European Journal of Operational Research 230 (2013) 464–474

Contents lists available at SciVerse ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Solving the shortest path tour problem

P. Festa [a], F. Guerriero [b,*], D. Laganà [b], R. Musmanno [b]

[a] Department of Mathematics and Applications, University of Napoli "Federico II", Compl. MSA, Via Cintia, 80126 Naples, Italy
[b] Department of Mechanical, Energy and Management Engineering, University of Calabria, Ponte Pietro Bucci, Building 41/C, 87036 Arcavacata di Rende (CS), Italy

A R T I C L E I N F O

Article history:
Received 27 March 2012
Accepted 16 April 2013
Available online 7 May 2013

Keywords:
Network flows
Shortest path
Structure path constraints
Labeling method

A B S T R A C T

In this paper, we study the shortest path tour problem in which a shortest path from a given origin node to a given destination node must be found in a directed graph with non-negative arc lengths. Such path needs to cross a sequence of node subsets that are given in a fixed order. The subsets are disjoint and may be different-sized. A polynomial-time reduction of the problem to a classical shortest path problem over a modified digraph is described and two solution methods based on the above reduction and dynamic programming, respectively, are proposed and compared with the state-of-the-art solving procedure. The proposed methods are tested on existing datasets for this problem and on a large class of new benchmark instances. The computational experience shows that both the proposed methods exhibit a consistent improved performance in terms of computational time with respect to the existing solution method.

1. Introduction

The shortest path tour problem ($\mathcal{SPTP}$) is a variant of the shortest path problem ($\mathcal{SPP}$) and appeared for the first time in the scientific literature in Bertsekas's dynamic programming and optimal control book [2].

The $\mathcal{SPTP}$ has been recently studied by Festa in [13]. The paper of Festa is the first systematic contribution for solving the $\mathcal{SPTP}$. The author proved that the problem belongs to the complexity class **P**. The polynomial Karp-reduction of the $\mathcal{SPTP}$ to the single-source single-destination $\mathcal{SPP}$ involves the construction of an expanded graph in which different algorithms for the $\mathcal{SPP}$ were tested and compared on pseudo-randomly generated instances. The results presented in [13] showed that Dijkstra's algorithm outperforms all the competitor algorithms.

Applications of the $\mathcal{SPTP}$ arise for example in the context of the manufacture workpieces, where a robot has to perform at least one operation selected from a set of S types of operations. In such case, the problem may be modeled as a $\mathcal{SPTP}$ in which operations are associated with nodes of a directed graph and the time needed for a tool change is represented by the distance between two nodes (see [13]).

The main scientific contribution of this paper consists in analyzing some basic theoretical properties of the $\mathcal{SPTP}$, in designing a dynamic programming-based algorithm ($\mathcal{DPA}$) for solving it, and showing how an ad hoc algorithm for acyclic graphs may be used to solve the $\mathcal{SPTP}$ after efficiently reducing it to a classical $\mathcal{SPP}$ through the method referred to as modified graph algorithm ($\mathcal{MGA}$).

The remainder of the paper is organized as follows. The problem is formally described in Section 2. The state-of-the-art algorithm to address the $\mathcal{SPTP}$ is presented in Section 3. Some properties concerning the reducibility of the problem to a classical $\mathcal{SPP}$ and the relevant consequences in terms of solvability are described in Section 4. A dynamic programming algorithm is illustrated in Section 5. Computational results and the analysis of the performance of the proposed algorithms are presented in Section 6. The paper ends with some concluding remarks stated in Section 7.

2. Problem description

Consider a directed graph $G = (N,A)$ defined by a set of nodes $N := \{1, \ldots, n\}$ and a set of arcs $A := \{(i,j) \in N \times N\colon\ i \neq j\}$, where $|A| = m$. A non-negative length c_{ij} is assigned to each arc $(i,j) \in A$. Let $F(i) := \{j \in N\colon (i,j) \in A\}$ and $B(i) := \{j \in N\colon (j,i) \in A\}$ be the *forward star* and *backward star* associated with each node $i \in N$, respectively. Moreover, let S denote a certain number of node subsets $T_1, \ldots, T_S$ such that $T_h \cap T_k = \emptyset, h, k = 1, \ldots, S, h \neq k$.

