Time-Dependent Bi-Objective Itinerary Planning Algorithm: Application in Sea Transportation*

Aphrodite Veneti[1], Charalampos Konstantopoulos[2], and Grammati Pantziou[3]

1 **Department of Informatics, University of Piraeus, Greece**
aveneti@webmail.unipi.gr
2 **Department of Informatics, University of Piraeus, Greece**
konstant@unipi.gr
3 **Department of Informatics, Technological Educational Institute of Athens, Greece**
pantziou@teiath.gr

Abstract

A special case of the Time-Dependent Shortest Path Problem (TDSPP) is the itinerary planning problem where the objective is to find the shortest path between a source and a destination node which passes through a fixed sequence of intermediate nodes. In this paper, we deviate from the common approach for solving this problem, that is, finding first the shortest paths between successive nodes in the above sequence and then synthesizing the final solution from the solutions of these sub-problems. We propose a more direct approach and solve the problem by a label-setting approach which is able to early prune a lot of partial paths that cannot be part of the optimal solution. In addition, we study a different version of the main problem where it is only required that the solution path should pass through a set of specific nodes irrespectively of the particular order in which these nodes are included in the path. As a case study, we have applied the proposed techniques for solving the itinerary planning of a ship with respect to two conflicting criteria, in the area of the Aegean Sea, Greece. Moreover, the algorithm handles the case that the ship speed is not constant throughout the whole voyage. Specifically, it can be set at a different level each time the ship departs from an intermediate port in order to obtain low cost solutions for the itinerary planning. The experimental results confirm the high performance of the proposed algorithms.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems; G.2.2 Graph Theory

Keywords and phrases Multi-criteria optimization, Label setting algorithm, Time dependent networks, Travel planning, Itinerary planning, Sea transportation

Digital Object Identifier 10.4230/OASIcs.ATMOS.2016.11

1 Introduction

The Time-Dependent Shortest Path Problem (TDSPP) is a fundamental and well studied multi-objective optimization problem with many applications. However, less effort has been made for addressing the Time-Dependent Shortest Path Problem that must go through a

* The publication of this paper has been partly supported by the University of Piraeus Research Center. Also, in this work the research carried out by the first author was partially funded by Onassis Scholarship Foundation.

16th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS'16).
Editors: Marc Goerigk and Renato Werneck; Article No. 11; pp. 11:1–11:14

Open Access Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

given sequence of nodes, which is also known as travel planning problem. In this case, apart from optimizing the selected objective, the given sequence defines intermediate nodes that must be visited after departing from the origin and before arriving at the destination. This problem arises from both travel industry and daily life activities. Trip planning applications, such as location-based services and car navigation systems need efficient algorithms for the earliest arrival problem as well as for multi-criteria problems in large road networks. Usually, the intermediate nodes represent specific Points of Interest (POIs) and the duration of each intermediate stop varies. As we are working with a time-dependent network, we need to know in advance how long the user expects to stay at each POI and specify the exact departure time from the intermediate POI.

An important property of the network which much differentiates the complexity of the time-dependent algorithms is the FIFO property [4, 12, 19]. A network is said to fulfill the FIFO property if all of its arcs fulfill that property i.e., for each arc (i, j) of the network, earlier departure from i always leads to earlier arrival at j, that is, the arrival events at j are in the same chronological order as the departure events at i. More explicitly, it may be defined in the following mathematical form:

$$\forall(i, j, t), \quad t + c(i, j, t) \leq (t + 1) + c(i, j, t + 1) \tag{1}$$

where cost function $c(i, j, t)$ denotes the cost for traversing arc (i, j) at time instance t and has integer-valued domain and positive integer-valued range. When the FIFO property holds, waiting at the nodes of the network is pointless since leaving immediately from each node is always a beneficial practice leading to optimal solution paths. Computing shortest paths in FIFO networks is a polynomially solvable problem [12]. On the other hand, when the FIFO property does not hold, optimal solutions may require waiting at certain nodes of the network. Therefore, in non-FIFO networks the complexity of the time-dependent shortest path problem depends on the waiting policy at nodes. If waiting is allowed, the problems is polynomially solvable otherwise, the problem is NP-hard [18].

This paper is motivated by itinerary planning problems in sea transportation. The duration of a voyage and the distance travelled in such a voyage are the two causal factors which determine the voyage cost [8]. The time spent while at a port is also accounted in the total voyage time. Especially, in short distance voyages, the delay incurred in ports is much more important for the whole voyage duration than the travelling time itself [15]. In particular, we address the problem of finding the Pareto optimal set of paths which pass through a fixed sequence of nodes with predefined visiting time and specific time constraints at each intermediate node, in a time-dependent setting. Indeed, there may be several constraints that should be considered when drawing up a travel plan. For instance, in maritime there are several charter types which imply different constraints. For example, a voyage charter specifies a period, known as laytime, for loading and unloading the cargo. If laytime is exceeded, the charterer must pay demurrage. If laytime is saved, the charter party may require the ship owner to pay despatch to the charterer. Moreover, in a contract of Affreightment, apart from the period in which the transfer of the cargo must be carried, the route of the voyage is also specified. On the contrary, in a time charter, only the period of time is defined and the charterer is responsible for the selection of the ports that the vessel approaches. The arrival time at a port affects directly the total visiting time at that port since in case of congestion at that particular port, the ship may have to queue for port facilities. Thus, besides optimizing several economical, safety and ecological objectives, it is also crucial to take into account the constraints imposed by the strict schedule of a vessel. To this end, the ship speed may be different but within an acceptable range for each travel

leg between successive intermediate ports in such a way that the stay at each port incurs the least operational cost.

