AN EFFECTIVE EXACT ALGORITHM FOR ONE PARTICULAR CASE OF THE TRAVELING-SALESMAN PROBLEM

Panishev A.V., Plechystyy D.D.

Abstract: The article presents the exact algorithm for solving one case of the job-scheduling problem for the case when the source matrix is ordered by rows.
Keywords: traveling-salesman problem, optimization, optimal path, permutations.

Introduction

In this paper the task is reviewed, for which the space of admissible solutions is the subset of set P of all permutations $\pi = (\pi[1],...,\pi[n])$ of n objects, numbered $1,2,...,n$. It is known, that functional $C(\pi)$, defined on P, reaches it's optimum in class $P_p \subset P$, where P_p is set of all pyramidal permutations $p = (p[1],..., p[n])$.

It should be noted that pyramidal permutation of n elements of set $\{1,2,...,n\}$ produces two pyramidal permutations of $n+1$ elements of set $\{1,2,...,n,n+1\}$ as a result of inserting the element $n+1$ directly before and after the element n. In such way all permutations of set P_p can be produced for any n.

Obviously, the pyramidal permutation $p = (p[1],..., p[n])$ is cyclic, and $P_p \subset P_\tau$, where P_τ is the set of all cyclic permutations of n elements of set $\{1,2,...,n\}$, $|P_\tau| = (n-1)!$. In a pyramidal permutation $p = (p[1],..., p[l-1], \ p[l], p[l+1],..., p[n])$, where $p[l] = n$, one of elements $p[l-1], p[l+1]$ is the element $(n-1)$ of set $\{1,2,...,n\}$.

Let us look at one of the tasks of minimization of functional, defined on the set P_τ of all cyclic permutations $\tau = (\tau[1],...,\tau[n])$, and in particular, at the case, where the optimal solution belongs to class P_p. The reviewed task is one of the fundamental tasks of the scheduling theory and is defined as follows.

Let's consider the conveyor system consisting of m machines job i, $i = \overline{1,n}$, is processed without interruptions. Let t_{ij} be the processing time of stage j of job i, $j = \overline{1,m}$. Let's assume that if $t_{ij} = 0$, then job i is processed by machine j, but the duration of stay at this machine is negligibly small. In the conveyor system processing n jobs schedule π is calculated from matrix $\left[t_{ij}\right]_{n\times m}$ and is defined by matrix of jobs' starting moments $\left[t^o_{ij}\right]_{n\times m}$ or by matrix of jobs' finishing moments $\left[\bar{t}_{ij}\right]_{n\times m}$. Considering these formulas, the moment of finishing all jobs of conveyor schedule π equals

$$\bar{t}_{\max}(\pi) = \max\left\{\bar{t}_{ij} \mid i = \overline{1,n},\ j = \overline{1,m}\right\}. \qquad (1)$$

The goal is to create an optimal schedule π^*, which delivers the minimum of $\bar{t}_{\max}(\pi)$.

It is known that for the described above problem the schedule of minimum duration π^* is located in the class of permutational schedules $\pi = (i_1,...,i_k,...,i_n)$, where i_k is the index of k-th from the left job in sequence π [1].

Conversion between job-scheduling and traveling-salesman problems

In article [1] the task of finding permutation $\pi^* = (i_1^*,...,i_n^*)$ of n uninterruptedly-processed jobs, delivering the minimum of $\bar{t}_{\max}(\pi)$, is formulated as the traveling-salesman problem (TSP) with $n+1$ cities, numbered $0,1,...,n$, and the matrix of costs $\left[c_{pq}\right]_{n+1}$, where $c_{pp} = \infty$, $c_{pq} \geq 0$, $p = \overline{0,n}$, $q = \overline{0,n}$, $p \neq q$. For such variant of TSP the task is to find permutation $\pi_0^* = (0, i_1^*,..., i_n^*)$, delivering the minimum of the following functional on the set of all $(n-1)!$ permutations $\pi_0 = (0, i_1, i_2,..., i_n)$:

$$C(\pi_0) = c_{0i_1} + \sum_{k=1}^{n-1} C_{i_k i_{k+1}} + C_{i_n 0}\ , \tag{2}$$

where $c_{0p} = 0$, $c_{p0} = \sum_{j=1}^{m} t_{pj}$, $c_{pq} = \max_{1 \le l \le m} \left(\sum_{j=1}^{l} t_{pj} - \sum_{j=1}^{l-1} t_{qj} \right)$, $p = \overline{1,n}$, $q = \overline{1,n}$, (2a)

$p \ne q$, $c_{00} = c_{pp} = \infty$.

