the latter case is treated separately.] In case 2 (that occurs with probability $\frac{1}{3}$), p_0 is the probability that one of the two 1-paths will be the edge (u, u'). It follows that

$$p_0 = \frac{1}{3}\frac{2}{3k+1} \le \frac{1}{3}\frac{(k-1)}{3k+1} + \frac{2}{3}\frac{k}{3k+1} = p_\infty.$$

Suppose now that u and u' are separated by two edges, $(u,z), (z,u')$, in C_i. p_0 is the

```
Delete
  input A set of cycles C = {C_1, ..., C_r}.
  returns A set of paths P.
  begin
  for i = 1, ..., r:
      Randomly select an edge from C_i and mark it e_1.
      Denote the edges of C_i in cyclic order according to an arbitrary
      orientation and starting at e_1 by e_1, ..., e_l, where l = |C_i|.
      if l mod 3 = 0
        then
              delete the edges e_j such that j mod 3 = 0.
        elseif l mod 3 = 1
        then
              delete the edges e_j such that j mod 3 = 0
              and also delete e_1 with probability 1/3.
        elseif l mod 3 = 2
        then
              delete the edges e_j such that j mod 3 = 0
              and also delete e_1 with probability 2/3.
        end if
      end for
  Denote the resulting path set by P.
  return P.
  end Delete
```

Figure 3: Procedure *Delete*

```
A2
  input
  1. A complete undirected graph G = (V, E) with weights w_ij i, j ∈ V.
  2. A cycle cover C.
  returns A tour T_2.
  begin
  Let E' be the edges of G with two ends in different cycles of C.
  Compute a maximum weight matching M' ⊂ E'.
  P := Delete(C).
  M := {(i, j) ∈ M' : i and j have degree 1 in P}.
  % M ∪ P consists of paths P*_1, ..., P*_s and cycles C*_1, ..., C*_t such that
  each cycle contains at least two edges from M.%
  P* := {P*_1, ..., P*_s}.
  begin deletion step:
    for i = 1, ..., t:
          Randomly select an edge e ∈ C*_i ∩ M.
          P* := P* ∪ (C*_i \ e).
          end for
   end deletion step
  Complete P* to a tour T_2 by arbitrary addition of edges.
  return T_2.
  end A2
```

Figure 4: Algorithm $A2$

probability that *Delete* forms a 2-path consisting of $(u,z),(z,u')$, while p_∞ is the probability that it forms a 2- **or** 3-path containing (z,u') with z as an end vertex. Clearly, $p_0 \le p_\infty$.

Finally, if u and u' are separated by three edges, $(u,z),(z,z'),(z',u')$, in C_i then p_0 is the probability that case 1 obtains and the 3-path consists exactly of the edges between u and u'. p_∞ is at least the probability that this 3-path has z at its end and u' as an internal vertex. Thus, $p_0 \le p_\infty$.

• $|C_i| = 3k+2$ $k \in \{1,2,...\}$. In this case *Delete* may result in two cases:
1. k 2-paths and one 1-path.
2. $k-1$ 2-paths and one 4-path.

Suppose first that u and u' are adjacent in C_i. In case 2 $p_0 = 0$, unless $k=1$ in which case $p_0 = p_\infty = \frac{1}{2}$. In case 1, $p_0 = \frac{1}{3k+2} \le \frac{k}{3k+2} = p_\infty$.

Suppose now that u and u' are separated by two edges, $(u,z),(z,u')$, in C_i. p_0 is the probability that *Delete* forms a path consisting of $(u,z),(z,u')$, while p_∞ is the probability that it forms a 2- **or** 3-path containing (z,u') such that z is an end vertex. Thus $p_0 \le p_\infty$.

Finally, if u and u' are separated by four edges in C_i then in case 1 $p_0 = 0$ (unless $k=1$ in which case $p_0 = p_\infty = \frac{1}{5}$) while in case 2 p_0 is the probability that the 4-path will consist exactly of the 4 edges between u and u', so that $p_0 = \frac{1}{3k+2} \le p_\infty$.

• $e=(u,v)$ connects cycles C_i and C_j such that $|C_i| = |C_j| = 4$. Here we assume that the deletion pattern is fixed for all cycles of $\mathcal{C}$ except for C_i and C_j. *Delete* creates from C_i with probability $\frac{1}{3}$ two 1-paths and with probability $\frac{2}{3}$ one 3-path. The probability that u will be an end of a path is 1 in the first case and $\frac{1}{2}$ in the second case, and $\frac{2}{3}$ altogether.

