Computing Maximum Hamiltonian Paths in Complete Graphs with Tree Metric

Wojciech Rytter[1,2,⋆] and Bartosz Szreder[1]

[1] Dept. of Mathematics, Computer Science and Mechanics,
University of Warsaw, Warsaw, Poland
{rytter,szreder}@mimuw.edu.pl
[2] Dept. of Math. and Informatics, Copernicus University, Toruń, Poland

Abstract. We design a linear time algorithm computing the maximum weight Hamiltonian path in a weighted complete graph K_T, where T is a given undirected tree. The vertices of K_T are nodes of T and $weight(i, j)$ is the distance between i, j in T. The input is the tree T and two nodes u, $v \in T$, the output is the maximum weight Hamiltonian path between these nodes. The size n of the input is the size of T (however the total size of the complete graph K_T is quadratic with respect to n). Our algorithm runs in $\mathcal{O}(n)$ time. Correctness is based on combinatorics of alternating sequences. The problem has been inspired by a similar (but much simpler) problem in a famous book of Hugo Steinhaus.

1 Introduction

The maximum Hamilton cycle and path problems are generally NP-hard, see [2,3]. We introduce an interesting class of graphs for which these problems are solvable in linear time. Although it is rather of small practical importance, it is combinatorially and algorithmically quite interesting.

In his famous book *"One Hundred Problems in Elementary Mathematics"* Hugo Steinhaus as Problem 65 asked for the value $\max(n)$ of a maximum Hamiltonian path in the graph K_n with weights of edges between a pair of vertices (i, j) given by $|i - j|$. In other words the weights of edges correspond to the metric of a simple path of nodes, a trivial case of an undirected tree. In this paper we extend this to aribitrary tree with positive weights on edges. In case of a metric given by a simple path there are very elementary closed formulas for the total weight of maximum Hamiltonian paths in graphs implied by this metric.

Lemma 1. [H. Steinhaus, see [1]]
If n is even then $\max(n) = \frac{n^2-2}{2}$, *otherwise* $\max(n) = \frac{n^2-3}{2}$.

In this paper we extend this to a more complicated problem of constructing in linear time a maximum Hamiltonian path between any given pair of nodes, with a metric implied by an arbitrary tree.

⋆ Supported by grant no. N206 566740 of the National Science Centre.

E. Kranakis, D. Krizanc, and F. Luccio (Eds.): FUN 2012, LNCS 7288, pp. 346–356, 2012.

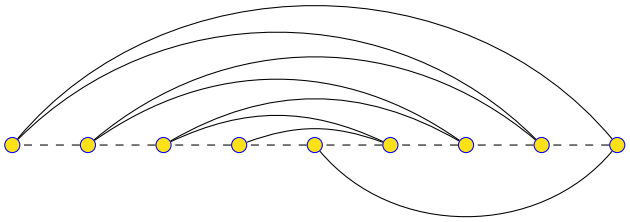


Fig. 1. A longest Hamiltonian path $\mathcal{P}$ when T is the line with $n = 9$ nodes and unit cost edges. According to Lemma 1 we have $weight(\mathcal{P}) = \frac{n^2-3}{2} = 39$. This path starts in a centroid and ends in its neighbor. Maximum paths between arbitrary pairs of nodes (usually not adjacent ones) are more complicated.

Assume T is an undirected tree with the set of nodes $\{1, 2, \ldots, n\}$. By $\mathrm{dist}_T(u, v)$ we denote the length of the shortest path between u and v in T. Let K_T be the complete graph K_n with weights of edges given by

$$weight(u, v) = \mathrm{dist}_T(u, v).$$

We define formally our problem as follows:

input: given a tree T with n nodes and two different nodes $u, v \in V(T)$;
output: the maximum weight Hamiltonian path in K_T from u to v.

Our main result is a linear time algorithm solving this problem. The main tools are *centroids* of a tree and *alternating sequences*: colored sequences in which adjacent elements are of diferent colors. In the next sections we discuss them in detail.

2 Alternating Sequences and Maximum Hamiltonian Paths

Assume we have a coloring $\mathcal{C}$ of a set of n elements. We represent a coloring as a partition of this set into color classes:

$$\mathcal{C} = (C_1, C_2, \ldots, C_k)$$

We say that a sequence γ over this set is $\mathcal{C}$-*alternating* if and only if it is a permutation of all n elements of this set and no two adjacent elements of the sequence are of the same color.

