# Dynamic Journeying in Scheduled Networks

Lauri Häme, Harri Hakula and Saara Hyvönen

***Abstract*—We study a dynamic journey planning problem for multimodal transportation networks. The goal is to find a journey, possibly involving transfers between different transport modes, from a given origin to a given destination within a specified time horizon. Transport services are represented as sequences of scheduled legs between nodes in the transportation network. Due to uncertainty in transport services, we consider for each pair of adjacent legs $i, j$ the probability of a successful transfer from $i$ to $j$. If a transfer between two legs is unsuccessful, the customer needs to reconsider the remaining path to the destination. The problem is modeled as a Markov decision process and the main contribution is a backward induction algorithm that generates an optimal policy for traversing the public transport network in terms of a given objective, for example, reliability, ride time, waiting time, walking time or the number of transfers. A straightforward method for maximizing reliability is also suggested, and the algorithms are tested on real-life Helsinki area public transport data. Computational examples show that with a given input, the proposed algorithms rapidly solve the journeying problem.**

***Index Terms*—Initerary planning, multimodal transportation network, Markov decision process**

## I. Introduction

THE urban itinerary planning problem involves determining a path, possibly involving transfers between different transport modes, from a specified origin to a similarly specified destination in a transport network. Common criteria used for evaluating itineraries include the total duration, number of transfers and cost [1]–[18].

We present a dynamic model taking into account the uncertainty of transport services (buses, trams, trains, ferries, . . . ). In contrast to existing itinerary planning algorithms designed for scheduled public transport networks, where the path is a priori optimized with respect to an objective, for example, [1], we take into consideration the fact that the realized journey may differ from the original plan.

Real-time information on the status of transport services is available via mobile devices with location-based capabilities. This makes it possible for a commuter to dynamically modify the planned journey in case of a delay or cancellation. For example, if a transfer from a transport service to another is unsuccessful due to a delay, the commuter may reconsider the remaining path to the destination. Clearly, the importance of being able to modify the planned journey dynamically is emphasized when the number of transfers between different transport services is increased.

Taking into account the uncertainty is particularly important in difficult weather conditions when delays are common. In addition to traditional public transport with fixed schedules, uncertainty should be given special attention in flexible transport services without fixed routes [19].

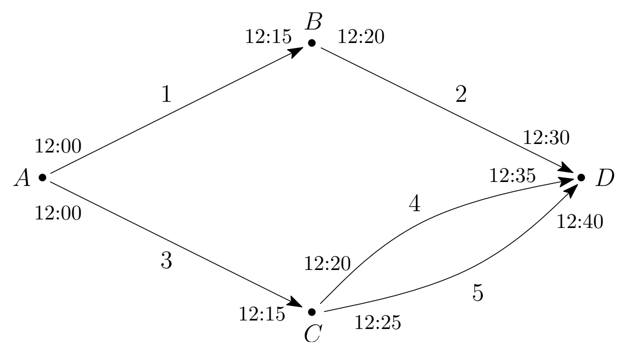


Fig. 1. The difference between dynamic and a priori journey planning for a commuter traveling from $A$ to $D$. The four points represent public transport stops $(A, B, C, D)$ and the arrows between them represent public transport services $(1, \ldots, 5)$. Initially, there are three possible journeys from $A$ to $D$: $(1, 2), (3, 4)$ and $(3, 5)$. If the commuter initially chooses service 1, the success of the journey is dependent of the success of the transfer from 1 to 2 at stop $B$. If the commuter chooses service 3 first, the destination is reached if one of the transfers $3 \to 4$ or $3 \to 5$ is successful at stop $C$.

A simplified example clarifying the main difference between dynamic and a priori journey planning is shown in Figure 1.

Generally, a commuter wishes to travel from an origin node $v_o$ to a destination node $v_d$ within a time horizon $[0, T]$ using different transport services. Each transport service is represented as a sequence of *legs*. Each leg is associated with a *start node* and *end node*, as well as a random *start time* and *end time*. Adjacent nodes in the network are connected with similarly defined walking legs.

A path from the origin to the destination is represented as a sequence of legs, in which the start node of each leg is equal to the end node of the previous leg. We assume that during the execution of a leg, the commuter receives information on which services have already visited the end node and which are yet to arrive. In other words, the customer "sees" the available successor legs of the current leg and may choose to (i) stay in the vehicle, (ii) transfer to another vehicle or (iii) get off the vehicle and start walking towards a nearby stop (or the destination). Our approach is to define an optimal policy specifying the actions that are executed in different situations in order to optimize reliability, ride time, waiting time, walking time, the number of transfers of a combination of these objectives.