Given that u is an end of a path, there are two possible 3-path outcomes of *Delete* and the two possible pairs of 1-paths. The probability for each of these four possibilities (conditioned on the event that u is an end of a path) is $\frac{1}{4}$. For example, for one of the possible 3-paths, the probability that it will be selected is $\frac{2}{3} \cdot \frac{1}{4}$ and the conditional probability of this event is obtained by dividing by the probability that u is an end vertex which is $\frac{2}{3}$. The same holds independently for C_j and v.

Considering the two cycles C_i and C_j, there are 11 outcomes (out of the 16 possibilities) under which both u and v are end vertices of paths, and thus satisfying the assumption $e \in M$. Out of these, 7 give that e is not on any cycle in $M \cup \mathcal{P}$. Since all possibilities have equal probabilities (by independence of the applications of *Delete* to C_i and C_j), $p_\infty \ge p_0$. ■

Theorem 4 $\max\{w(T_1), w(T_2)\} \ge \frac{5(1-\epsilon)}{7-6\epsilon} opt$.

Proof: Let T be an optimal tour. Define T_{int} (T_{ext}) to be the edges of T whose end vertices are in the same (in different) connectivity components of $\mathcal{C}$. Suppose $w(T_{int}) = \alpha w(T) = \alpha opt$. Consider the tour T_1. For each short cycle of $\mathcal{C}$ Algorithm $A1$ computed a maximum weight Hamiltonian path and therefore its contribution to the weight of T is at least the weight of T_{int} in the graph induced by its vertices. Since $\mathcal{C}$ is a maximum cycle cover, $w(C_i)$ is at least the weight of T_{int} in the subgraph induced by the vertices of C_i. In each long cycle we deleted a minimum weight edge, thus subtracting from its weight at most a factor of ϵ. Therefore, $w(T_1) \ge (1-\epsilon) w(T_{int}) \ge (1-\epsilon)\alpha opt$.

Now consider T_2. We constructed T_2 by first computing a maximum matching M' over G'. $w(M') \ge \frac{1}{2} w(T_{ext})$ since T_{ext} can be covered by two disjoint matchings in G'. We then obtained M by deleting all of the edges of M' except those whose two ends have degree 1 in $\mathcal{P}$. By Lemma 2, each edge in G' has with probability $4/9$ two ends that have degree 1 in $\mathcal{P}$. Therefore, $w(M) \ge \frac{4}{9} w(M') \ge \frac{2}{9} w(T_{ext}) = \frac{2}{9}(1-\alpha) opt$. At this stage we considered the edges of M on cycles of $M \cup \mathcal{P}$. By Lemma 3, Algorithm $A2$ deletes each $e \in M$ with probability at most $\frac{1}{4}$. The expected weight of the remaining edges is at least $\frac{3}{4} w(M) \ge \frac{1}{6}(1-\alpha) opt$. Finally, we obtained T_2 by connecting the remaining edges to $\mathcal{P}$. This step may only increase the weight of the solution. By Lemma 1, $w(\mathcal{P}) \ge \frac{2}{3} w(\mathcal{C}) \ge \frac{2}{3} opt$. Thus $w(T_2) \ge (\frac{2}{3} + \frac{1}{6}(1-\alpha)) opt$.

We conclude that $max\{w(T_1), w(T_2)\} \ge \max\{(1-\epsilon)\alpha, \frac{2}{3} + \frac{1}{6}(1-\alpha)\} opt$. The minimum value of the right hand side obtains when $\alpha = \frac{5}{7-6\epsilon}$ and it then equals $\frac{5(1-\epsilon)}{7-6\epsilon} opt$. ■

The two time consuming parts of the algorithm are the computation of a maximum 2-matching and the computation of maximum Hamiltonian paths on the subgraphs induced by the short cycles. The first can be done in $O(n^3)$ time and the latter can be done by applying dynamic programming in time $O(l^2 2^l)$ per subgraph induced by l vertices. Since for short cycles $l \leq \epsilon^{-1}$ this amounts to $O(n^2 2^{1/\epsilon})$. Thus the overall complexity is $O(n^2(n + 2^{1/\epsilon}))$. Given any factor $r < \frac{5}{7}$ we can fix $\epsilon > 0$ so that $r = \frac{5-5\epsilon}{7-6\epsilon}$ and obtain a solution of value at least r *opt* in $O(n^3)$ time.