Example 1. Let

$$\mathcal{C} = (\{1, 2\}, \{3, 4, 5\}, \{6, 7, 8, 9, 10, 11\}),$$

then as an alternating sequence we can take

$$\gamma = (6\ 4\ 7\ 1\ 8\ 3\ 9\ 2\ 10\ 5\ 11).$$

We project our notion of coloring onto the set of vertices of a tree. Assume $r \in T$. We root the tree in this node. Then by r-coloring we mean a coloring such that the color of each node $v \neq r$ corresponds to the subtree of $T - \{r\}$ containing v and r has unique color.

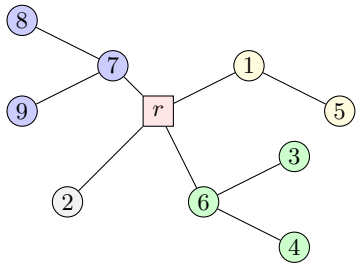


Fig. 2. A tree rooted in r gives r-coloring as a family of sets $\mathcal{C} = (\{r\}, \{2\}, \{1, 5\}, \{3, 4, 6\}, \{7, 8, 9\})$. Each set represents different color and corresponds to a distinct subtree of the tree rooted in r, the vertex r is colored by its own color.

For $u \in V(T)$ denote by $\Delta(u)$ the sum of distances from u to all other vertices in T.

Lemma 2. *Let r be any node of T and $\mathcal{C}$ be the r-coloring of all nodes of T. If γ is a $\mathcal{C}$-alternating sequence starting with u and ending with v then:*
(a) *γ is a maximum weight Hamiltonian path from u to v in K_T*
(b) *$weight(\gamma) = 2\Delta(r) - \mathrm{dist}(u, r) - \mathrm{dist}(v, r)$.*

Proof. We first prove the following fact:

Claim. For any node r' and a path $\gamma' = x_1 x_2 \ldots x_n$, where $x_1 = u$, $x_n = v$, we have

$$weight(\gamma') \leq 2\Delta(r') - \mathrm{dist}(u, r') - \mathrm{dist}(v, r')$$

Proof. Observe that the triangle inequality

$$\mathrm{dist}(x_i, x_{i+1}) \leq \mathrm{dist}(x_i, r') + \mathrm{dist}(r', x_{i+1})$$

implies the following:

$$\begin{aligned} weight(\gamma') \leq\ & \mathrm{dist}(x_1, r') + \mathrm{dist}(r', x_2) + \\ & + \mathrm{dist}(x_2, r') + \mathrm{dist}(r', x_3) + \\ & \ldots \\ & + \mathrm{dist}(x_{n-1}, r') + \mathrm{dist}(r', x_n) \\ & = 2\Delta(r') - \mathrm{dist}(u, r') - \mathrm{dist}(v, r') \end{aligned}$$

This completes the proof of the claim.

Consequently a sequence γ (if it exists) satisfying the property in the assumption of Lemma 2 is of maximum weight. This result follows from definition of r-coloring: x_i and x_{i+1} are of different colors, so they reside in different subtrees of a tree rooted in r (or one of them is r, but the argument holds anyway). Because of that we now have:

$$\forall_{1 \le i < n} \ \mathrm{dist}(x_i, x_{i+1}) = \mathrm{dist}(x_i, r) + \mathrm{dist}(r, x_{i+1})$$

This completes the proof of the lemma.

Unfortunately the lemma gives no hint about existence and efficient location of a node r satisfying the assumption. The next two sections give tools to find good r.

3 Some Useful Combinatorics of Alternating Sequences

Assume $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ such that $|C_1| \le |C_2| \le \ldots |C_k|$. Denote by $\max(\mathcal{C})$ the size of the largest set of elements of the same color and by $rem(\mathcal{C})$ the number of remaining elements, hence:

$$rem(\mathcal{C}) + \max(\mathcal{C}) = n.$$

For two sequences

$$\alpha = \ (a_1, a_2, \ldots, a_p), \ \beta = (b_1, b_2, \ldots, b_q), \ q \le p$$

we define the *interleave* operation as follows:

$$interleave(\alpha, \beta) = (a_1, b_1, a_2, b_2, a_3, b_3, \ldots, a_q, b_q, a_{q+1}, a_{q+2}, \ldots, a_p)$$

Example 2. For sequences $\alpha = (1, 2, 5, 8, 9)$ and $\beta = (6, 3, 4)$ we have $interleave$ $(\alpha, \beta) = (1, 6, 2, 3, 5, 4, 8, 9)$.