A greedy approach to solve the itinerary planning problem is to search for independent shortest paths locally between each pair of successive stops. More specifically, we first locate the nearest stop from the query location, then we locate the nearest stop from the current stop, and so on, until the destination is reached. As discussed in [21], the travel planning problem can not be solved optimally by using this approach and this clearly also applies for the time-dependent problem.

In [1], the bi-criteria itinerary problem for time-dependent networks is studied. The proposed method is based on the decomposition of the problem into a sequence of elementary itinerary sub-problems, solved by a backward dynamic programming algorithm. Adapting also the Bellman's backward optimality principle, the solution of the sub-problems starts from the destination and traverses the route backwards using the Decreased Order Time (DOT) technique described in [4].

In [3], a travel planning problem was proposed which consists in finding the best travel plan from a origin to a destination that follows a given sequence of nodes on a transportation network with deterministic time-dependent travel times. The authors proposed a decomposition scheme in which the whole problem is divided into sub-problems and each of them is solved as a one-to-many shortest path problem by adding a surrogate node to the graph.

In [5], an algorithm is proposed for finding the shortest distance route that passes through a fixed sequence of POIs in a time-dependent road network. The method is based on A^* algorithm, together with a suitable admissible heuristic function and a pruning scheme that reduces the search space. The method is also applicable to static road networks.

Another relevant problem was proposed in [14], termed as the Trip Planning Query (TPQ). In TPQ, only the subset of the intermediate POIs is defined by the user. Aim of the optimization problem is to find the path that minimizes the travel cost between origin and destination, and subsequently to define the visiting order of POIs. As this problem is NP-hard due to the existence of multiple possibilities in POI ordering, the authors proposed a number of approximation algorithms.

Recent research for trip planning in public multi-modal transportation networks has produced several speedup techniques and algorithms. In this paper, we focus on the travel itinerary problem in sea transportation. The objective is to reach the destination port, at minimum fuel consumption and maximum safety, visiting all the predefined intermediate ports and at the same time, respecting the constraint on the travelling time as well as other technical and operational restrictions. The itinerary planning problem can be cast as a multi-objective, non-linear optimization problem with constraints where the desired solution should be found among a number of conflicting objectives.

The main complicating factor in sea transportation is the weather. Regardless of the specific objectives, the cost functions in this kind of transportation depend heavily on weather conditions and thus the itinerary planning problem for ships is a time-dependent problem. Besides weather variations, moving obstacles with known or unknown trajectories, such as other vessels or marine protected populations, are additional factors that make the problem even more dynamic.

In our problem setting, the ship speed along each sub-route between successive intermediate ports are decision variables whose values should be optimized for returning low cost solutions. It is worth mentioning that since the ship speed can vary only within a certain range due to operational constraints and since the relation between fuel consumption and speed is approximated by a cubic function [20], it is always beneficial to sail at the lowest

allowable speed that does not disrupt the ship schedule. Indeed, when fuel consumption is the only objective, it has been proven that this strategy for selecting the ship speed gives the optimal solution [10]. However, since our problem is time-dependent and there is also the objective of the minimum incurred risk other than the fuel consumption, the above rule cannot be applied for finding the optimal speed. For example, a high speed may negatively affect the fuel consumption criterion but it may help in avoiding adverse weather conditions. Thus, sailing at the minimum allowable speed is not profitable for both objectives in this case.

The rest of the paper is organised as follows. In the next section, the Time-Dependent Bi-criteria Shortest Path Problem with fixed sequence of intermediate stops is defined. In section 3, the proposed algorithm is described and then in section 4, the algorithm is tested in maritime scenarios for finding ship itineraries. Finally, the conclusions are discussed in section 5.

2 Preliminaries

Firstly, a description of the Time-Dependent Bi-criteria Shortest Path Problem (TDBiSPP) is given and then we define the itinerary planning problem addressed in this paper.

Let $G = (V, A)$ be a directed graph, where V is the set of nodes and A is the set of arcs with $|V| = n$ and $|A| = m$. Each arc $(i, j) \in A$ is associated with three attributes $c_1(i, j, t, U)$, $c_2(i, j, t, U)$ and $pt(i, j, t, U)$, whose values are assumed to be non-negative and may change over time; $c_1(i, j, t, U)$ and $c_2(i, j, t, U)$ denote the two costs for traversing (i, j) with speed U and $pt(i, j, t, U)$ denotes the travel time for traversing (i, j), departing from node i at time instance t with speed U. In the TDBiSPP the speed U is assumed to be constant. The two costs are the objectives to be minimized, while the travel time is the "resource" constrained in the problem. The frozen arc model [19] is also assumed where the arc cost is determined at the arrival time at the tail of the arc and does not change during its traversal.