Dynamic path finding problems (see [20]–[30]) are often modeled as *Markov decision processes* [31], [32], in which the *actions* of a decision maker at a given *state* are independent of all previous actions and states. In this paper we define the current state as the current leg. For a more detailed model

Lauri Häme and Harri Hakula are with the Department of Mathematics, Aalto University School of Science, P.O.Box 11000, FI-00076 AALTO, FINLAND (e-mail: Lauri.Hame@tkk.fi, Harri.Hakula@tkk.fi).

Saara Hyvönen is with Nokia Corp. (email: Saara.Hyvonen@nokia.com)

taking into account the travel history, we refer to [33].

This work is partially motivated by a demand-responsive transport (DRT) service currently being planned to operate in Helsinki. Helsinki Region Transport board has approved a plan under which the trial period of the service takes place from 2012 to 2014. Similarly as the current flexible service routes [34], the new DRT service is designed to operate on a demand-responsive basis, that is, vehicle routes are modified according to the demand situation. The main difference to existing services is that no pre-order times for trips are required and the trips can be booked "on the fly" by means of an interactive user interface. This type of new service calls for a journey planner that is capable of communicating with flexible services as well as traditional public transport, thus combining the benefits of both transport modes.

In addition to public transport, a journey planning problem with a similar objective function arises in freight transportation by for-hire carriers.

The remainder of this document is organized as follows: The multimodal transport network is modeled as an acyclic graph in Section II. Section III-A suggests a straightforward method for maximizing the reliability of a journey and an algorithm that generates an optimal policy for the Markov decision process is presented in Section III-B. The solution methods are evaluated by numerical examples in Section IV.

## II. MODEL

Let $\mathcal{V}$ denote a set of *nodes* representing public transport stops in a specific area and let $\mathcal{K} \subset \mathbb{N}$ denote a set of public transport *services* operating in this area, indexed by natural numbers.

Each service $k \in \mathcal{K}$ follows a *route*, represented as a sequence of nodes $(v_1^k, \dots, v_m^k)$ in $\mathcal{V}$. Note that it is possible for a service to visit the same node more than once. For example, if node $v_i^k$ is included in the route twice, there exists an index $j \neq i$ for which $v_i^k = v_j^k$. Each service departs at node $v_1^k$ at a specific time and proceeds to nodes $v_2^k, \dots, v_m^k$ in the order determined by the route. The *expected arrival time* of service $k$ at node $v_j^k$ is denoted by $t_j^k$. Thus, a service $k$ can be represented as a sequence of nodes and expected arrival times $((v_1^k, t_1^k), \dots, (v_m^k, t_m^k))$, see Figure 2.

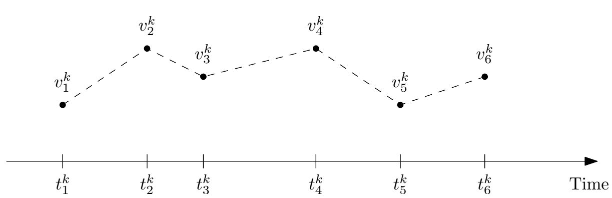


Fig. 2. The route and schedule of a transport service. The points represent the nodes $v_j^k$ that define the route of service $k$ and the real numbers $t_j^k$ on the timeline represent the expected arrival times of the service at the nodes.

### A. Service legs

Each service $k \in \mathcal{K}$ can be decomposed as a set of scheduled *legs* between subsequent stops. That is, each leg has a start node, end node, expected start time and expected end time. By this decomposition, we can represent any path in the transport network as a sequence of legs indexed by natural numbers. For example, consider a transport network consisting of ten legs numbered from 1 to 10. Then, the sequences $(5, 2, 7)$ and $(5, 8, 3, 7)$ define two paths in the network involving two and three *transfers*, respectively. Each transfer between two legs has a specific probability of success. Transfers between subsequent legs of the same service are assumed to be successful with probability 1.

