The Traveller's Problem

Iva Babukova

October 6, 2016

In this work we present the Traveller's Problem (TP), a computational task whose extensions and variations are often encountered by travellers around the world. The task is concerned with creating a valid travel schedule, using airplanes as a means of transportation and in accordance with certain constraints specified by the traveller.

1 Problem Formulation

Each instance of TP consists of:

1. A set of airports $A = \{A_0, ..., A_n\}$. Each airport $A_i \in A$ represents a location the traveller can begin their commute in, visit as a desired destination, or connect in on the way to their destination.

2. The trip starts and ends at the same airport A_0, which is referred to as the *home point*.

3. The total travel time T, within which the traveller must have visited all destinations and returned to the home point.

4. A set of flights $F = \{f_0, ..., f_m\}$. Each flight f_j has:

   - departure airport A_j^d,
   - arrival airport A_j^a,
   - date t_j,
   - duration Δj,
   - cost c_j,

   for some non-negative integer j less than or equal to n. The date t_j is a positive integer less than or equal to T that shows at which day f_j leaves its departure airport. The duration Δj is a positive fraction that shows the amount of time that takes for flight f_j to go to A_j^a from A_j^d. The cost c_j is a positive number that denotes the number of units of some currency ϵ that the traveller pays in order to be able to board flight f_j.

5. A set of *destinations* $D = \{D_0, ..., D_l\}$, $D \subseteq A$. The traveller must visit every airport in D.

A solution to any instance of TP is a sequence s of k valid flights, $f_1, f_2, ..., f_k$. We say that s is valid if the flights in s have the following properties:

1. $A_1^d = A_k^d = A_0$
2. For any two consecutive flights f_i and f_{i+1}, $A_i^a = A_{i+1}^d$.
3. For any two flights in s f_i and $f_{i+\delta}$, $t_i < t_{i+\delta}$ for some integers $i, \delta > 0$.
4. For any two consecutive flights f_i and f_{i+1} in s, $t_{i+1} \geq t_i + \Delta_{i+1}$.
5. $t_k + \Delta_k \leq T$

Note that a valid sequence of flights may contain 0, 1 or more flights to and from airports that are not destinations. Such airports are called *connections*.

The *optimization* version of TP (TPO) asks for an *optimal* solution, which is a solution that minimizes the total sum of the prices of the flights in s, denoted as $c(s)$.

The *decision* version of TP (TPD) asks whether there exists a valid sequence of flights s, such that the sum of the costs of all flights in s is less than or equal to some integer B. The solution of this problem is a 'yes' or 'no' answer.

There exist a variety of additional constraints and extensions that can be added to TP. Our problem formulation has only presented the hard constraints so far. Every valid solution to a TP instance must satisfy these constraints. In real-world problems, travellers may have additional preferences (soft constraints) and requirements (hard constraints) with regards to their travel. These are discussed in the next two sections.

1.1 Hard Constraints

- Travellers may require to spend a given date at a given destination, for example due to an event occurring on that date in this destination.
- Travellers may have a certain price threshold (budget) B, such that if the total travel cost exceeds that threshold the traveller will discard a solution as invalid.

1.2 Soft Constraints

It may be desirable to search for a solution that satisfies some of the following requirements:

- Travellers may wish to spend a certain amount δ_i of days in each destination D_i, where δ_i may be specified as a lower or an upper bound.

- Travellers may wish to avoid taking connection flights. In such requirement, we wish to maximise the number of flights to and from destinations.
- Travellers may want to spend as little time on flying as possible. In such case, we wish to find a solution that minimises the sum of the durations of all flights. This can be viewed as a multi-objective or lexicographic optimisation. In the former case, we need to find a solution that satisfies more than one hard constraint simultaneously. The latter assumes that the constraints can be ranked in an order of importance. To what extend a given constraint is satisfied depends on its ranking.

It is therefore suggested that any attempt at an investigation of TP assumes as an additional non-functional requirement that any proposed model to solve TP is flexible and can be easily extended by adding, removing and modifying the aforementioned constraints.

2 Worked Example

We present an example instance of TP and its most optimal solution. Note that in this example we do not wish to satisfy any of the previously listed soft constraints.

Example. A traveller wishes to visit 4 airports from a set of 7 airports available to travel to and from:

Glasgow (G), Berlin (B), Milan (M), Amsterdam (A), Paris (P), Frankfurt (F), London (L).

Airport G is the home point, F and L are connections, and B, M, A and P are the destinations. The travel time of the traveller is 15 days. All available flights are listed on Table 1. For simplicity, the duration of each flight is assumed to be 1 day.

Solution. A valid solution of the TP instance in the example above is the sequence of flights s, where the each flight is represented by its flight number, specified in the first column of Table 1:

$$s = \{GA1, AP4, PM6, MF9, FB11, BG13\}$$

The total flights cost $c(s)$ is 699.