Let $C_1(P)$, $C_2(P)$ denote the total cost of a path P according to the first and the second criterion respectively, and $PT(P)$ denote the total travel time along P. Given a start node $s \in V$, a destination node $d \in V$, a departure time t_{start}, an upper bound T on the maximum permissible total travel time, with no waiting at nodes, the problem is to find a path P from s to d departing at t_{start} from s that minimizes the two objectives $C_1(P)$ and $C_2(P)$ without violating the travel time constraint i.e., $PT(P) \leq T$. Since the objective functions may be conflicting, a single solution that simultaneously optimizes each objective may not exist and therefore, the problem actually is to find the set of Pareto optimal or non dominated solutions. A solution is termed as a non dominated solution if none of the objectives can be improved in value without degrading the value of the other objective. Formally, a path p is said to dominate another path p' if $(C_1(p) < C_1(p')$ and $C_2(p) \leq C_2(p'))$ or $(C_1(p) \leq C_1(p')$ and $C_2(p) < C_2(p'))$.

In TDBiSPP with fixed sequence of intermediate stops, besides the usual origin and destination node, a fixed sequence of nodes through which the path must pass is also given as an input. This sequence of intermediate stops will be denoted by IS where $IS \subseteq V$. For each intermediate stop is_k, the Earliest Arrival Time (EAT) and the Latest Arrival Time is specified and thus is_k should be visited within the time window $[EAT_{is_k}, LAT_{is_k}]$. Also, the visiting time of is_k may not be always the same but may vary according to the type of the vessel and the arrival time at is_k. Specifically, we use the notation $L(is_k, t)$ to denote the laytime of a vessel reaching the intermediate stop is_k at time instance t. Moreover, a vessel is allowed to extend its laytime in a port if that helps in avoiding congestion or adverse

weather conditions. Recall also that waiting at intermediate stops is beneficial in non-FIFO networks. However, although waiting is allowed at stops, it might be an upper bound on this time. Thus, we define the Latest Departure Time (LDT) for each intermediate stop is_k denoted by LTD_{is_k}.

Upon departing from an intermediate stop or the origin node, we have to decide the nominal cruising speed U which must lie in the interval $\{U_{min}, U_{max}\}$ where U_{min} and U_{max} are the minimum and the maximum allowable speed respectively. Along a sub-route between successive intermediate stops, the nominal speed is assumed to be constant.

It is now clear that the itinerary planning problem can be solved as a Time-Dependent Shortest Path Problem (TDSPP) where the arcs are labelled with a pair of costs, namely fuel consumption and risk incurred along the arcs. With regard to the TDSPP classification, the itinerary planning problem is characterized by the following:

- Goal: minimization of two costs.
- Decision variables: ship course and ship speed along each sub-route.
- Waiting at stops: waiting is allowed only at the intermediate stops and not at any other node of the input graph.
- Source and destination: one source to one destination node through many intermediate stops.
- Network properties: non-periodic network and non-FIFO regarding the costs c_1 and c_2.

3 The proposed bi-objective algorithm

We propose a forward label setting algorithm (Algorithm 1) for finding the set of Pareto optimal paths in time-dependent non-FIFO networks, visiting also a fixed sequence of intermediate destinations, based on a Time-Dependent Bi-criteria Shortest Path algorithm [24].

In order to improve the performance of the algorithm, we employ a heuristic function h to estimate a lower bound of the passage time of the path from the current position to the final destination passing through the remaining intermediate stops. Summing the current cost of the path (denoted as g) and the h value, we compute the f value, which is a lower bound of the total passage time of the path. We use this estimation to check if the extension of the current, partial, path will violate the constraint of the total passage time T. Using this heuristic, we can prune paths from a very early stage of the algorithm, since we can be sure that this path is never going to reach the destination node in time. The heuristic function is computed in a preprocessing phase by running a single-objective Dijkstra algorithm for finding the least passage time path from the destination to any other node of the input graph. For this computation, we use as edge weight the least passage time cost of each edge occurring within the passage time window namely, $[t_{start}, t_{start} + T]$ and as ship speed the maximum allowable speed U_{max}. Likewise, we also employ two heuristic functions $h_1(i, k, k')$ and $h_2(i, k, k')$ with $k < k'$ to estimate a lower bound of costs C_1 and C_2 respectively of the path starting from the node i and visiting the $k^{th}, (k+1)^{th}, \ldots, k'^{th}$ intermediate stop in that order. These heuristic estimates are used for checking the domination relation between two different paths. Similarly with the heuristic function h, the heuristic functions h_1 and h_2 are pre-computed by running a single-objective Dijkstra algorithm for finding the optimal path from each intermediate stop to any other node of the input graph, that minimizes the fuel consumption and the risk respectively. For this computation, we use as edge weight the least fuel consumption and risk cost of each edge occurring within the passage time window

$[t_{start}, t_{start} + T]$ and for all possible speed levels. All heuristic functions, h, h_1 and h_2, are admissible and hence they provide a lower bound on the corresponding real cost value.

Algorithm 1 keeps a set of labels $l_i(t)$ where i is a node and t a time instance. Each label is a tuple of six elements namely, $l_i(t) = (C_1, C_2, j, prev\_ptr, k, Spd)$ and corresponds to a path from the source node s to i arriving at node i exactly at time t. C_1, C_2 is the total cost of the estimated path from s to i with respect to the two criteria c_1, c_2 respectively. The integer j is the predecessor node of i on the path from s to i. The pointer $prev\_ptr$ points to the Pareto optimal label of the j whose extension gave $l_i(t)$. The integer k denotes that the most recently visited intermediate stop is the k^{th} stop in the IS. The integer Spd is the nominal travel speed between the k^{th} and the $(k+1)^{th}$ intermediate stop.

