Polynomial Algorithms for Shortest Hamiltonian Path and Circuit

Dhananjay P. Mehendale
Sir Parashurambhau College, Tilak Road, Pune 411030,
India

Abstract

The problem of finding shortest Hamiltonian path and shortest Hamiltonian circuit in a weighted complete graph belongs to the class of NP-Complete problems [1]. This well known problem asks for a method or algorithm to locate such path or circuit that passes through every vertex only once in the given weighted complete graph. In this paper we begin with proposing two approximation algorithms for shortest Hamiltonian graphs which essentially consists of applying certain chosen permutations (transpositions or product of transpositions) on the adjacency matrix of given weighted complete graph causing reshuffling of the labels of its vertices. We change the labels of vertices through proper choice of permutations in such a way that in this relabeled graph the Hamiltonian path $1\rightarrow 2\rightarrow 3\rightarrow \ldots .k\rightarrow (k+1)\rightarrow \ldots \rightarrow p$ becomes approximation to shortest path in the given weighted complete graph under consideration. We then define so called ordered weighted adjacency list for given weighted complete graph and proceed to the main result of the paper, namely, the exact algorithm based on utilization of ordered weighted adjacency list and the simple properties that any path or circuit must satisfy. This algorithm performs checking of sub-lists, containing (n-1) entries (edge pairs) for paths and n entries (edge pairs) for circuits, chosen from ordered adjacency list in a well defined sequence to determine exactly the shortest Hamiltonian path and shortest Hamiltonian circuit. The procedure has intrinsic advantage of landing on the desired solution in quickest possible time and even in worst case in polynomial time.

Introduction: Let G be a weighted complete graph with the vertex set $V(G)$ and edge set $E(G)$ respectively:

$V(G) = \{v_1, v_2, \cdots, v_p\}$ and

$E(G) = \{e_1, e_2, \cdots, e_q\}$

Let $A_G = [w_{ij}]_{p \times p}$ denotes the weighted adjacency matrix of *G*.

Note: Applying transposition (m, n) on A_G is essentially equivalent to interchanging rows as well as columns, m and n. That is replace m-th row in A_G by n-th row and vice versa and then in thus transformed matrix replace m-th column by n-th column and vice versa (order of these operations, i.e. whether you interchange rows first and then interchange columns or you interchange columns first and then interchange rows, is immaterial as it produce same end result). Note that this transformation essentially produces a new weighted adjacency matrix that will result due to interchanging labels of vertices v_m, v_n in the original weighted complete graph. We now discuss following algorithm which essentially is an approximation algorithm that produce the result comparable to one obtains from known approximation algorithms.

Algorithm 1 (An Approximation Algorithm):

(1) If entry at position (1, 2) in the matrix, i.e. weight w_{12} is already smallest in the first row then proceed to step 2. Else, among the weights $w_{1j}, j = 2,3,..., p$, find minimum weight, say w_{1j_1}. Apply transposition $(2, j_1)$ on A_G, producing new weighted adjacency matrix, say A_{G_1}.

(2) If entry at position (2, 3) in the matrix, i.e. weight w_{23} is already smallest in the second row then proceed to step 3. Else, among the weights $w_{2j}, j = 3,4,..., p$, find minimum weight, say w_{2j_2}. Now apply transposition $(3, j_2)$ on A_{G_1}, producing new weighted adjacency matrix, say A_{G_2}.

(3) If entry at position (3, 4) in the matrix, i.e. weight w_{34} is already smallest in the third row then proceed to step 4. Else, among the weights $w_{3j}, j = 4,5,..., p$, find minimum weight, say w_{3j_3}. Now apply

transposition $(4, j_3)$ on A_{G_2}, producing new weighted adjacency matrix, say A_{G_3}.

(4) Continue this procedure applying appropriate transpositions till we finally reach (p-2)-th row and among the weights $w_{(p-2)j}, j=(p-1), p$, find minimum weight, say $w_{(p-2)j_{(p-2)}}$. Now apply transposition $((p-1), j_{(p-2)})$ on $A_{G_{(p-3)}}$, producing new weighted adjacency matrix, say $A_{G_{(p-2)}}$.