We also introduce the operation *sequentialize* which for a sequence of disjoint sets produces the sequence of all elements of these sets starting with elements of the first set and continuing with elements of consecutive sets (order inside sets is arbitrary).

Lemma 3 (on existence of an alternating sequence).

1. *A $\mathcal{C}$-alternating sequence exists if and only if* $\max(\mathcal{C}) \le rem(\mathcal{C}) + 1$.
2. *If it exists then it can be constructed in linear time.*
3. *If both end-elements of the computed sequence γ are of the same color then this color occurs $rem(\mathcal{C}) + 1$ times.*

Proof.
The necessity is obvious: if $\max(\mathcal{C}) > rem(\mathcal{C}) + 1$ then we do not have enough remaining elements to separate all elements of the largest color, so in any permutation two of them should be neighbors.

Assume now that $\max(\mathcal{C}) \leq rem(\mathcal{C}) + 1$ and consider the case of odd number n of elements. We can construct the required sequence γ in the following way:

Algorithm 1. AlterSeq1($\mathcal{C}$)

input : $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, $|C_1| \leq |C_2| \leq \ldots \leq |C_k|$
output: $\mathcal{C}$-alternating sequence of $C_1 \cup C_2 \cup \ldots \cup C_k$
1 Find j and a partition of $C_j = C'_j \cup C''_j$ such that
$|C_1 \cup C_2 \cup \ldots \cup C'_j| + 1 = |C''_j \cup C_{j+1} \cup \ldots \cup C_k|$;
2 $\alpha := sequentialize(C_k, C_{k-1}, \ldots C_{j+1}, C''_j)$;
3 $\beta := sequentialize(C'_j, C_{j-1}, \ldots C_2, C_1)$;
4 $\gamma := interleave(\alpha, \beta)$;
5 **return** γ;

A similar argument can be used in case of even n: now we split the coloring to get equality:

$$|C_1 \cup C_2 \cup \ldots \cup C'_j| = |C''_j \cup C_{j+1} \cup \ldots \cup C_k|$$

It is easy to see that the algorithm *AlterSeq1* produces $\mathcal{C}$-alternating sequence γ of $C_1 \cup C_2 \cup \ldots \cup C_k$ in linear time, assuming that $\max(\mathcal{C}) \leq rem(\mathcal{C}) + 1$. Moreover, both end-elements of the computed sequence γ are of the same color if and only if this color occurs $rem(C) + 1$ times. This completes the proof of the Lemma.

Example 3. We show how the algorithm works for the following coloring:

$$\mathcal{C} = (C_1, C_2, C_3, C_4) = (\{1, 2\}, \{3, 4\}, \{5, 6, 7, 8\}, \{9, 10, 11, 12, 13\})$$

In this case

$$j = 3, \quad C'_3 = \{5, \ 6\}, \ C''_3 = \{7, \ 8\}$$

$$\alpha = (13, 12, 11, 10, 9, 8, 7), \quad \beta = (6, 5, 4, 3, 2, 1)$$

and finally the result of the algorithm is:

$$\gamma = interleave(\alpha, \beta) = (13, 6, 12, 5, 11, 4, 10, 3, 9, 2, 8, 1, 7)$$

Much more complicated is the question of alternating sequences starting and ending in given nodes u, v.

Example 4. Suppose $\mathcal{C} = (\{1, 2\}, \{3, 4\})$ and we impose the condition that the sequence starts in $u = 1$ and ends with $v = 2$. Then there is no such alternating sequence. However if we strengthen the inequality from Lemma 3 to $\max(\mathcal{C}) < rem(\mathcal{C})$ then such a sequence exists.

Lemma 4. *Assume $\mathcal{C}$ is a coloring of n elements, where $n > 1$, $u \neq v$ and $\max(\mathcal{C}) \leq rem(\mathcal{C}) - 1$. Then there is a $\mathcal{C}$-alternating sequence γ from u to v.*

Proof. The case $n \leq 3$ can be checked directly. Hence from now on we assume $n \geq 4$.

Instead of elements u, v we consider their colors A, B. Hence we need a sequence which has fixed colors A, B at its ends (it is not relevant which ends). If we remove u and v from our universe then the resulting coloring $\mathcal{C}'$ satisfies the condition from Lemma 3: we have $\max(\mathcal{C}') \leq rem(\mathcal{C}') + 1$.