We also need to revisit the definition of dominance between two paths. Suppose two paths p and p' starting from the same source node and leading to the same destination node i. We assume also that paths p and p' have already visited the k and k'^{th} intermediate stop respectively and $k' < k$. The path p is said to dominate p' ($p \prec p'$) if ($C_1(p) < C_1(p') + h_1(i, k'+1, k)$ and $C_2(p) \leq C_2(p') + h_2(i, k'+1, k)$) or ($C_1(p) \leq C_1(p') + h_1(i, k'+1, k)$ and $C_2(p) < C_2(p') + h_2(i, k'+1, k)$). In case that the two paths p and p' have already visited exactly the same intermediate nodes ($k' = k$), they are comparable without using the heuristic functions h_1 and h_2 and the path p is said to dominate another path p' ($p \prec p'$) if ($C_1(p) < C_1(p')$ and $C_2(p) \leq C_2(p')$) or ($C_1(p) \leq C_1(p')$ and $C_2(p) < C_2(p')$). Finally, when $k' > k$ and it holds either that $C_1(p) + h_1(i, k+1, k') < C_1(p')$ and $C_2(p) + h_2(i, k+1, k') \leq C_2(p')$ or that $C_1(p) + h_1(i, k+1, k') \leq C_1(p')$ and $C_2(p) + h_2(i, k+1, k') < C_2(p')$, no conclusion about the dominance between these two paths can be safely reached; path p must surely be extended till it reaches the intermediate node k' and this additional route may eventually make the cost of p higher than that of p' even though the heuristic estimates show otherwise.

It is important to note that the FIFO property does not hold for the cost functions employed in our scenario, since leaving as much as earlier from a node does not necessarily reduce the cost of the outgoing edge. Thus, there is no possible way to know the ideal time of leaving a node in advance for obtaining an optimal cost path to the destination node. As a result, for each node and for each arrival time instance at this node, we should keep all the non-dominated labels referring to that particular arrival time. In contrast, in a static bi-criteria shortest path problem there is no need to partition the labels at each node according to their arrival time and only a single set of non-dominated labels can be kept at each node.

Algorithm 1 iterates over all integer time instances in the interval $[t_{start}, T + t_{start}]$, where t_{start} is the departure time from the source node and T is the maximum allowable total travel time. During the execution, the algorithm maintains two groups of label lists at each node and for each time instance namely, the permanent and temporary lists. The labels of the temporary list of a node for a given time instance are all transferred one by one to the corresponding permanent list when the algorithm iteration proceeds to that particular instance. However, just before that transfer, each of these labels is extended, thus creating a new label. Each of these new labels is kept only if it does not violate the total travel time constraint (line 19) and also if it is not dominated by another label belonging to the same node with the same arrival time (lines 31 and 47). In the case that a label corresponds to the next intermediate stop along the path, it must satisfy the earlier and the latest arrival time constraint (line 23) in addition. In this case, we must also consider the laytime at the intermediate stop and the fact that the speed can change. As a consequence, a label corresponding to an intermediate stop is replicated in order to take into account all possible

departure time instances from the intermediate stop and all allowable ship speed values (lines 27-28). Also, labels already in the temporary list for that arrival instance are compared against this newly generated label and discarded when they are dominated by the new label (line 51). The order in which the labels of the current time instance are extended is not actually important, since all labels of that time instance should be extended first before proceeding to the next iteration step. Thus, the label to be extended (pivot label) each time can be selected randomly (line 9). By the end of the execution, all Pareto-optimal solution paths will have been stored as permanent labels of the destination node d. Notice also that for the node d, no label extension takes place (line 15) since the algorithm does not need to go further down the destination node due to the fact that the cost functions are non-negative-valued and hence going further only adds to the total cost of the path. Also, in Algorithm 1, we assume a single departure time from the source node but it is trivial to handle the case of multiple possible departure time instances too.

The same algorithm can be applied for solving the problem when the only requirement is to visit a set of intermediate stops regardless of the particular visit order. Notice that this version of the problem is NP-hard even in the case of only one objective by reduction from the Hamiltonian path problem [6]. Now, each label should keep the stops already visited and in the domination check between two labels l_i and l_j, l_i wins if its intermediate stops are superset of the stops of l_j and the total estimated costs of the extended path of l_j which also includes the missing intermediate stops from l_i are higher than the corresponding ones of the path of l_j. Furthermore, the heuristic functions h, h_1, h_2 are redefined as $h(i, S)$, $h_1(i, S)$, $h_2(i, S)$ where S is a subset of intermediate stops that should be visited at minimum cost and in any order starting from node i. In the case of h, after visiting the stops in S, the path should also reach the final destination. For computing these functions, all possible permutations of stops in S should be generated, keeping that which derives the lowest cost. Although computing the heuristic functions for all possible subsets S and nodes in advance seems to be a heavy computation, in practice it is not, because the set of unvisited intermediate stops is small. Finally, during the generation of a new label, at line 21, we need to check that this node is an intermediate stop which has not already been considered as such along the so far constructed path.

4 Experimental results

We tested the performance of the proposed itinerary planning algorithm in a maritime application where ship itineraries should be determined. In this problem, the objective is to reach the destination port after visiting a predefined sequence of intermediate ports, at minimum fuel consumption and maximum safety, respecting the constraint on the travelling time as well as other technical and operational restrictions. The proposed algorithm is compared to the common approach of decomposing the bi-criterion itinerary planning problem with mandatory intermediate stops into a series of bi-criterion shortest path problems between successive intermediate stops [1]. For a fair comparison, we have also enhanced the basic algorithm in [1] with heuristic functions analogous with those employed in the proposed algorithm.