(5) Find the sum of weights of edges in the Hamiltonian path

$$1 \to 2 \to 3 \to \cdots \to j \to (j+1) \to \cdots \to (p-1) \to p$$

□

Remark: After carrying out "algorithm 1" on given weighted complete graph the Hamiltonian path

$$1 \to 2 \to 3 \to \cdots \to j \to (j+1) \to \cdots \to (p-1) \to p$$

produces a good **approximation** for shortest Hamiltonian path in the given (and conveniently relabeled due to applied transpositions) weighted complete graph. This Hamiltonian path thus obtained will not necessarily be a shortest one.

What we have necessarily achieved is as follows: By application of permutation (transposition) we bring smallest weight entry in the first row at position (1, 2) in the weighted adjacency matrix. This is achieved by transposition of type $(2, j_1)$, where $j_1 > 2$. The algorithm then applies transposition which brings smallest weight entry in the second row at position (2, 3), in the transformed weighted adjacency matrix that results after applying transposition mentioned above. This is achieved by transposition of type $(3, j_2)$, where $j_2 > 3$. Note that because of its special form this second transposition doesn't affect the smallest entry achieved at position (1, 2) while bringing smallest entry (weight) in the second row at

position (2, 3) by this second transposition! This story continues, i.e. the later applied transpositions doesn't affect the results of earlier transpositions because of the special choice of the transpositions and at end achieves smallest possible weights in the rows at positions on the **diagonal neighboring the principle diagonal**, i.e. at positions (1, 2), (2, 3), …., (p-1, p), of the evolved weighted adjacency matrix, evolved through the successive transpositions of specially chosen type. Thus, we have achieved the neighboring diagonal which represents the weights on the Hamiltonian path

$$1 \to 2 \to 3 \to \cdots \to j \to (j+1) \to \cdots \to (p-1) \to p$$

to contain smallest entries from the rows of initially given weighted adjacency matrix.

Question 1: When the Hamiltonian path

$$1 \to 2 \to 3 \to \cdots \to j \to (j+1) \to \cdots \to (p-1) \to p$$

thus produced by this algorithm will be the desired shortest Hamiltonian path?

Answer: The Hamiltonian path

$$1 \to 2 \to 3 \to \cdots \to j \to (j+1) \to \cdots \to (p-1) \to p$$

will be shortest if and only if we will (somehow) manage the maximization of sum of weights of entries in the triangular shaped submatrix of the transformed adjacency matrix, i.e. when the following sum

$$\sum_{i=1}^{p-2} \sum_{j=(i+2)}^{p} w_{ij}$$

gets **maximized.**

Now, the next question that naturally arises is as follows:

Question 2: How to maximize this sum?

We will try to come to its possible answer but before that let us consider following

Example 1: We consider following weighted adjacency matrix representing a weighted complete graph and find the Hamiltonian path in its relabeled copy in the form

$$1 \to 2 \to 3 \to \cdots \to j \to (j+1) \to \cdots \to (p-1) \to p$$

by applying "algorithm 1". We will see that this Hamiltonian path is not shortest one. We consider the following weighted adjacency matrix and apply "algorithm 1" to it:

$$\begin{bmatrix} 0 & 1 & 6 & 8 & 4 \\ 1 & 0 & 8 & 5 & 6 \\ 6 & 8 & 0 & 9 & 7 \\ 8 & 5 & 9 & 0 & 8 \\ 4 & 6 & 7 & 8 & 0 \end{bmatrix}$$

(1) Since entry at position (1, 2) is already smallest in the first row we proceed to next step.

(2) Since entry in position (2, 4) = 5 is smallest in second row we apply transposition (3, 4) on the above matrix that results into matrix

$$\begin{bmatrix} 0 & 1 & 8 & 6 & 4 \\ 1 & 0 & 5 & 8 & 6 \\ 8 & 5 & 0 & 9 & 8 \\ 6 & 8 & 9 & 0 & 7 \\ 4 & 6 & 8 & 7 & 0 \end{bmatrix}$$

(3) Since entry in position (3, 5) = 8 is smallest in third row we apply transposition (4, 5) on the above matrix that results into matrix

$$\begin{bmatrix} 0 & 1 & 8 & 4 & 6 \\ 1 & 0 & 5 & 6 & 8 \\ 8 & 5 & 0 & 8 & 9 \\ 4 & 6 & 8 & 0 & 7 \\ 6 & 8 & 9 & 7 & 0 \end{bmatrix}$$

Clearly, in this transformed weighted adjacency matrix the Hamiltonian path

$$1 \to 2 \to 3 \to 4 \to 5$$

has total weight $\sum_{i=1}^{4} w_{i,(i+1)} = 21$

Now, it is easy to check that $\sum_{i=1}^{p-2} \sum_{j=(i+2)}^{p} w_{ij}$ in this case is equal to 41.