Formally, letting $\mathcal{L} \subset \mathbb{N}$ denote the set of indices of legs, a leg corresponding to index $i \in \mathcal{L}$ is defined by a pair of node-arrival time pairs and a service number $((v_i, t_i), (v_i', t_i'), k_i)$, where $v_i$ is the *start node*, $v_i'$ is the *end node*, $t_i$ is the *expected start time*, $t_i'$ is the *expected end time* and the *service number* $k_i$ is the index of the service that executes leg $i$. The *expected ride time* of service leg $i$ is defined by $r_i = t_i' - t_i$.

By using this definition, a service $((v_1^k, t_1^k), \dots, (v_m^k, t_m^k))$ is represented by a set of legs $\{i_1^k, \dots, i_{m-1}^k\} \subset \mathcal{L}$, where leg $i_h^k$ is determined by

$$\left((v_{i_h^k}, t_{i_h^k}), (v'_{i_h^k}, t'_{i_h^k}), k_{i_h^k}\right) = \left((v_h^k, t_h^k), (v_{h+1}^k, t_{h+1}^k), k\right) \quad (1)$$

for $h \in \{1, \dots, m-1\}$. Note that the expected end time $t'_{i_h^k}$ of leg $i_h^k$ refers to the same variable as the expected start time $t_{i_{h+1}^k}$ of leg $i_{h+1}^k$, that is, $t'_{i_h^k} = t_{i_{h+1}^k} = t_{h+1}^k$. To keep track which legs are subsequent legs of the same service, we define for each leg $i_h^k$ the *set of immediate successors* $I_{i_h^k} = \{i_{h+1}^k\}$ for $h \in \{1, \dots, m-1\}$.

### B. Walking legs

After each service leg, the commuter may choose to continue the journey by foot. Similarly as the service legs defined above, each *walking leg* $i \in \mathcal{L}$ is determined by a start node $v_i$, end node $v_i'$, expected start time $t_i$, expected end time $t_i'$ and service number $k_i$. For all walking legs, we choose $k_i < 0$ in order to have a distinction between walking legs and service legs.

Walking legs are added by associating with each service leg $((v_i, t_i), (v_i', t_i'), k_i)$ a set $L_i \subset \mathcal{L}$ of walking legs beginning at $v_i'$ and ending at a stop within a specific *maximum walking distance* $d_w^{\max}$ from $v_i'$, see Figure 3a. Such walking legs $j \in L_i$ are of the form $\left((v_j, t_j), (v_j', t_j'), k_j\right) = \left((v_i', t_i'), (v_j', t_j'), -k_i\right)$, where the walking distance $d_w$ satisfies $d_w(v_i', v_j') \leq d_w^{\max}$.

The *expected walking time* of walking leg $i$ is defined by $l_i = t_i' - t_i$. The expected ride times of walking legs and the expected walking times of service legs are equal to zero.

For each service leg $i$, the walking legs are added to the set of immediate successors $I_i$ of $i$, that is, $I_i \leftarrow I_i \cup L_i$.

Sequential walking legs are added to the network iteratively: First, walking legs (indexed with service number $-k_i$) are added to the end of each each service leg $i$ (step 1). Then, walking legs (indexed with service number $k_j$) are added to the end of each walking leg $j$ (step 2). Step 2 is repeated until the desired amount of walking legs is obtained. The number of walking legs included in the model is thus controlled by two parameters: The maximum walking distance $d_w^{\max}$ and the maximum number of sequential walking legs.

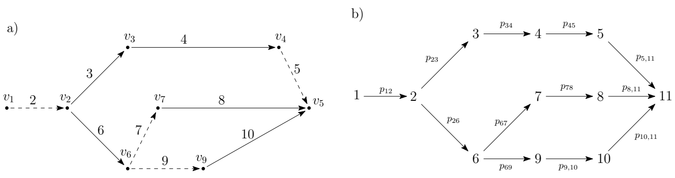


Fig. 3. a) A sample transport network with eight stops and five transport services consisting of a single leg (solid arrows $3, 4, 6, 8, 10$). Adjacent stops are connected with walking legs (dashed arrows $2, 5, 7, 9$). b) A directed graph representing the relations of legs in Figure 3a. The origin and destination are represented by legs 1 and 11. The transfer probability from leg $i$ to leg $j$ is denoted by $p_{ij}$. Note that $p_{12} = p_{45} = p_{67} = p_{69} = 1$, since $2, 5, 7$ and $9$ are walking legs.