4.1 Case Study

The two algorithms were tested in finding ship routes in the region of the Aegean sea in Greece. For modelling this area, the grid structure developed for the AMINESS system [7] is used. The data grid structure holds a great amount of data, spatially stored in the

grid nodes and edges. Static as well as dynamic information is taken into account, such as geographic and bathymetric data, protected areas, risk estimation [13] as well as weather and sea state predictions. A grid point is assumed to be valid if it does not correspond to a landmass point. All valid grid points correspond to nodes which are connected with each other via bidirectional edges. Each node is connected to the 16 geographically closest nodes in the grid. Furthermore, voyage safety is enhanced by following the IMO recommendations [11] for avoiding dangerous situations. Regarding IMO recommendations, surf-riding and broaching-to situations should be avoided when navigating in severe weather conditions. Surf-riding and broaching-to may occur when the conditions below are satisfied:

$$135^o < a < 225^o \text{ and } V_R > \frac{1,8\sqrt{L}}{cos(180 - a)}$$

with a, V_R and L being the ship-wave angle, the speed and the length of the ship respectively.

Since ship speed is constant along each sub-route, the only way to avoid the above situations is by rejecting the edges of the grid where these conditions arise.

As has been already mentioned, the objectives to be optimized are the fuel consumption and the risk. For each possible nominal speed U and edge e, the fuel consumption and incurred risk for traversing e with speed U is assigned to e. For calculating the navigation time between the endpoints of an edge, we have to consider the actual ship speed along the edge which is usually lower than the nominal one due to the added resistances induced by irregular waves and wind during ship navigation. To this end, the ship model described in [16] is used.

This model is general, independent of specific ship features. According to this model, the speed reduction depends only on the significant wave height H and the ship-wave relative direction Θ.

Estimating the fuel consumption rate along a vessel route is a complex issue still under investigation. As a rule of thumb, the following formula is used in practice [2, 9, 17, 23, 22]:

$$F = K \cdot P \tag{2}$$

where F is the rate of fuel consumption measured in kg/h, K is the specific fuel consumption of the ship and P is the engine power in BHP[1] (kW) of the ship. The engine power of a ship also determines the nominal speed of the ship. Now, the total fuel consumption along a route with constant engine power P in the ship and hence constant nominal speed is the product of its passage time PT and the rate of fuel consumption per hour F:

$$FC_{total} = F \cdot PT \tag{3}$$

Concerning implementation setup, algorithms were implemented in C++. The experiments were performed on a system with an Intel(R) Xeon(R) E5-2430 v2 processor at 2.50GHz and 16 GB RAM. It is also worth mentioning that we did not exploited any parallelism in this multi-core architecture during the tests.

For each test case, a random set of starting, intermediate and destination ports were chosen and the maximum passage time for each voyage was proportional to the sum of the straight line distances connecting the consecutive ports. The speed of the ship ranges between 12 and 25 knots.

[1] Brake HorsePower (BHP) is the total measure of engine power at the output shaft of the engine.

Algorithm 1 Bi-objective, Fixed Sequence, Time-Dependent and Time-Constrained Shortest Path Algorithm

Require: $G = (V, A)$, and C, the cost matrix for all arcs $(i, j) \in A$
Ensure: All Pareto optimal paths from the node s to the node d passing through a fixed sequence of intermediate nodes

s: the start node
d: the destination node
$\mathbf{t_{start}}$: the departure time
T: the maximum allowed total travel time
$\mathbf{IS} = (is_1, is_2, \ldots, is_k)$: the fixed sequence of intermediate nodes
$\mathbf{EAT_j}$: the Earliest Arrival Time at intermediate node j
$\mathbf{LAT_j}$: the Latest Arrival Time at intermediate node j
$\mathbf{LDT_j}$: the Latest Departure Time from intermediate node j
$\mathbf{L(j, t)}$: the laytime of an intermediate stop arriving at intermediate node j at time instance t
U: the nominal speed at which the ship sails since departing from the most recent intermediate stop
$\mathbf{pt(i, j, t, U)}$: the travel time for traversing the arc (i, j) with speed U departing from the node i at time instance t
$\mathbf{fc(i, j, t, U)}$: the total fuel consumption for traversing the arc (i, j) with speed U departing from the node i at time instance t
$\mathbf{r(i, j, t, U)}$: the total risk incurred for traversing the arc (i, j) with speed U departing from the node i at time instance t
$\mathbf{t^j_{ar}}$: an arrival time instance at the node j
$\mathbf{l_i(t)}$: a label of the node i corresponding to the path from s to i, arriving at i at time instance t
$\mathbf{Ltemp_i(t)}$: the list of temporary labels of node i at time instance t
$\mathbf{Lperm_i(t)}$: the list of permanent labels of node i at time instance t
$\mathbf{card(l_i(t), Lperm_i(t))}$: the position of $l_i(t)$ in the list of permanent labels of node i
$\mathbf{l^P_i(t)}$: the p^{th} component of a label $l_i(t)$