A more optimal solution is the following sequence:

$$s' = \{GA1, AP4, PM6, MF9, FB11, BL13, LG14\}$$

Here $c(s')$ is 483. s' is considered as more optimal, because $c(s')$ is less than $c(s)$. This shows that s can not be the optimal solution and potentially s' is the one.

Flight No	**Departs**	**Arrives**	**Date**	**Price**
GA1	G	A	1	74
GF1	G	F	1	86
GL3	G	L	3	25
MF3	M	F	3	120
AP4	A	P	4	58
LG14	L	G	14	24
PL12	P	L	12	45
PM6	P	M	6	71
MF9	M	F	9	39
FB11	F	B	11	122
FM8	F	M	8	234
FM12	F	M	12	250
BG13	B	G	13	335
BL13	B	L	13	102

Table 1: List of flights with departure and arrival airports, flight date and price.

3 Complexity of TP

We state a theorem about the complexity of TP and and prove it.

Theorem 1. The decision version of TP is NP-complete.

Proof. This proof first shows the membership of TP in the NP class of problems. Second, we prove the NP-hardness of TP by constructing a polynomial-time reduction from a known NP-complete problem Π to TP, where Π is chosen to be the traveling salesman problem (TSP). Its NP-hardness follows by a reduction from the Hamiltonian Cycle problem. The proof is presented in [**?**].

Given an instance of TP and s, which is a sequence of flights from F, we can write an algorithm V that checks in polynomial time whether s is a valid solution. To accept or reject validity, V only needs to traverse s and check that each of the properties of a valid solution are satisfied. Therefore, TP is in NP.

Let π be an instance of TSP that has:

- a set A of n cities,
- a distance $\mathrm{d}(A_i,\, A_j)$ between each pair of cities $A_i,\, A_j \in A$, and
- a positive integer B.

TSP asks whether there exists a sequence of cities $\gamma = A_1, A_2, ..., A_n$, that starts and ends at the same city A_1 and visits all cities in A, having *length* B or less. The sequence γ is called a *tour* and the *length* of γ L_γ is equal to the sum of the distances from each city to its successor in γ, that is:

$$L_\gamma = \sum_{i=1}^{i=n-1} d(A_i, A_{i+1}); A_i, A_{i+1} \in \gamma$$

Let π' be an instance of TPD with the following properties:

- The set of airports is identical to the set of cities in π and it is similarly denoted as A (a city in π is called an airport in π').
- Each airport in A is also a destination.
- T is equal to $n + 1$.
- For every $f_k \in F$ with A_k^d and A_k^d its departure and arrival airports respectively, c_k is equal to $\mathrm{d}(A_k^d,\, A_k^a)$ in π.
- For every $f_k \in F$, $\Delta_k = 1$.
- B is the upper bound on the allowed total flights cost.
- Let C be the Cartesian product of the airports in A with itself, that is $A \times A = \{(A_i, A_j) : A_i \in A, A_j \in A, \mathrm{i} \neq \mathrm{j}\}$. Then F be a set of flights, such that for every $(A_i, A_j) \in C$, there exists a flight f_k in F, where $A_k^d = A_i$ and $A_k^d = Aj$ for every date $t \leq T$.

Suppose that $\gamma = A_1, A_2, ..., A_n$ is solution of π with $L_\gamma \leq B$. In π', γ is equivalent to the order of visited airports by some sequence of flights $s = f_1, f_2, ...f_n$, such that for every $f_k \in s, 1 \leq k \leq n$:

- $A_k^d = A_k, A_k \in \gamma$, and
- $A_k^a = \begin{cases} A_{k+1} & \text{if } k < n, \\ A_1 & \text{if } k = n \end{cases}$,

From the two conditions above it follows that s satisfies property 1 and 2 of a valid solution. To satisfy property 3, 4 and 5 it is sufficient to choose flights from F such that for every $f_k \in s$, $\Delta_k = k$. We know that such flights exist in F by the construction of F. Therefore, s is a solution of π', from which it follows that a solution of π is also a solution of π'.

Let s be a solution of π', where flights in s visit destinations in the sequence $\gamma' = A_1, A_2, ..., A_n$. By construction of π', γ' contains all airports in A and $L_{\gamma'} \leq B$. Clearly, γ' is a valid solution of π. Therefore, a solution of π' is also a solution of π.

The transformation from a TSP instance to an instance of TP can be done in polynomial time. For each of the $n(n\text{-}1)/2$ distances d(A_i, A_j) that must be specified in π, it is sufficient to check that the same cost is assigned to the flights from A_i to A_j for all dates.

Therefore, TP is in NP and TSP can be reduced to TPD in polynomial time, from which it follows that TP is NP-complete.

□

4 Existing work

No existing work on this problem has been found so far in the literature. However, a similar problem, namely TSP has been a subject to extensive research in the recent years.

TSP

TSP with time windows (TSPTW)

vehicle-routing problem (VRP)

job-shop scheduling problem (JSSP)