Let $\gamma' := \mathit{AlterSeq1}(\mathcal{C}')$ be a $\mathcal{C}'$-alternating sequence missing two elements (with regard to $\mathcal{C}$) with the colors A, B. Now we insert two previously removed elements u, v with colors A and B into γ'. The main point is to show how to do it. We have several cases depending on the colors C, D of the first and the last element of γ'.

Case 1: The trivial case: $\{A, B\} \cap \{C, D\} = \emptyset$.
We insert u at arbitrary end of γ' and v at the other end of γ', thus obtaining a desired alternating sequence γ.

Case 2: Also rather straightforward case: $(A \neq B \ \wedge \ C \neq D)$.
There is a possibilty that either C or D equals either A or B, so we might be constrained with placing u or v at one end of γ'.

Algorithm 2. AlterSeq2($\mathcal{C}$, u, v)

input : $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, $|C_1| \leq |C_2| \leq \ldots \leq |C_k|$, $u \in C_i$, $v \in C_j$, $\max(\mathcal{C}) \leq rem(\mathcal{C}) - 1$, $i < j$
output: $\mathcal{C}$-alternating sequence of $C_1 \cup C_2 \cup \ldots \cup C_k$ with u, v at its ends

1 $C'_i = C_i - \{u\}$;
2 $C'_j = C_j - \{v\}$;
3 $\mathcal{C}' := \{C_1, C_2, \ldots, C_{i-1}, C'_i, C_{i+1}, \ldots, C_{j-1}, C'_j, C_{j+1}, \ldots, C_k\}$;
4 $\gamma' := \mathit{AlterSeq1}(\mathcal{C}')$;
5 $A := color(u)$, $B := color(v)$;
6 $C := color(first(\gamma'))$, $D := color(last(\gamma'))$;
7 **if** $(\{A, B\} \cap \{C, D\} = \emptyset)$ **then**
8 **return** $u\,\gamma'\,v$;
9 **else if** $(A \neq B \ \wedge \ C \neq D)$ **then**
10 **if** $(u\,\gamma'\,v)$ *is an alternating sequence* **then**
11 **return** $u\,\gamma'\,v$;
12 **else**
13 **return** $v\,\gamma'\,u$;
14 **else** // Assume $A = B = C \neq D$
15 $\gamma'' := \gamma'$ with first element removed;
16 $\gamma'' := u\gamma''v$;
17 insert $first(\gamma')$ into γ'' without violating alternating property;
18 return γ'';

Case 3: $(C = D)$

Then by Lemma 3 the color C is *exhausted* and none of A, B equals C. This is in fact a special subcase of **Case 1**.

We can assume now that $C \neq D$ and $A = B$. Without loss of generality let $A = C$. So the only remaing case is as follows.

Case 4: $(A = B = C \neq D)$.

We insert one element of color A after D. We're left with one element of color A, but we cannot put it at the end of our sequence – we have to insert it in between a pair of elements of γ' such that neither of them is of color A. We can use a simple counting argument that it can be done due to the inequality $max(\mathcal{C}) \leq rem(\mathcal{C}) - 1$. We omit technical details.

The whole algorithm is written in *pseudocode* as AlterSeq2. This completes the proof.

Example 5. We show how the algorithm works if **Case 2** applies, let

$$\mathcal{C} = (\{1\}, \{3, 4, 5, 6\}, \{7, 8, 9, 10\}),\ u = 3,\ v = 8.$$

After removing elements u, v we get coloring $\mathcal{C}' = (\{1\}, \{4, 5, 6\}, \{7, 9, 10\})$. Then after applying algorithm from Lemma 3 we obtain:

$$AlterSeq1(\mathcal{C}') = \gamma' = (7\ 5\ 9\ 4\ 10\ 1\ 6).$$

Now we have the second case from the last proof. We insert removed elements at the ends in a suitable way and get the final result:

$$AlterSeq2(\mathcal{C}, 3, 8) = (3\ 7\ 5\ 9\ 4\ 10\ 1\ 6\ 8)$$

Example 6. We show now an example when **Case 1** applies, let:

$$\mathcal{C} = (\{1\}, \{2\}, \{3, 4\}, \{5, 6, 7\}) \quad u = 1, v = 2.$$

We have $n = 7$ and $|C_4| = \max(\mathcal{C}) < \frac{n}{2}$. Then the algorithm constructs the sequence $\gamma = (1\ 5\ 3\ 6\ 4\ 7\ 2)$. In fact there are 12 such sequences.