The solution of TSP in its given formulation gives optimal schedule $\pi^* = (i_1^*, ..., i_n^*)$ for n uninterruptedly-processed jobs in the conveyor system with m machines. In this article we present an algorithm which finds exact solution for the particular case of minimization problem (1) in polynomial time.

If in the source matrix $\left[t_{ij}\right]_{n \times m}$ of problem (1), where $m > 2$, it is true for any pair of jobs i and k that either $t_{ij} \ge t_{kj}$, or $t_{ij} \le t_{kj}$, for all $j = \overline{1,m}$, then matrix $\left[t_{ij}\right]_{n \times m}$ is said to be *ordered by rows.*

Let's assume that in the matrix ordered by rows $t_{ij} \le t_{kj}$ and $i < k$, $j = \overline{1,m}$. If for any pair of jobs i and k in the source matrix ordered by rows it is true that $t_{ij} \ge t_{kj}$, $i < k$, $j = \overline{1,m}$, then we should permute rows in the non-decreasing order of values t_{ij}, $i = \overline{1,n}$, $j = \overline{1,m}$. It is known from article [1] that the optimal solution of problem (1) for the case, when matrix $\left[t_{ij}\right]_{n \times m}$ is ordered by rows, is located in the class of pyramidal permutations $p = (p[1], ..., p[n])$, and such solution can be achieved in polynomial time. We shall describe the procedure, which is faster than the algorithm presented in [1].

Using correlations (2a) we'll transform the matrix $\left[t_{ij}\right]_{n \times m}$ ordered by rows into matrix of costs $\left[c_{pq}\right]_{n+1}$, $p, q = \overline{0,n}$. The sought for solution of the particular case of problem (1) is a sequence $p_o^* = (0, p^*[1], ..., p^*[n]) = (0, p^*)$, where $p^* = (p^*[1], ... \ ..., p^*[n])$ is a permutation of columns of matrix $\left[c_{pq}\right]_{n+1}$ with indexes $1, ..., n$, $p^* \in P_p$.

Let's use the known properties of matrix $\left[c_{pq}\right]_{n+1}$, achieved from transformation of matrix $\left[t_{ij}\right]_{n \times m}$ ordered by rows:

a) $c_{rk} \ge c_{ik}$, $n \ge r \ge i \ge 1$, $r \ne k$, $i \ne k$, $k = \overline{1,n}$;

b) $c_{ik} \ge c_{ij}$, $n \ge j \ge k \ge 1$, $i \ne k$, $i \ne j$, $i = \overline{1,n}$;

c) $c_{rk} - c_{ik} \ge c_{rj} - c_{ij}$, $n \ge j > k \ge 1$, $n \ge r > i \ge 1$, $r \ne k$, $i \ne k$, $r \ne j$, $i \ne j$.

Similarly to the way in which inequality c) was proven [1], we can show that elements of the matrix $\left[c_{pq}\right]_{n+1}$ are connected by inequality

d) $c_{rk} - c_{rj} \ge c_{ik} - c_{ij}$, $n \ge j > k \ge 0$, $n \ge r > i \ge 0$,
$r \ne k$, $i \ne k$, $r \ne j$, $i \ne j$.

Sequence $p = (p[1], ..., p[n])$ defines closed route or path $\tau(p) = (0, p[1], ..., p[i], ..., p[n], 0)$, in which all $p[i] \in N$ are different. Cost of path $\tau(p)$ is calculated as follows

$$C(\tau(p)) = c_{0p[1]} + \sum_{i=1}^{n-1} c_{p[i]p[i+1]} + c_{p[n]0}\ , \tag{3}$$

where $c_{0p[1]} = 0$ $c_{0p[1]} = 0$.