/* Initialization of temporary and permanent labels of every node and for all $t \in \{t_{start}, t_{start} + 1, ..., T + t_{start}\}$ */
$Ltemp_i(t), Lperm_i(t) \leftarrow \emptyset, \quad \forall i \in V, \forall t \in \{t_{start}, t_{start} + 1, ..., T + t_{start}\}$
/* Initialization of temporary labels of source node for all $t \in \{t_{start}, t_{start}+1, ..., T+t_{start}\}$ and for all $U \in \{U_{min}, U_{max}\}$ */
for $U = U_{min}$ to U_{max} **do**
    $Ltemp_s(t_{start}) \leftarrow \{(0, 0, null, null, 0, U)\}$
end for

for $t_{ar} = t_{start}$ to $T + t_{start}$ **do**
    while $(\cup_{i \in V} Ltemp_i(t_{ar}) \neq \emptyset)$ **do**
        Select a pivot label $l_{i^*}(t_{ar})$ from $\cup_{i \in V} Ltemp_i(t_{ar})$
/* Remove $l_{i^*}(t_{ar})$ from $Ltemp_i(t_{ar})$ and add it to $Lperm_{i^*}(t_{ar})$ */
        $Ltemp_{i^*}(t_{ar}) \leftarrow Ltemp_{i^*}(t_{ar}) \backslash \{l_{i^*}(t_{ar})\}$
        $Lperm_{i^*}(t_{ar}) \leftarrow Lperm_{i^*}(t_{ar}) \cup \{l_{i^*}(t_{ar})\}$
/* Store the position of label $l_{i^*}(t_{ar})$ in the list $Lperm_{i^*}(t_{ar})$ */
        $p \leftarrow card(l_{i^*}(t_{ar}), Lperm_{i^*}(t_{ar}))$
        if $i^* \neq d$ **then**

/* Label all the successors of i^* */
  for all $(i^*, j) \in E$ **do**
    $t^j_{ar} \leftarrow t_{ar} + pt(i^*, j, t_{ar}, l^6_{i^*}(t_{ar}))$
/* Check the total duration constraint */
    if $t^j_{ar} - t_{start} + h \leq T$ **then**
/* Check if successor node j is the next Intermediate Stop */
      if $j = is_{(l^5_{i^*}(t_{ar})+1)}$ **then**
/* Check the EAT and LAT constraint at Intermediate Stop j */
        if $t^j_{ar} \geq EAT_j$ and $t^j_{ar} \leq LAT_j$ **then**
/* Take into account the laytime at Intermediate Stop j */
          $t^j_{ar} \leftarrow t^j_{ar} + L(j, t^j_{ar})$
          for $t = t^j_{ar}$ to LDT_j **do**
            for $U = U_{min}$ to U_{max} **do**
              $l_j(t) \leftarrow ({l_{i^*}}^1(t_{ar}) + fc(i^*, j, t_{ar}, l^6_{i^*}(t_{ar})),\ {l_{i^*}}^2(t_{ar})$
                $+r(i^*, j, t_{ar}, l^6_{i^*}(t_{ar})),\ i^*,\ p,\ j,\ U)$
/* Check that there is no label $l'_j(t)$ of node j at time instance t dominating label $l_j(t)$*/
              if $\nexists l'_j(t) \in Ltemp_j(t) : l'_j(t) \prec l_j(t)$ **then**
/* Store the label $l_j(t)$ of node j at time instance t as temporary */
                $Ltemp_j(t) \leftarrow Ltemp_j(t) \cup \{l_j(t)\}$
/* Delete all temporary labels of node j at time instance t dominated by $l_j(t)$ */
                $Ltemp_j(t) \leftarrow Ltemp_j(t) \backslash \{l'_j(t) \in Ltemp_j(t)\ and$
                  $l_j(t) \prec l'_j(t)\}$
              end if
            end for
          end for
        end if
      end if
    else
/* node j is not the next Intermediate Stop */
      $l_j(t^j_{ar}) \leftarrow ({l_{i^*}}^1(t_{ar}) + fc(i^*, j, t_{ar}, l^6_{i^*}(t_{ar})),\ {l_{i^*}}^2(t_{ar})$
        $+r(i^*, j, t_{ar}, l^6_{i^*}(t_{ar})),\ i^*,\ p,\ {l_{i^*}}^5(t_{ar}), l^6_{i^*}(t_{ar}))$
/* Check that there is no label $l'_j(t^j_{ar})$ of node j at time instance t^j_{ar} dominating label $l_j(t^j_{ar})$ */
      if $\nexists l'_j(t^j_{ar}) \in Ltemp_j(t^j_{ar}) : l'_j(t^j_{ar}) \prec l_j(t^j_{ar})$ **then**
/* Store the label $l_j(t^j_{ar})$ of node j at time instance t^j_{ar} as temporary */
        $Ltemp_j(t^j_{ar}) \leftarrow Ltemp_j(t^j_{ar}) \cup \{l_j(t^j_{ar})\}$
/* Delete all temporary labels of node j at time instance t^j_{ar} dominated by $l_j(t^j_{ar})$ */
        $Ltemp_j(t^j_{ar}) \leftarrow Ltemp_j(t^j_{ar}) \backslash \{l'_j(t^j_{ar}) \in Ltemp_j(t^j_{ar})\ and\ l_j(t^j_{ar})$
          $\prec l'_j(t^j_{ar})\}$
      end if
    end if
  end for
end if
end while
end for
/* Output:$Lperm_d$, all Pareto optimal paths from the node s to the node d */
return $Lperm_d$

Table 1 Itinerary planning with predefined visiting order of intermediate ports and with no other constraints in these ports.