This sum is actually **not maximized** as we will see below. We actually need to apply some more permutations on the given weighted adjacency matrix that are suitable to maximize this sum. Now without displaying all necessary suitable permutations we have to carry out for maximization of this sum we give the final result below depicting the transformed form of the **same** weighted adjacency matrix with which we started applying algorithm 1. It is

$$\begin{bmatrix} 0 & 5 & 8 & 8 & 9 \\ 5 & 0 & 1 & 6 & 8 \\ 8 & 1 & 0 & 4 & 6 \\ 8 & 6 & 4 & 0 & 7 \\ 9 & 8 & 6 & 7 & 0 \end{bmatrix}$$

Clearly, in this transformed weighted adjacency matrix the Hamiltonian path

$$1 \to 2 \to 3 \to 4 \to 5$$

has total weight $\sum_{i=1}^{4} w_{i,(i+1)} = 17$

Now, it is easy to check that $\sum_{i=1}^{p-2} \sum_{j=(i+2)}^{p} w_{ij}$ in this case is equal to 45.

It can be checked by **brute force** that is this desired sum has **maximized** now and so the Hamiltonian path that we have thus obtained in the transformed weighted adjacency matrix is actually the **shortest** one!

Thus, the problem of finding shortest Hamiltonian path in the form

$$1 \to 2 \to 3 \to \cdots \to j \to (j+1) \to \cdots \to (p-1) \to p$$

has become a **constrained optimization problem** of following type:

Problem: Given weighted adjacency matrix corresponding to a given weighted complete graph then devise permutations which will transform this matrix such that path

$$1 \to 2 \to 3 \to \cdots \to j \to (j+1) \to \cdots \to (p-1) \to p$$

has shortest length. In other words, devise sequence of permutations to be applied on given weighted adjacency matrix to be applied on the given weighted complete graph such that $\sum_{i=1}^{p-2} \sum_{j=(i+2)}^{p} w_{ij}$ gets maximized and thus the transformed matrix represents the same weighted complete graph in disguise.

Now, this sum can be seen as made up of sum of entries in columns p, (p-1), (p-2), …, such that p-th column contains entries w_{1p}, w_{2p}, …. , $w_{(p-2)p}$, (p-1)-th column contains entries $w_{1(p-1)}$, $w_{2(p-1)}$, …. , $w_{(p-3)(p-1)}$, etc.

We now proceed to discuss a possible algorithm to tackle this problem.

Algorithm 2 (An Approximation Algorithm):

(1) We begin with maximizing w_{1p} . Pick largest weight edge say w_{ij} in the given weighted adjacency matrix, A_G . Transform this weight to position w_{1p} by applying product of transpositions $(1,i)(j,p)$ on A_G . Thus, we have now maximized w_{1p} .

(2) Now, among the edges emerging from vertex with label 1 and p in the transformed weighted adjacency matrix, A_G , due to step (1), find some edges $(1,i),(p,j),i \neq j$ such that $(1,i)$ is smallest among the edges emerging from 1 and (p,j) is smallest among the edges emerging from p. Apply product of transpositions $(2,i)(p-1,j)$ on new transformed A_G we get after step (1) so that weights w_{12} and $w_{(p-1)p}$ are now smallest possible for the situation.

(3) Now, among the edges emerging from vertex with label 2 and (p-1) in the transformed weighted adjacency matrix, A_G , due to step (2), find some edges $(2,i),(p-1,j),i \neq j$ such that $(2,i)$ is smallest among the edges emerging from 2 and $(p-1,j)$ is smallest among the edges emerging from (p-1). Apply product of transpositions $(3,i)(p-2,j)$ on new transformed A_G we get after step (2) so that weights w_{23} and $w_{(p-2)(p-1)}$ are now smallest possible for the situation.