Thus the minimization task (1) for the case, when the source matrix of processing times $\left[t_{ij}\right]_{n \times m}$ is ordered by rows, is a traveling-salesman problem with target functional (3), defined on the set of all pyramidal permutations P_p, and with the matrix of costs $\left[c_{pq}\right]_{n+1}$, whose elements are bound by inequalities a)-d).

Let $p_l = (p[1], ..., p[k-1], p[k], p[k+1], ..., p[l])$ be one of pyramidal permutations of subset $\{1, 2, ..., l\}$ of set N, $l > 2$, $p[k] = l$. Cost of path $\tau(p_l)$, corresponding to p_l, can be calculated from table $\left[c_{pq}\right]_{l+1}$, which is achieved by deleting rows and columns numbered $l+1, ..., n$ from matrix $\left[c_{pq}\right]_{n+1}$:

$C\left(\tau\left(p_l\right)\right)=\sum_{i=1}^{l-1} C_{p[i]p[i+1]}+C_{p[i]0}$.

Sequence p_l produces two partial pyramidal permutations with size $l+1$:

$p_{l+1}^1=\left(...,p\left[k-1\right],l+1,p\left[k\right],p\left[k+1\right],...\right)$,

$p_{l+1}^2=\left(...,p\left[k-1\right],p\left[k\right],l+1,p\left[k+1\right],...\right)$

The costs of corresponding paths are:

$C\left(\tau\left(p_{l+1}^1\right)\right)=C\left(\tau\left(p_l\right)\right)+c_{p[k-1](l+1)}+c_{(l+1)p[k]}-c_{p[k-1]p[k]}$,

$C\left(\tau\left(p_{l+1}^2\right)\right)=C\left(\tau\left(p_l\right)\right)+c_{p[k](l+1)}+c_{(l+1)p[k+1]}-c_{p[k]p[k+1]}$.

The general idea of the offered algorithm of minimization of (3) is step-by-step construction of partial pyramidal permutations with size l, $l=\overline{3,n}$, and exclusion of those permutations which lead to loss of optimal solution.

First of all it should be noted that when $l=3$ there are two pairs of pyramidal permutations and corresponding paths: $\tau\left(p_3^1\right)=\left(0,1,3,2,0\right)$, $\tau\left(p_3^3\right)=\left(0,3,2,1,0\right)$ и $\tau\left(p_3^2\right)=\left(0,1,2,3,0\right)$, $\tau\left(p_3^4\right)=\left(0,2,3,1,0\right)$. As a result of path comparison in received pairs using properties a)-d) we can choose those pairs, which do not lead to loss of optimal solution of task of minimization of functional (3).

The algorithm

Let's consider two pyramidal permutations p_l, σ_l and corresponding paths:

$\tau\left(p_l\right)=\left(0,...,p\left[k-1\right],p\left[k\right],p\left[k+1\right],...,0\right)$, $p\left[k\right]=l$, $\mathrm{k}=\overline{1,l}$;

$\tau\left(\sigma_l\right)=\left(0,...,\sigma\left[i-1\right],\sigma\left[i\right],\sigma\left[i+1\right],...,0\right)$, $\sigma\left[i\right]=l$, $i=\overline{1,l}$.

If $k=l$ then $p\left[k+1\right]=0$. The same way, if $i=l$ then $\sigma\left[i+1\right]=0$.

If $p\left[1\right]=l$ then $p\left[0\right]=0$, and if $\sigma\left[1\right]=l$, then $\sigma\left[0\right]=0$.

There're following possible cases of comparison of paths $\tau\left(p_l\right)$ and $\tau\left(\sigma_l\right)$.

1. $p\left[k-1\right]=\sigma\left[i-1\right]$, $p\left[k+1\right]=\sigma\left[i+1\right]$. Obviously, the permutation with greater cost should be excluded. If $C\left(\tau\left(p_l\right)\right)=C\left(\tau\left(\sigma_l\right)\right)$ then either one of two permutations p_l and σ_l can be excluded.