### C. Transfers

A transfer from leg $i$ to leg $j$ is possible only if leg $j$ begins at the node from which $i$ ends, that is, $v'_i = v_j$ and $t'_i \leq t_j$ (see Figure 3b). Letting $\mathcal{L}$ denote the set of legs and $S_i = \{j \in \mathcal{L} \mid v'_i = v_j, t'_i \leq t_j\}$ denote the *successor set* of leg $i$, the *transfer probability* from leg $i$ to leg $j$ is defined as a real number $p_{ij}$ satisfying

$$p_{ij} \begin{cases} = 1, & \text{if } j \in I_i, \\ \in [0, 1], & \text{if } j \in S_i \setminus I_i, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Clearly, the legs and transfer probabilities form a directed acyclic graph, as in [31], see Figure 3b. We assume that the transfers between legs are independent events.

The *expected waiting time* of a transfer from leg $i$ to $j$ is defined by $w_{ij} = t_j - t'_i$. Note that if $j$ is an immediate successor of $i$, we have $w_{ij} = 0$.

### D. Paths

A path can be represented as a sequence of legs $(i_1, \ldots, i_m)$ satisfying $i_{h+1} \in S_{i_h}$ for all $h \in \{1, \ldots, m-1\}$, see Figure 4. The path is *successful* with probability $\prod_{h=1}^{m-1} p_{i_h i_{h+1}}$ and the *subjective price* of the path is defined by

$$\begin{aligned} &C\left((i_1, i_2, \ldots, i_m)\right) \\ &= \sum_{h=1}^{m-1} (a_1 w_{i_h i_{h+1}} + a_2 \chi_{i_h i_{h+1}}) + \sum_{h=1}^{m} (a_3 r_{i_h} + a_4 l_{i_h}), \end{aligned} \tag{3}$$

where $w_{ij}$ is the expected waiting time of a transfer from $i$ to $j$, $\chi_{ij}$ is a transfer variable defined by $\chi_{ij} = 0$ if $j \in I_i$ and $\chi_{ij} = 1$ otherwise, $r_i$ is the expected ride time and $l_i$ is the expected walking time of leg $i$. The parameters $a_i, i \in \{1, \ldots, 4\}$ describe the commuter's valuation for unit waiting time, a transfer between services, unit ride time and unit walking time, respectively.

### E. Time horizon

Let $[0, T] \subset \mathbb{R}$ be the time horizon of the problem, let $v_1$ denote the origin node and $v_d$ denote the destination node. The *origin leg* is defined by $((v_1, 0), (v_1, 0), 0)$, and the *destination leg* is defined by $((v_d, T), (v_d, T), 0)$. By the above definitions, we can represent the entire journey as legs, including the origin and the destination.

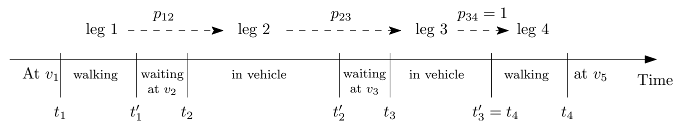


Fig. 4. Representation of a path. The ticks on the time axis denote the schedule of a path from $v_1$ to $v_5$. The path is represented by four legs: (1) A commuter starts walking from the origin $v_1$, and arrives at stop $v_2$. (2) A transport service departs at $v_2$ and travels to stop $v_3$. (3) A transport service departs at $v_3$ and arrives at stop $v_4$. (4) The commuter continues by foot to the destination $v_5$. Note that the path is successful with probability $p_{12}p_{23}$.

With no loss of generality, all legs for which the expected start or end time is outside the time horizon $[0, T]$ and all legs for which the start node is the destination node (except the destination leg) are excluded from the problem. The cropped set of legs is defined by

$$\begin{aligned} \mathcal{L}_{[0,T]} = \{i \in \mathcal{L} \mid 0 \leq t_i \leq T \text{ and } 0 \leq t'_i \leq T\} \\ \setminus \{i \in \mathcal{L} \setminus \{d\} \mid v_i = v_d\}. \end{aligned}$$

For clarity, we define $n := |\mathcal{L}_{[0,T]}|$ and the legs in $\mathcal{L}_{[0,T]}$ are numbered from 1 to $n$, where the origin leg is indexed by 1 and the destination leg is indexed by $n$.

## III. Problem solution

In the following, we present two alternative approaches to the problem. In the first approach (Section III-A), we present a straightforward method to maximize the reliabilty of a journey. In the second approach (Section III-B), we consider a more general objective function by modeling the problem as a Markov decision process.