(4) Continue steps like (2) and (3) of applying suitable transpositions till we reach at the state having smallest possible weights for w_{12} , w_{23} , w_{34} ,

…., $w_{(p-2)(p-1)}$, $w_{(p-1)p}$ in the given situation and all other edges have larger weights.

(5) Now, all vertices have been relabeled and we have assigned smallest possible weights to edges comprising the Hamiltonian path

$$1 \to 2 \to 3 \to \cdots \to j \to (j+1) \to \cdots \to (p-1) \to p$$

Now, apply suitable transpositions of type $(i, i+2), i = 1,2,\cdots$ by checking that they cause improvement in the sum $\sum_{i=1}^{p-1} w_{i(i+1)}$.

□

Example 2: We consider following weighted adjacency matrix to apply "algorithm 2".

$$\begin{bmatrix} 0 & 11 & 2 & 5 & 3 \\ 11 & 0 & 1 & 6 & 3 \\ 2 & 1 & 0 & 12 & 4 \\ 5 & 6 & 12 & 0 & 8 \\ 3 & 3 & 4 & 8 & 0 \end{bmatrix}$$

(1) Since w_{34} is largest so we apply product of transpositions (1, 3)(4,5) on this matrix leading to

$$\begin{bmatrix} 0 & 1 & 2 & 4 & 12 \\ 1 & 0 & 11 & 3 & 6 \\ 2 & 11 & 0 & 3 & 5 \\ 4 & 3 & 3 & 0 & 8 \\ 12 & 6 & 5 & 8 & 0 \end{bmatrix}$$

(2) w_{12} ia already minimum so to minimize w_{45} we apply transposition (3, 4) on above matrix leading to new matrix as follows:

$$\begin{bmatrix} 0 & 1 & 4 & 2 & 12 \\ 1 & 0 & 3 & 11 & 6 \\ 4 & 3 & 0 & 3 & 8 \\ 2 & 11 & 3 & 0 & 5 \\ 12 & 6 & 8 & 5 & 0 \end{bmatrix}$$

(3) As per step (5) of the algorithm to achieving further minimization, we apply transposition (1, 3) on above matrix leading to new matrix as follows:

$$\begin{bmatrix} 0 & 3 & 4 & 3 & 8 \\ 3 & 0 & 1 & 11 & 6 \\ 4 & 1 & 0 & 2 & 12 \\ 3 & 11 & 2 & 0 & 5 \\ 8 & 6 & 12 & 5 & 0 \end{bmatrix}$$

Clearly, in this transformed weighted adjacency matrix the Hamiltonian path

$$1 \to 2 \to 3 \to 4 \to 5$$

has total weight $\sum_{i=1}^{4} w_{i,(i+1)} = 11$

The Ordered Weighted Adjacency List and Sub-lists: We now proceed with the discussion of the main results of this paper. We propose a smart algorithm to find shortest Hamiltonian path and shortest Hamiltonian circuit in the given weighted complete graph.

Definition 1: The **weighted adjacency list**, $WAL(G)$, associated with the given weighted complete graph, G, on p vertices, is the following bitableau in which the left tableau represents weights of the edges represented by vertex pairs written in the same row in the right tableau.

Definition 2: The weighted adjacency list is called **ordered weighted adjacency list** and denoted as $OWAL(G)$ when the rows of weighted adjacency list are so permuted that the weights in the left tableau get ordered, i.e. these weights form a nondecreasing sequence in the downward

direction. In other words, the weighted adjacency list becomes ordered weighted adjacency list when the left tableau becomes a nondecreasing column of entries representing weights of the edges written in the corresponding row in the right tableau. The $WAL(G)$ is following bitableau:

$$WAL(G) = \begin{bmatrix} w_{i_1 j_1} & (i_1, j_1) \\ w_{i_2 j_2} & (i_2, j_2) \\ \bullet & \bullet \\ \bullet & \bullet \\ w_{i_l j_l} & (i_l, j_l) \\ \bullet & \bullet \\ w_{i_m j_m} & (i_m, j_m) \end{bmatrix}$$

where, $m = \dfrac{p(p-1)}{2}$.