We are mostly interested in colorings given by r-colorings in trees. Assume until end of the paper that the smallest color corresponds to a singleton set (in case of r-colorings to $\{r\}$).

Lemma 5. *Assume $\mathcal{C}$ is a coloring of n elements with $|C_1| = 1$ and $\max(\mathcal{C}) = rem(\mathcal{C})$. Then there exists $\mathcal{C}$-alternating sequence γ starting from u and ending in v if and only if at least one of u, v is of a largest color.*

Proof. In this case $\max(\mathcal{C}) = rem(\mathcal{C}) = \frac{n}{2}$. Assume C_1 consists of a single element and u is of the largest color. We provide an algorithm for constructing a proper alternating sequence in linear time.

Algorithm 3. AlterSeq3($\mathcal{C}$, u, v)

input : $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, $|C_1| \leq |C_2| \leq \ldots \leq |C_k|$, $|C_1| = 1$, $u \in C_k$, $\max(\mathcal{C}) = rem(\mathcal{C})$
output: $\mathcal{C}$-alternating sequence of $C_1 \cup C_2 \cup \ldots \cup C_k$ with u, v at its ends

1 $\alpha := sequentialize(C_k)$;
2 $\beta := sequentialize(C_{k-1}, C_{k-2}, \ldots, C_1)$;
3 $\gamma := interleave(\alpha, \beta)$;
/* Notice that γ starts with the largest color and ends with C_1 */
4 exchange the first element of γ with u;
5 exchange the last element of γ with v;
6 **return** γ;

Observation. If the assumption $|C_1| = 1$ is dropped then the last lemma is false, for example if $\mathcal{C} = \{\{1, 2\}, \{3, 4\}\}$ and $color(u) = color(v)$ then $\max(\mathcal{C}) = rem(\mathcal{C})$ but there is no $\mathcal{C}$-alternating path from $u$to v.

4 Centroids in Trees

We show that r satisfying Lemma 2 for given u, v can be chosen as one of the potentially at most two nodes minimizing the separability.

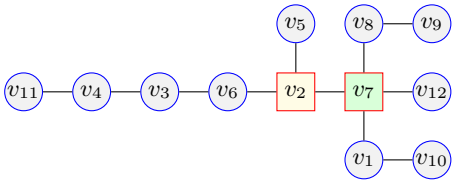


Fig. 3. The nodes v_2 and v_7 are the centroids

The measure of *separability* of a vertex v in tree T, denoted by $\beta_T(v)$, is the size of maximum component of $V(T) - \{v\}$. A vertex v is called a *centroid* if it has minimal separability over all vertices in T.

Lemma 6. [Folklore]
(a) *If a vertex v is a centroid then $\beta_T(v) \leqslant \lfloor \frac{n}{2} \rfloor$.*
(b) *A tree with an odd number of vertices has exactly one centroid.*
A tree with an even number of vertices either has only one centroid v, in which case $\beta_T(v) < \frac{n}{2}$, or it has two adjacent centroids u and v, in which case $\beta_T(u) = \beta_T(v) = \frac{n}{2}$ and $\Delta(u) = \Delta(v)$.

Lemma 7.
(a) *For any chosen pair of distinct nodes u, v for at least one (of at most two) centroids r of T there is a $\mathcal{C}$-alternating path between nodes u, v, where $\mathcal{C}$ is the r-coloring.*
(b) *Let r be a centroid and $\mathcal{C}$ be r-coloring. Then any $\mathcal{C}$-alternating sequence starting with u and ending with v is a maximum weight Hamiltonian path from u to v in K_T.*

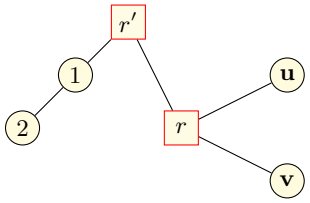


Fig. 4. A bicentroidal tree with centroids denoted by r, r'. There is no alternating path from u to v with respect to r-coloring, but there exists such path with respect to r'-coloring: $\gamma = (u\ 1\ r\ 2\ r'\ v)$.

5 Computing Maximum Hamiltonian Paths

We present a linear time algorithm constructing a maximum Hamiltonian path in K_T between two different nodes $u, v \in V(T)$.