### A. Expected number of paths

Let us present a method for finding the most reliable journey from the origin leg 1 to the destination leg $n$. Our approach is to maximize the *expected number of successful paths* to the destination leg from each leg of the journey. The approach is motivated by the idea that paths that allow several detours are considered more reliable than paths with no alternatives (see Figure 1).

Let $h_i$ denote the expected number of successful paths from leg $i \in \{1, \ldots, n-1\}$ to the destination leg $n$. When the commuter is at leg $i$, the available transfer options $X$ to successor legs are revealed. The commuter transfers to the leg $i' \in X$ for which $h_{i'}$ is maximized.

Theorem 1 establishes a relation between eigenvectors and the expected number of successful paths. For this purpose, we define a *sink graph* as follows.

*Definition 1:* Let $G = (V, A)$ be a weighted directed acyclic graph, where a weight $p_{ij} \in [0, 1]$ is assigned to each arc $(i, j) \in A$ and let $s \in V$ be a node such that $(s, i) \notin A$ for all $i \in V$. The graph $G_s = (V, A \cup (s, s))$, where $p_{ss} = 1$, is called a *sink graph*.

In other words, a sink graph is a directed acyclic graph $(V, A)$ with the exception that one node $s \in V$ with zero outdegree is associated with a loop $(s, s)$.

*Theorem 1:* Let $P$ denote the adjacency matrix of a sink graph $G_s = (V, A)$, where $V = \{1, \ldots, |V|\}$, let $h_i$ denote the expected number of succesful paths from $i$ to $s$ for $i \in V \setminus \{s\}$ and let $h_s = 1$. Then, $h = (h_1, \ldots, h_{|V|})^T$ is a unique dominant eigenvector of $L$.

*Proof:* Since the adjacency matrix of a directed acyclic graph is an upper triangular matrix with zeros on the diagonal, the adjacency matrix $P$ of a sink graph is an upper triangular matrix with diagonal elements $p_{ii} = 0$ except for the sink node $s$, for which we have $p_{ss} = 1$. The eigenvalues of an upper triangular matrix are equal to the diagonal elements [35] and thus there exists a unique dominant eigenvalue 1. Let us show that $h = (h_1, \ldots, h_{|V|})^T$ is the corresponding eigenvector of $P$.

Since all paths in a directed acyclic graph are acyclic and by definition we have $h_s = 1$, the expected number of paths $h_i$ from node $i$ to the sink node $s$ in the sink graph $G_s$ satisfies $h_i = \sum_{j \in V} p_{ij} h_j$ for $i \in V \setminus \{s\}$ and $h_s$ satisfies $h_s = 1 = p_{ss} h_s = \sum_{j \in V} p_{sj} h_j$. In matrix form, we have $h = Ph$ and thus $h$ is the unique eigenvector corresponding to eigenvalue 1. ■

Theorem 1 states that by constructing a sink graph, the expected number of successful paths $h_i$ from leg $i$ to the destination leg $n$ is given by the dominant eigenvector of the adjacency matrix $P$ consisting of the transfer probabilities $p_{ij}$ between legs (and $p_{nn} = 1$).

The expected number of successful paths can also be calculated for all legs that are reachable from the origin leg by means of Algorithm 1. The complexity of the algorithm is

**if** $h_i \geq 0$ *($h_i$ has already been determined )* **then**
  Return $h_i$;
**end**
**forall** $i' \in S_i$ *such that* $p_{ij} > 0$ *($S_i$ = successor set of $i$)* **do**
  $h_{i'} \leftarrow \mathrm{Ex}(i')$;
**end**
$h_i \leftarrow \sum_{j \in S_i} p_{ij} h_j$;
Return $h_i$;

**Algorithm 1**: A recursive function $\mathrm{Ex}(i)$ for calculating the expected number of feasible paths $h_i$ from leg $i$ to the destination leg $n$. Initially, set $h_i = -1$ for all $i \in \{1, \ldots, n-1\}$ and $h_n \leftarrow 1$.

of order $\mathcal{O}(n + |A|)$, where $A$ is the set of transfers for which $p_{ij} > 0$. Letting $m = \max i \in \{1, \ldots, n-1\} |S_i|$ denote size of the largest successor set, the complexity is bounded above by $\mathcal{O}(n(m+1))$.

### B. Markov Decision Process

In order to be able to handle a general objective function, we propose a finite-state Markov decision process $(S, A, P_\cdot(\cdot,\cdot), R_\cdot(\cdot,\cdot))$, where the parameters are defined as follows.