The $OWAL(G)$ is the following bitableau:

$$OWAL(G) = \begin{bmatrix} w_{i_1 j_1} & (i_1, j_1) \\ w_{i_2 j_2} & (i_2, j_2) \\ \bullet & \bullet \\ \bullet & \bullet \\ w_{i_l j_l} & (i_l, j_l) \\ \bullet & \bullet \\ w_{i_m j_m} & (i_m, j_m) \end{bmatrix}$$

where $m = \frac{p(p-1)}{2}$ and in addition, $w_{i_1 j_1} \leq w_{i_2 j_2} \leq \cdots \leq w_{i_m j_m}$

Definition 3: The **weighted adjacency sub-list**, $SubWAL(G)$, is essentially any sub-bitableau made by picking some portion of the $WAL(G)$, i.e. made by picking any rows among the rows in $WAL(G)$.

Definition 4: The **ordered weighted adjacency sub-list**, $SubOWAL(G)$, is essentially any sub-bitableau made by picking some portion of the $OWAL(G)$ and keeping them in the same nondecreasing order, i.e. made by picking any rows among the rows in $OWAL(G)$ and keeping them in the same ordered form.

Definition 5: The **size** of weighted adjacency sub-list, or ordered weighted adjacency sub-list, is the cardinality of entries in the sub-list, i.e. number of rows in the sub-bitableau representing that sub-list.

Definition 6: The **weight** of the weighted adjacency sub-list, or ordered weighted adjacency sub-list, is the sum of weights in the left sub-tableau representing that sub-list.

It is **easy to check** that

(A) A set of (p-1) vertex pairs in the right tableau of $OWAL(G)$ together represents a Hamiltonian path if (i) these pairs together contain all the vertices, (ii) the degrees of all but two vertices is two, (iii) the degree of the left out two vertices is one, and (iv) these vertex pairs together form a connected graph.

(B) A set of p vertex pairs in the right tableau of $OWAL(G)$ together represents a Hamiltonian circuit if (i) these pairs together contain all the vertices, (ii) the degrees of all vertices is two, and (iii) these vertex pairs together form a connected graph.

These two easy checks will form an **important backbone** of our exact algorithm for finding shortest Hamiltonian path or shortest Hamiltonian circuit in the given weighted complete graph.

We now proceed with

Algorithm 3 Shortest Hamiltonian Path (Exact):

(1) Form ordered weighted adjacency list, $OWAL(G)$, corresponding to given weighted complete graph.

(2) Form all possible ordered weighted adjacency sub-lists, $SubOWAL(G)$, each of size (p-1).

(3) Arrange these sub-lists in lexicographic order in accordance with their respective weights.

(4) Use easy check (A) mentioned above in succession (starting with smallest weight sub-list) on each sub-list and stop at the first success.

(5) Record the Hamiltonian path thus obtained and its weight. This will be a desired **shortest Hamiltonian path!**

□

Algorithm 3 Shortest Hamiltonian Circuit (Exact):

(1) Form ordered weighted adjacency list, $OWAL(G)$, corresponding to given weighted complete graph.

(2) Form all possible ordered weighted adjacency sub-lists, $SubOWAL(G)$, each of size p.

(3) Arrange these sub-lists in lexicographic order in accordance with their respective weights.

(4) Use easy check (B) mentioned above in succession (starting with smallest weight sub-list) on each sub-list and stop at the first success.

(5) Record the Hamiltonian circuit thus obtained and its weight. This will be a desired **shortest Hamiltonian circuit!**

□

Example 3: Consider following weighted adjacency matrix:

$$\begin{bmatrix} 0 & 1 & 6 & 8 & 4 \\ 1 & 0 & 8 & 5 & 6 \\ 6 & 8 & 0 & 9 & 7 \\ 8 & 5 & 9 & 0 & 8 \\ 4 & 6 & 7 & 8 & 0 \end{bmatrix}$$

The ordered weighted adjacency list for this is as below, conveniently in the form of a 2-columned table:

1	(1,2)
4	(1,5)
5	(2,4)
6	(1,3)
6	(2,5)
7	(3,5)
8	(2,3)
8	(1,4)
8	(4,5)
9	(3,4)

It is easy to check that pairs {(1,2), (1,5), (2,4), (3,5)} together form the desired shortest Hamiltonian path as per the "algorithm 3" and pairs {(1,2), (1,5), (2,4), (3,5), (3,4)} together form the desired shortest Hamiltonian circuit as per the "algorithm 4".

References

1. Christos H. Papadimitriou, Computational Complexity, Page 190, Addison-Wesley Publishing Company, Inc.,1994.