Given two nodes $i_1, i_\pi \in N, i_1 \neq i_\pi$, the *path* P_{i_1,i_π} from i_1 to i_π is defined as a sequence of nodes $P_{i_1,i_\pi} = \{i_1, \ldots, i_\pi\}$ such that $(i_j, i_{j+1}) \in A, j = 1, \ldots, \pi - 1$. Observe that $i_j, j = 1, \ldots, \pi$, represents the node index occurring in position j in path P_{i_1,i_π}. A path P_{i_1,i_π} is said to be *elementary* whether $i_l \neq i_j,\ l, j = 1, \ldots, \pi$ and $l \neq j$. We refer to the length of path P_{i_1,i_π} as $l(P_{i_1,i_\pi})$ representing the sum of the lengths of the arcs connecting consecutive nodes in P_{i_1,i_π}, i.e., $l(P_{i_1,i_\pi}) = \sum_{j=1}^{\pi-1} c_{j,j+1}$.

The $\mathcal{SPTP}$ aims at finding a shortest path $\mathcal{P}_{s,d}$ from origin node $s \in V$ to destination node $d \in V$ in the directed graph G, such that it

* Corresponding author. Tel.: +39 0984 494620.
E-mail addresses: paola.festa@unina.it (P. Festa), guerrier@unical.it (F. Guerriero), demetrio.lagana@unical.it (D. Laganà), musmanno@unical.it (R. Musmanno).

0377-2217/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ejor.2013.04.029

visits successively and sequentially the following subsets T_k, $k = 0, \ldots, S+1$, such that $T_0 = \{s\}$ and $T_{S+1} = \{d\}$. Note that sets T_k, $k = 1, \ldots, S$, must be visited in exactly the same order in which they are defined.

Consequently, a path P_{i_1,i_π} is said to be a *feasible solution* for the $\mathcal{SPTP}$ if:

$$\exists\, g_0, g_1 \ldots g_{S+1} \in [1,\pi] : g_0 < g_1 < \cdots < g_{S+1},\\ i_{g_0} \in P_{i_1,i_\pi} \cap T_0, i_{g_1} \in P_{i_1,i_\pi} \cap T_1, \ldots, i_{g_{S+1}} \in P_{i_1,i_\pi} \cap T_{S+1}. \quad (1)$$

Conditions (1) mean that an increasing sequence of natural numbers exists such that the corresponding nodes of the path P_{i_1,i_π} belong to the ordered sequence of subsets $T_0, T_1, \ldots, T_{S+1}$. A small instance of the $\mathcal{SPTP}$ is depicted in Fig. 1, where $N = \{s = 1, 2, 3, 4, 5, 6, d = 7\}$, $S = 2, T_0 = \{s = 1\}$, $T_1 = \{3\}$, $T_2 = \{2, 4\}$, $T_3 = \{d = 7\}$. The shortest path from node 1 to node 7 is $\mathcal{P}_{1,7} = \{1,3,7\}$ with length 5, while the shortest path tour between the same origin and destination nodes is $P_{1,7} = \{1, 3, 2, 3, 7\}$ with length 11. Such path is not elementary, since it passes twice through node 3.

3. The state-of-the-art

The state-of-the-art consists of the expanded graph method proposed by Festa in [13], and referred to as $\mathcal{EGA}$ in the sequel. A brief description of how the $\mathcal{EGA}$ works is given in the following.

The $\mathcal{EGA}$ relies on a polynomial-time reduction algorithm that transforms any $\mathcal{SPTP}$ instance defined on a single-stage graph G into a single-source single-destination $\mathcal{SPP}$ instance defined on a multi-stage graph $G' = (V',A')$ with $S+2$ stages, each replicating G, and such that $V' = \{1, \ldots, (S+2)n\}$ and $|A'| = (S+1)m$. More precisely, the reduction algorithm performs the following operations:

i. $V' := \{1, \ldots, (S+2)n\}$; $A' := \emptyset$;
ii. at each iteration, an arc(a,b) is added to A'. In particular, for each stage $k \in \{0, \ldots, S\}$, for each node $v \in \{1, \ldots, n\}$, and for each adjacent node $w \in FS(v)$, $(a,b) = (v + kn, w + (k+1)n)$ with length c_{vw}, if $w \in T_{k+1}$; $(a,b) = (v + kn, w + kn)$ with length c_{vw}, otherwise.

Since $|A'| = (S+1)m$, the computational complexity of the reduction algorithm is $O(Sm)$.[1] Once the multi-stage graph G' is obtained, to solve the resulting $\mathcal{SPP}$ any shortest path algorithm can be applied. By applying Dijkstra's algorithm that uses a binary heap for storing temporary node labels, the overall worst case computational complexity of $\mathcal{EGA}$ is $O(|A'|\log|V'| + |V'|\log|V'|)$, which is dominated by $O(|A'|\log|V'|)$, that is $O(Sm\log n)$.

4. A modified graph method

In this section, we will focus on some basic properties related to the reducibility of any $\mathcal{SPTP}$ instance into a single-source single-destination $\mathcal{SPP}$ instance.