2. $p\left[k+1\right]=\sigma\left[i+1\right]=l-1$. It can be shown that if $p\left[k-1\right]>\sigma\left[i-1\right]$ and $C\left(\tau\left(p_l\right)\right)\le C\left(\tau\left(\sigma_l\right)\right)$ then only permutation p_l should be considered later on.

Permutation $p_l=\left(...,p\left[k-1\right],l,l-1,...\right)$, $l<n$, produces two partial permutations $p_{l+1}^1=\left(...,p\left[k-1\right],l+1,l,l-1,...\right)$ and $p_{l+1}^2=\left(...,p\left[k-1\right],\ ,l,l+1,l-1,...\right)$.

3. $p\left[k-1\right]=\sigma\left[i-1\right]=l-1$. It can be shown that if $p\left[k+1\right]>\sigma\left[i+1\right]$ and $C\left(\tau\left(p_l\right)\right)\le C\left(\tau\left(\sigma_l\right)\right)$ then permutation σ_l should be excluded from consideration.

Permutation $p_l=\left(...,l-1,l,p\left[k+1\right],...\right)$, $l<n$, produces two partial permutations with size $l+1$: $p_{l+1}^1=\left(...,l-1,l+1,l,p\left[k+1\right],...\right)$ and $p_{l+1}^2=\left(...,l-1,l,l+1,p\left[k+1\right],...\right)$.

4. $p\left[k+1\right]=\sigma\left[i+1\right]=l-1$, $p\left[k-1\right]>\sigma\left[i-1\right]$, $C\left(\tau\left(\sigma_l\right)\right)<C\left(\tau\left(p_l\right)\right)$, $l<n$.

Properties c) and d) do not allow to exclude from consideration either one of permutations p_l and σ_l. If all possible permutations with size $l+1$ are produced from p_l and σ_l then it is clear that in case 4 if $C\left(\tau\left(p_{l+1}^1\right)\right)\le C\left(\tau\left(\sigma_{l+1}^1\right)\right)$ only two of partial solutions σ_{l+1}^2 and p_{l+1}^1 need to be considered further, or if $C\left(\tau\left(\sigma_{l+1}^1\right)\right)<C\left(\tau\left(p_{l+1}^1\right)\right)$ then only three permutations σ_{l+1}^2, p_{l+1}^1, σ_{l+1}^1 need to be considered.

5. $p\left[k-1\right]=\sigma\left[i-1\right]=l-1$, $p\left[k+1\right]>\sigma\left[i+1\right]$, $C\left(\tau\left(\sigma_l\right)\right)<C\left(\tau\left(p_l\right)\right)$, $l<n$.

The same way as in case 4, both permutations p_l and σ_l need to be expanded. If $C\left(\tau\left(p_{l+1}^2\right)\right) \le C\left(\tau\left(\sigma_{l+1}^2\right)\right)$ then only two permutations $\sigma_{l+1}^1 = \left(...,l-1,l+1,l,\sigma\left[i+1\right],...\right)$ and $p_{l+1}^2 = \left(...,l-1,l,l+1,p\left[k+1\right],...\right)$ remain for consideration, or if $C\left(\tau\left(\sigma_{l+1}^2\right)\right) < C\left(\tau\left(p_{l+1}^2\right)\right)$ then three partial solutions σ_{l+1}^1, p_{l+1}^2, $\sigma_{l+1}^2 = \left(...,l-1,l+1,\sigma\left[i+1\right],...\right)$ need to be considered further to avoid loss of optimal solution of task (3).

Let us describe the exact algorithm for solution of limited version of TSP, formulated by (3).

S0. $\left[c_{pq}\right]_{n+1}$ – matrix of costs for TSP, in which values c_{pq}, $p = \overline{0,n}$, $q = \overline{0,n}$ are bound by properties a)-d);

Build paths $\tau_3^1 = (0,1,3,2,0)$, $\tau_3^2 = (0,1,2,3,0)$, $\tau_3^3 = (0,3,2,1,0)$, $\tau_3^4 = (0,2,3,1,0)$, calculate their costs $C\left(\tau_3^1\right)$, $C\left(\tau_3^2\right)$, $C\left(\tau_3^3\right)$, $C\left(\tau_3^4\right)$; set $l = 3$, $m_l = 4$.