3 Concluding remarks

Algorithm Max_TSP can be derandomized to give a deterministic polynomial algorithm with the same performance guarantee. To execute Algorithm Max_TSP we generate a random variable, X_i, for every $C_i \in \mathcal{C}$ in order to determine its deletion pattern. We will show that the analysis of the algorithm does not require full independence of these random variables but rather 3-wise independence. This enables its derandomization by replacing the underlying exponentially large sample space by one of polynomial size (see, for example, [1]).

Lemma 1 and the first part of Lemma 2 do not assume any independence relation among the random variables while the second part of Lemma 2 only requires pairwise independence.

The proof of Lemma 3 is concerned with the probability, p_0, that the pattern selected by X_i for C_i contains a subpath with ends u and u', given that the deletion patterns selected by two or three of the other cycles of $\mathcal{C}$ generate (together with M) a path between these nodes. Thus, the lemma only requires X_i to be independent of any two other variables and 3-wise independence of X_j $j = 1, ..., r$ is sufficient to prove that $p_0 \leq p_\infty$.

The algorithm also uses randomization in the deletion step of $A2$. To complete the derandomization we replace the deletion step by a deterministic one. Instead of deleting a random edge in each set $C_i^* \cap M$ we delete a smallest weight edge in this set. The weight of the resulting set of paths is at least that of $\mathcal{P}^*$ and therefore Theorem 4 still holds.

When applying the technique of [1] we compute a prime number $q(n) \geq n$ and generate r 3-wise independent uniform random variables $V_1, ..., V_r$ with range $\{0, ..., q(n) - 1\}$. To generate X_i we map each V_i to the three possible deletion patterns for C_i if $|C_i|$ *mod 3* $= 0$. Otherwise we map it to the $2|C_i|$ patterns that are possible for C_i. We would like to maintain that X_i has the desired probabilities for each pattern. These are $\frac{1}{3}$ in the first case and $\frac{1}{3|C_i|}$ or $\frac{2}{3|C_i|}$ otherwise. We select $q(n)$ such that $\frac{q(n)}{n}$ slowly increases to ∞, and then the desired probabilities can be approached to any desired accuracy. Thus, we obtain that the lemmas and theorem asymptotically hold and a solution with value at least r *opt* can be obtained for any $r < \frac{5}{7}$. The size of the sample space is $q(n)^3$.

Finally, we note that there exists an attractive version of our algorithm whose analysis seems to be more difficult but its actual bound may be better. Apply *Delete* before computing the matching M'. Then define E" as the set of edges connecting pairs of nodes that are ends of paths generated from distinct cycles. Compute a maximum matching on E", call it M", and continue as in $A2$ with M" replacing M. The advantage of this version is that $w(M") \geq w(M)$. However, in general we now have $p_\infty = 0$ and Lemma 3 doesn't hold.

Acknowledgment

We thank Randeep Bhatia for suggesting that we derandomize Max_TSP using the fact that it only assumes limited independence.

References

[1] N. Alon, L. Babai and A. Itai, "A fast and simple randomized parallel algorithm for the maximal independent set problem", *Journal of Algorithms* **7** (1986) 567-583.

[2] A. I. Barvinok, "Two algorithmic results for the traveling salesman problem," *Mathematics of Operations Research* **21** (1996),65-84.

[3] A. I. Barvinok, D. S. Johnson, G. J. Woeginger, and R. Woodroofe, "Finding maximum length tours under polyhedral norms," (1998).

[4] R. Bhatia, private communication.

[5] D. Blokh and G. Gutin, "Maximizing traveling salesman problem for special matrices," *Discrete Applied Mathematics* **56** (1995) 83-86.

[6] V. G. Deĭneko and G. J. Woeginger, "The maximum traveling salesman problem on Demidenko matrices," Report Woe-09, TU Graz, September 1997.

[7] M.L. Fisher, G.L. Nemhauser and L.A. Wolsey, "An analysis of approximations for finding a maximum weight Hamiltonian circuit," *Operations Research* **27** (1979) 799-809.

[8] R. Hassin and S. Rubinstein, "An approximation algorithm for maximum packing of 3-edge paths," *Information Processing Letters*, **63** (1997) 63-67.

[9] B. Korte and D. Hausmann, "An analysis of the greedy heuristic for independence systems," *Annals of Discrete Mathematics* **2**, 1978, 65-74.

[10] S.R. Kosaraju, J.K. Park and C. Stein, "Long tours and short superstrings," *35th Annual IEEE Symposium on Foundations of Computer Science (FOCS)* (1994) 166-177.