Case No	Start Time	Maximum Travel Duration (hours)	No of Intermediate Stops	CPU time (in seconds)	
				Decomposition Algorithm	Proposed Algorithm
1	10:00 am	4	1	88	57
2	1:00 pm	4.5	2	102	68
3	3:00 pm	5	3	117	79
4	3:30 pm	6	4	140	92
5	7:00 am	6.5	5	151	100
6	8:00 am	7.5	6	169	113
7	9:00 am	8.5	7	189	146
8	8:30 am	9.5	8	203	156
9	7:30 am	10	9	224	171
10	7:00 am	11	10	235	194

4.2 Computational Results

Firstly, the proposed algorithm is evaluated in the case that there are no constraints on the arrival or departing time from the intermediate ports, however, waiting at these ports is forbidden. Thus, $\forall j \in IS$ we can assume that EAT_j is equal to zero, LAT_j is equal to the maximum voyage duration and $LDT_j = t^j_{ar}$ (see Algorithm 1, Line 27), since a vessel can not wait at an intermediate port. The only constraint imposed is on the arrival time at the final destination. In Table 1, the experimental results clearly show that the proposed algorithm has lower execution time than that of the common decomposition approach. In the next experimental setup, the arrival time at each intermediate port is not arbitrary as in the first case but should be within a certain time window. This scenario is more realistic since the mooring at the port is a predefined procedure, taking place in specific time frame. In this case, EAT and LAT parameters of each intermediate port have different values, but again it is assumed that waiting is not allowed at intermediate ports. The computational results in Table 2 confirm the high performance of the proposed algorithm, compared to the common decomposition approach.

Although the anchorage duration is usually fixed, we investigate the possibility of extending that duration without additional cost. Specifically, we assume that the anchorage duration could be extended up to 30 minutes and hence $LDT_j = t^j_{ar} + 30$. Clearly, the results of the Table 3 show that the execution time is increased in both algorithms but still our algorithm exhibits the best performance.

Our algorithm can also handle the more general case where the visiting order of the intermediate stops is not predefined. In this scenario, the algorithm should select the most beneficial visiting order according to the objectives being optimized given the constraint on the total travel time. The common decomposition algorithm could not be applied in this case because despite the fact that the initial problem could be divided in sub-problems, the sub-problem order is unknown and due to time-dependency, synthesizing the sub-problems solutions is even more difficult. Table 4 lists the execution time of our algorithm for this problem variation and for different input instances. Interestingly, although the search space in this variation is much larger than in the case when there is a fixed sequence of intermediate ports, the execution time of our algorithm is comparable with that in the case of the predefined visiting order of intermediate stops.

Table 2 Travel planning with predefined visiting order of intermediate ports and time windows in these ports.

Case No	Start Time	Maximum Travel Duration (hours)	No of Intermediate Stops	CPU time (in seconds)	
				Decomposition Algorithm	Proposed Algorithm
1	10:00 am	4	1	77	49
2	1:00 pm	4.5	2	89	58
3	3:00 pm	5	3	105	70
4	3:30 pm	6	4	127	81
5	7:00 am	6.5	5	134	89
6	8:00 am	7.5	6	145	101
7	9:00 am	8.5	7	162	123
8	8:30 am	9.5	8	181	134
9	7:30 am	10	9	196	149
10	7:00 am	11	10	201	163

Table 3 Itinerary planning with predefined visiting order of intermediate ports and waiting allowed at these ports.

Case No	Start Time	Maximum Travel Duration (hours)	No of Intermediate Stops	CPU time (in seconds)	
				Decomposition Algorithm	Proposed Algorithm
1	10:00 am	4	1	95	61
2	1:00 pm	4.5	2	113	73
3	3:00 pm	5	3	129	85
4	3:30 pm	6	4	158	98
5	7:00 am	6.5	5	173	112
6	8:00 am	7.5	6	180	129
7	9:00 am	8.5	7	201	155
8	8:30 am	9.5	8	226	168
9	7:30 am	10	9	248	183
10	7:00 am	11	10	255	203

5 Conclusions

We have focused on the problem of finding all the Pareto optimal itinerary plans which passes through a fixed sequence of nodes, in a deterministic time-dependent setting considering two conflicting objectives and with a constraint on the total duration of the itinerary. We have proposed a general algorithm which can be applied to several application scenarios. The time-dependent network on which we search for the Pareto optimal solutions is a non-FIFO network. Thus, waiting at intermediate stops may be a valid option for achieving low-cost solutions. We considered also the case where there is a fixed schedule for visiting the intermediate stops and a constraint on the latest departure time from each intermediate stop. In order to evaluate the performance of the proposed algorithm, it was compared with the common decomposition approach in a case study relevant to the sea transportation. The optimization criteria in the experiments were the total fuel consumption and the total risk of the itinerary. The main conclusion from these experiments is that due to efficient early pruning of candidate solutions, our algorithm outperforms the common decomposition

Table 4 Itinerary planning with no predefined visiting order of the intermediate ports.