S1. If $l = n$ then finish: the cost of the optimal path τ^* equals $C\left(\tau^*\right) = \min\left\{C\left(\tau_l^{j_l}\right) \mid j_l = \overline{1,m_l}\right\}$.

S2. Separate set $\left\{\tau_l^{j_l} \mid j_l = \overline{1,m_l}\right\}$ into two subsets T_R and T_L in such manner that in each path belonging to set T_R index $l-1$ comes after l, and in each path belonging to set T_L it comes before l. Exclude from considerations "hopeless" paths in T_R and T_L, using adequately cases 1,2 and 1,3.

S3. $l = l+1$; expand every of remaining paths into two paths $(...,l,l-1,...)$ and $(...,l-1,l,...)$, calculate their costs $C\left(\tau_l^{j_l}\right)$; go to step S1.

In the following passage we prove that the algorithm correctly solves the particular case of TSP, minimizing functional (3).

Conclusion

Let us describe the process of construction of minimal-cost path τ^* using the tree of n levels. The only vertex of the first level, which is the root of the tree, is matched by path $(0,1,0)$ and its cost c_{10}. The second level consists of 2 vertices $\tau_2^1 = (0,1,2,0)$, $\tau_2^2 = (0,2,1,0)$ and their costs $C\left(\tau_2^1\right) = c_{12} + c_{20}$ and $C\left(\tau_2^2\right) = c_{21} + c_{10}$. Every second-level vertex produces two third-level vertices $\tau_3^1 = (0,1,3,2,0)$, $\tau_3^2 = (0,1,2,3,0)$, $\tau_3^3 = (0,3,2,1,0)$, $\tau_3^4 = (0,2,3,1,0)$, which define the set of all pyramidal permutations of size 3. Vertex τ_2^1 produces a pair of descendants τ_3^1 and τ_3^2, and vertex τ_2^2 produces τ_3^3 and τ_3^4. Every third-level vertex $\tau_3^{j_3}$, $j = \overline{1,4}$, is assigned a weight equal to the cost of corresponding path: $C\left(\tau_3^1\right) = C\left(\tau_2^1\right) + c_{13} + c_{32} - c_{12}$, $C\left(\tau_3^2\right)$, $C\left(\tau_3^3\right)$, $C\left(\tau_3^4\right)$. Any vertex of level l, $l = \overline{4,n}$, produces not more than two descendants.

The set of all paths $\tau_n^{j_n}$, $j_n = \overline{1,m_n}$, containing the optimal task solution, can be presented as a union of non-overlapping subsets T_{32} and T_{23}. Subset T_{32} consists of all paths in which index 2 follows index 3, while subset T_{23} contains all paths where index 2 comes before index 3. Obviously, values $C\left(\tau_3^1\right)$ and $C\left(\tau_3^3\right)$ limit from below the minimum of functional (3) defined on subset T_{32}. In the same way costs $C\left(\tau_3^2\right)$ and $C\left(\tau_3^4\right)$ are lower bounds for the minimum of functional (3) on subset T_{23}.

During comparison of paths τ_3^1 and τ_3^3 which compose subset T_R, if $C\left(\tau_3^1\right) \le C\left(\tau_3^3\right)$ then τ_3^3 is excluded, which means that all admissible solutions $\tau_n^{j_n} \in T_{32}$ with lower bound greater or equal to $C\left(\tau_3^3\right)$ are also excluded. Thus at level 4 vertex τ_3^1 produces two vertices $\tau_4^1 = (0,1,4,3,2,0)$ and $\tau_4^2 = (0,1,3,4,2,0)$, $\tau_4^1 \in T_R$, $\tau_4^2 \in T_L$.