4.1. On the reduction of the $\mathcal{SPTP}$ to the $\mathcal{SPP}$

Given an instance of the $\mathcal{SPTP}$ on a directed graph $G = (N,A)$ the following definition is applied.

[1] For bounding the computational complexity of an algorithm, several *asymptotic notations* are used [6]. If n is the input size, the computational complexity of an algorithm $f(n)$ is $O(g(n))$, if there exist positive constants a and n_0 such that $0 \leqslant f(n) \leqslant ag(n)$ for all $n \geqslant n_0$.

Definition 1. Let $G^{(a)} = (N^{(a)},A^{(a)},c^{(a)})$ be a weighted directed graph obtained from G in such a way that:

- $N^{(a)} = \bigcup_{k=0}^{S+1} T_k$;
- $A^{(a)} = \bigcup_{k=0}^{S} A_k^{(a)}$, where $A_k^{(a)} := \{(i,j) \in T_k \times T_{k+1} : i \in T_k \text{ and } j \in T_{k+1}\}$;
- $c^{(a)} : A^{(a)} \mapsto \mathbb{Z}^+$ is a function that associates an integer non-negative number $c_{ij}^{(a)}$ to each arc $(i,j) \in A^{(a)}$, where $c_{ij}^{(a)} := l(P_{i,j})$ is the length of a shortest path from node $i \in T_k$ to node $j \in T_{k+1}$ on graph G.

The following property holds for the arc set $A^{(a)}$.

Property 1. *Let $G^{(a)}$ be the weighted directed graph associated with an instance of the $\mathcal{SPTP}$, then $|A^{(a)}| \leqslant n(n-2)$.*

Solving a $\mathcal{SPTP}$ instance may be performed by finding a shortest path in $G^{(a)}$. Indeed, the following property holds.

Property 2. *Every path $P_{s,d}^{(a)}$ from s to d in $G^{(a)}$ defines a $\mathcal{SPTP}$ solution in G with the same cost, and vice versa.*

Such a property derives from the construction of graph $G^{(a)}$ given in Definition 1.

The optimal cost of any $\mathcal{SPTP}$ instance is equal to the cost of the shortest path from node s to node d computed on $G^{(a)}$, as shown in the following property.

Property 3. *The cost of a shortest path $\mathcal{P}_{s,d}^{(a)}$ in $G^{(a)}$ is equal to the cost of an optimal $\mathcal{SPTP}$ in G.*

Proof. Such property can be proved by contradiction. From Property 2, it follows that the shortest path $\mathcal{P}_{s,d}^{(a)}$ in $G^{(a)}$ corresponds to a feasible path tour $P_{s,d}^*$ in G. Hence, suppose that a shortest path tour $P_{s,d}$ different from $P_{s,d}^*$ exists in G, such that $l(P_{s,d}) < l\left(P_{s,d}^*\right)$. This means that a feasible path $P_{s,d}^{(a)'}$ exists in $G^{(a)}$, associated with $P_{s,d}$, whose cost is less than the cost of the shortest path $P_{s,d}^{(a)}$. This conclusion contradicts the initial assumption. □

4.2. Solving the $\mathcal{SPTP}$ as $\mathcal{SPP}$

The framework of the proposed $\mathcal{MGA}$ is sketched in Algorithm 1:

Algorithm 1. Modified graph algorithm

```
Step 1 (Initialization)
Compute a shortest path from each node $i \in T_k$ to each node
  $j \in T_{k+1}$, $k = 0, \ldots, S$.
Step 2 (Graph Construction)
Build the digraph $G^{(a)} = (N^{(a)},A^{(a)})$ as detailed in Definition 1.
Step 3 (Topological enumeration of $N^{(a)}$)
Step 4 (Shortest Path Computation)
Let $l(\mathcal{P}_{1,i})$ denote the length of the shortest path from node
  $s = 1$ to node $d = i$, and $p(i)$ denote
the predecessor node of $i$ in $\mathcal{P}_{1,i}$.
Set $l(\mathcal{P}_{1,1}) := 0, p(1) := 1$ and $iteration_{\mathcal{MGA}} := 0$.
for all $j = 2, \ldots, |N^{(a)}|$ do
  $l(\mathcal{P}_{1,j}) = \min_{i \in N^{(a)}:(i,j) \in A^{(a)}} \{l(\mathcal{P}_{1,i}) + c_{ij}\}$.  Update $p(j)$ with the
  value of $i$ whereby the minimum of $l(\mathcal{P}_{1,j})$ occurs.
  Update the number of iterations as
  $iteration_{\mathcal{MGA}} = iteration_{\mathcal{MGA}} + 1$.
end for
```