Case No	Start Time	Maximum Travel Duration (hours)	No of Intermediate Stops	CPU time (in seconds) Proposed Algorithm
1	10:00 am	4	2	61
2	1:00 pm	4.5	2	71
3	3:00 pm	5	3	87
4	3:30 pm	6	4	99
5	7:00 am	6.5	5	112
6	8:00 am	7.5	6	127
7	9:00 am	8.5	7	156
8	8:30 am	9.5	8	170
9	7:30 am	10	9	188
10	7:00 am	11	10	205

method in all tests. Moreover, our algorithm turns out to be applicable also in the case where the visiting order of the intermediate stops is not predefined. This generalized problem is more difficult than the original problem, because the set of efficient solutions is even larger. Finally, another interesting feature of the proposed algorithm is that it permits the change of the vessel speed between successive intermediate ports. A more general scenario, where the speed of a ship is allowed to change more frequently even from node to node in the input grid is not very interesting since the common practice is exactly the opposite, that is, the ship speed is usually maintained constant while at sea and when there is no emergency.

References

1 Konstantinos N Androutsopoulos and Konstantinos G Zografos. Solving the multi-criteria time-dependent routing and scheduling problem in a multimodal fixed scheduled network. *European Journal of Operational Research*, 192(1):18–28, 2009.

2 Kyriakos Avgouleas. *Optimal ship routing*. PhD thesis, Massachusetts Institute of Technology, 2008.

3 Jean-François Bérubé, Jean-Yves Potvin, and Jean Vaucher. Time-dependent shortest paths through a fixed sequence of nodes: application to a travel planning problem. *Computers & operations research*, 33(6):1838–1856, 2006.

4 Ismail Chabini. Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. *Transportation Research Record: Journal of the Transportation Research Board*, 1645(1):170–175, 1998.

5 Camila F Costa, Mario A Nascimento, José AF Macêdo, Yannis Theodoridis, Nikos Pelekis, and Javam Machado. Optimal time-dependent sequenced route queries in road networks. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 56. ACM, 2015.

6 Rafael Castro de Andrade. New formulations for the elementary shortest-path problem visiting a given set of nodes. *European Journal of Operational Research*, 254(3):755–768, 2016.

7 Theodoros Giannakopoulos, Ioannis A Vetsikas, Ioanna Koromila, Vangelis Karkaletsis, and Stavros Perantonis. Aminess: a platform for environmentally safe shipping. In *Proceedings of the 7th International Conference on PErvasive Technologies Related to Assistive Environments*, page 45. ACM, 2014.

8 Konstantinos G Gkonis and Harilaos N Psaraftis. Some key variables affecting liner shipping costs. *Laboratory for Maritime Transport, National Technical University of Athens*, 2010.

9 Jörn Hinnenthal and Günther Clauss. Robust pareto-optimum routing of ships utilising deterministic and ensemble weather forecasts. *Ships and Offshore Structures*, 5(2):105–114, 2010.

10 Lars Magnus Hvattum, Inge Norstad, Kjetil Fagerholt, and Gilbert Laporte. Analysis of an exact algorithm for the vessel speed optimization problem. *Networks*, 62(2):132–135, 2013.

11 MSC IMO. 1/circ. 1228. *Revised guidance to the master for avoiding dangerous situations in adverse weather and sea conditions, adopted 11th January*, 2007.

12 David E Kaufman and Robert L Smith. Fastest paths in time-dependent networks for intelligent vehicle-highway systems application. *Journal of Intelligent Transportation Systems*, 1(1):1–11, 1993.

13 Ioanna Koromila, Zoe Nivolianitou, and Theodoros Giannakopoulos. Bayesian network to predict environmental risk of a possible ship accident. In *Proceedings of the 7th International Conference on PErvasive Technologies Related to Assistive Environments*, page 44. ACM, 2014.

14 Feifei Li, Dihan Cheng, Marios Hadjieleftheriou, George Kollios, and Shang-Hua Teng. On trip planning queries in spatial databases. In *Advances in Spatial and Temporal Databases*, pages 273–290. Springer, 2005.

15 John J Liu. *Supply chain management and transport logistics*. Routledge, 2011.

16 G Mannarini, G Coppini, P Oddo, and N Pinardi. A prototype of ship routing decision support system for an operational oceanographic service. *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, 7(1):53–59, 2013.

17 Stéphane Marie, Eric Courteille, et al. Multi-objective optimization of motor vessel route. In *Proceedings of the Int. Symp. TransNav*, volume 9, pages 411–418, 2009.

18 Ariel Orda and Raphael Rom. Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. *Journal of the ACM (JACM)*, 37(3):607–625, 1990.

19 Ariel Orda and Raphael Rom. Minimum weight paths in time-dependent networks. *Networks*, 21(3):295–319, 1991.

20 David Ronen. The effect of oil price on the optimal speed of ships. *Journal of the Operational Research Society*, 33(11):1035–1040, 1982.

21 Mehdi Sharifzadeh, Mohammad Kolahdouzan, and Cyrus Shahabi. The optimal sequenced route query. *The VLDB journal*, 17(4):765–787, 2008.

22 J Szlapczynska and R Smierzchalski. Multicriteria optimisation in weather routing. In *Proceedings of the Int. Symp. TransNav*, volume 9, pages 423–429, 2009.

23 K Takashima, B Mezaoui, and R Shoji. On the fuel saving operation for coastal merchant ships using weather routing. In *Proceedings of Int. Symp. TransNav*, volume 9, pages 431–436, 2009.

24 Aphrodite Veneti, Charalampos Konstantopoulos, and Grammati Pantziou. Continuous and discrete time label setting algorithms for the time dependent bi-criteria shortest path problem. In *Operations Research and Computing: Algorithms and Software for Analytics*, pages 62–73, Virginia, January 11-13 2015. 14th INFORMS Computing Society Conference Richmond.