Comparison of paths τ_3^2 and τ_3^4 which compose subset T_L, in the case when $C\left(\tau_3^4\right) \le C\left(\tau_3^2\right)$ allows to exclude τ_3^2 and all admissible solutions in T_{23} for which the lower bound of (3) is no less than $C\left(\tau_3^2\right)$. At level $l = 4$ vertex τ_3^4 produces two vertices $\tau_4^3 = (0,2,4,3,1,0)$, $\tau_4^4 = (0,2,3,4,1,0)$, $\tau_4^3 \in T_R$, $\tau_4^4 \in T_L$.

If $C(\tau_3^3) < C(\tau_3^1)$ then as a result of comparison of costs of paths $(0,1,3,4,2,0)$ and $(0,3,4,2,1,0)$, correspondingly equal to $C(\tau_3^1) + c_{34} + c_{42} - c_{32}$ and $C(\tau_3^3) + c_{34} + c_{42} - c_{32}$, path $(0,1,3,4,2,0)$ is excluded from subset T_{32}, and, obviously, all admissible solutions like $(0,1,3,...,n,...,2,0)$ are also excluded. In this case at level $l = 4$ vertices τ_3^1 and τ_3^3 produce three vertices $\tau_4^1 = (0,1,4,3,2,0)$, $\tau_4^2 = (0,4,3,2,1,0)$, $\tau_4^3 = (0,3,4,2,1,0)$, τ_4^1, $\tau_4^2 \in T_R$, $\tau_4^3 \in T_L$.

If $C(\tau_3^2) < C(\tau_3^4)$ then as a result of comparison of costs of paths $(0,1,2,4,3,0)$ and $(0,2,4,3,1,0)$, correspondingly equal to $C(\tau_3^2) + c_{24} + c_{43} - c_{23}$ and $C(\tau_3^4) + c_{24} + c_{43} - c_{23}$, partial solution $(0,2,4,3,1,0)$ and all admissible solutions like $(0,2,...,n,...,3,1,0)$ from subset T_{23} are excluded. Then at level 4 vertices τ_3^2 and τ_3^4 produce three vertices $\tau_4^4 = (0,1,2,4,3,0)$, $\tau_4^5 = (0,1,2,3,4,0)$, $\tau_4^6 = (0,2,3,4,1,0)$, $\tau_4^4 \in T_R$, τ_4^5, $\tau_4^6 \in T_L$.

In general case as a result of comparing paths received at level $l-1$, $l = \overline{3,n-1}$ there are formed subsets of partial solutions T_R and T_L, where the former one contains solutions like $(...,l,l-1,...)$ and the latter one contains solutions like $(...,l-1,l,...)$. At the n-th level of solution tree set $T_R \cup T_L$ is matched by a subset of set $T_{32} \cup T_{23}$ of admissible tasks solution, containing the optimal solution.

Using the obvious logic it can be shown that at level l, $l = \overline{2,n}$, the number of partial solutions excluding the loss of optimal one is limited by value $2l-2$.

Let us estimate the upper bound for the number of all solution-tree vertices:

$$\sum_{l=1}^{n} m_l \le \sum_{l=1}^{n} (2l-2) = n(n-1).$$

The above-given arguments allow us to make the following statement.

Theorem. The algorithm presented by a sequence of steps S0-S3 constructs solution which delivers the minimum of functional (3) in time $O(n^2)$.

Bibliography

[1] Танаев В.С., Сотсков Ю.Н., Струсевич В.А. Теория расписаний. Многостадийные системы. – М: Наука, 1989. 327с.

[2] Axsaeter S. On scheduling in a semi-ordered flow shop without intermediate queues // AIIE Trans. 1982.v. 14, №2. pp. 128-130.

Author information

Anatoliy Vasilyovich Panishev – head of the chair of informatics and computer modeling, professor, doctor of technical sciences, Zhitomir State Technological University, Chernyahovskogo 103, Zhitomir, Ukraine, 10003; e-mail: dep@ziet.zhitomir.ua

Dmytro Dmytrovych Plechystyy – teacher-assistant of the chair of informatics and computer modeling, aspirant, Zhitomir State Technological University, Chernyahovskogo 103, Zhitomir, Ukraine, 10003; e-mail: dep@ziet.zhitomir.ua