Lemma 8.
For a given tree T we can compute in linear time: centroids of T, $\Delta(v)$ and the distance to each centroid, for all $v \in V(T)$.

First we present a theorem which describes how to compute the maximum value of a path. This theorem follows directly from the results in the previous two sections.

Theorem 1. *Let r be a centroid of T such that there is an alternating sequence γ with respect to r from u to v. We know such r exists. Then γ is the maximum path from u to v in K_T and its weight equals $2 \cdot \Delta(r) - dist(u, r) - dist(v, r)$.*

Corollary 1.
(a) *The maximum weight of a Hamiltonian path in K_T equals $2 \cdot \Delta(r) - dist(r, v)$, where r is one of the centroids of T and v is the closest neighbor of r (i.e. $dist(r, v)$ is minimal).*
(b) *The maximum weight of a Hamiltonian cycle in K_T equals $2 \cdot \Delta(r)$, where r is a centroid of T.*

Lemma 8 together with Lemma 4 and Lemma 5 directly imply the following fact.

Theorem 2. *Algorithm MaxPath(T, u, v) computes a maximum Hamiltonian path in K_T between two given nodes in linear time.*

Algorithm 4. MaxPath(T, u, v)

```
1  if (there are two centroids in T) then
2      choose centroid r such that at least one of u, v is of the largest color
       in r-coloring;
3  else
4      r becomes the unique centroid;
5  C := r-coloring of T;
6  if (T is bicentroidal) then
7      γ := AlterSeq3(C, u, v);
8  else
9      γ := AlterSeq2(C, u, v);
10 return γ;
```

To quickly output the length of a maximum Hamiltonian path between a given pair of nodes in K_T (without the path itself) we need to find the (at most two) centroids of T. For each centroid r we have to compute distance to all other nodes in v. Both operations can be done in linear time. If there's only one centroid r, then we already have all necessary information to compute the value of maximum Hamiltonian paths in constant time. When there are two centroids $r \neq r'$ in tree T then there's one more preprocessing step involved. We need to compute size of the largest subtrees of trees rooted in r and r' and should be able to quickly recognize which vertices belong to these subtrees. This is useful in cases similar to 4 when we need to determine which centroid to use for construction of maximum Hamiltonian paths in K_T. This additional step can also be done in linear time which leads us to the following theorem.

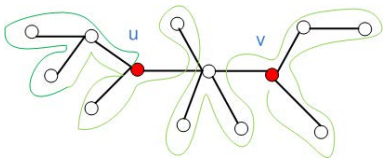
Theorem 3. *We can preprocess a given tree in linear time to allow queries about the value of a maximum path between two nodes in $O(1)$ time.*

6 Final Remarks

We have shown that finding maximum Hamiltonian path between two vertices of K_T is algorithmically and combinatorially interesting. It is based on the algorithm for constructing sequences with alternating colors and choosing centroids as *good* vertices (satisfying assumption of Lemma 2). However the *good* vertex should not be a centroid. For example when the tree is a single path $1-2-3-4-5$, and $u = 4,\ v = 5$ we can chose $r = 2$ as a *good* vertex, but 2 is not a centroid, though $r = 2$ satisfies Lemma 2 in case of this tree.

One can ask a natural related question about the minimum Hamiltonian path between two distinct vertices u, v of K_T. Now there is much less fun and the solution is rather straightforward. We double each edge of T, except the edges on the shortest path from u to v. In this way we obtain a multigraph T'. In this graph each vertex has even degree except u and v. Let us take an Euler path π from u to v in T', see the figure below. Then traverse π and

jump over already visited vertices, vertices are listed when they are visted for the first time. The resulting path γ is the Hamiltonian path in K_T from u to v of minimum weight. Its total weight is the double sum of weights of all edges minus the sum of weights of edges of the shortest path from u to v.



We apply here the technique called *Euler Tour* method. This technique been used previously for trees especially in parallel computing, see [4]. The weight of a minimum weight Hamiltonian cycle is the double sum of weights of all edges of T, a cycle can be generated as a sequence of first-time visited vertices during DFS traversal starting from any node of the tree.

References

1. Steinhaus, H.: One Hundred Problems in Elementary Mathematics. Dover Publications (September 1, 1979)
2. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: The Design and Analysis of Computer Algorithms. Addison-Wesley (1974)
3. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman (1979)
4. Gibbons, A., Rytter, W.: Efficient parallel algorithms. Cambridge University Press (1988)