*1) States = Legs:* The set of *states* $S$ is equal to the set of legs, that is, $S = \mathcal{L}_{[0,T]} = \{1, \ldots, n\}$. State 1 is referred to as the *origin state* and state $n$ is referred to as the *destination state*. Since legs beginning from the destination node are not included in the problem, we have $S_n = I_n = \emptyset$.

*2) Actions:* The set of *actions* $A$ consists of sets $A_s$ of actions available at states $s \in S$. An action $a \in A_s$ is defined as a *preference order* of the successor states $s' \in S_s$, that is, a bijection $a : S_s \to \{1, \ldots, |S_s|\}$, where $a(s')$ denotes the ranking of the successor state $s' \in S_s$ in the preference order. The successor states of $s$ ranked by the preference order $a$ are denoted by $s^{a,a(s')}$. Given the the sorted successor states $s^{a,1}, \ldots, s^{a,|S_s|} \in S_s$ of $s$, the commuter transfers to state $s^{a,k}$ if (i) the transfer to $s^{a,g}$ is unsuccessful for $1 \leq g < k$ and (ii) the transfer to $s^{a,k}$ is successful.

*3) Transition probabilities:* $P_a(s, s')$ is the probability that action $a$ in state $s$ at step $t$ will lead to state $s' \in S_s$ at step $t+1$. Given the preference order $s^{a,1}, \ldots, s^{a,|S_s|} \in S_s$ defined by action $a \in A_s$, since successful transfers are assumed to be independent events, we have

$$P_a(s, s^{a,k}) = \sum_{k=1}^{h} p_{ss^{a,k}} \prod_{g=1}^{k-1} (1 - p_{ss^{a,g}}), \tag{4}$$

where $p_{ss^{a,k}}$ denotes the transfer probability from $s$ to $s^{a,k}$, as defined in Equation (2). For example, let $s^{a,1}, s^{a,2}, s^{a,3}$ denote three successor states of state $s$, sorted in the preference order defined by action $a$. If $s^{a,2}$ is an immediate successor of $s$, that is, $s^{a,2} \in I_s$, and $s^{a,1} \notin I_s$, we have $P_a(s, s^{a,1}) = p_{ss^{a,1}}$, $P_a(s, s^{a,2}) = 1 - p_{ss^{a,1}}$ and $P_a(s, s^{a,3}) = 0$.

*4) Rewards:* $R_a(s, s')$ is the expected immediate reward received after transition from state $s \in S$ to state $s' \in S$ with transition probability $P_a(s, s')$.

The expected profit of a path $(1, s_1, \ldots, s_m, n)$ from the origin state 1 to the destination state $n$ is given by

$$\pi\left((s_0, \ldots, s_m)\right) = p_{s_m n} u - C\left((s_0, \ldots, s_m)\right), \tag{5}$$

where $u$ is the utility of arriving at the destination before the deadline $T$ (or at $T$) and $C\left((s_0, s_1, \ldots, s_{m-1}, s_m)\right)$ is the subjective price of the path defined by (3).

Since the objective is to maximize (5), we define $R_a(s, s')$ by

$$R_a(s, s') = \begin{cases} p_{ss'} u, & \text{if } s' = n \ , \\ -a_1 w_{ss'} - a_2 \chi_{ss'} - a_3 r_{s'} - a_4 l_{s'}, & \text{otherwise.} \end{cases} \tag{6}$$

Note that $R_a(s, s')$ is independent of the action $a$. Thus, for the remainder of this document, we will use the notation $R(s, s') = R_a(s, s')$.

Clearly, the above model can be used to maximize the probability of arriving at the destination in time by defining $u = 1$ and $a_1 = a_2 = a_3 = a_4 = 0$.

### C. Problem solution

The solutions to Markov decision processes are characterized as policies, that is, functions $\pi$ that specify the action $a(s)$ that the commuter chooses when in state $s$. The goal is to find a policy $\pi$ that maximizes the expected reward. Generally, the calculation of an optimal policy requires two arrays indexed by state: value $V$, which contains real values, and policy $\pi$ which contains actions. In our problem, $V(s)$ corresponds to the expected profit to be earned by following a policy that maximizes the expected profit from $s$ onwards.

Similarly as in [36], the value $V(s)$ is defined by

$$V(s) := \max_{a \in A_s} \left\{ \sum_{s' \in S_s} P_a(s, s') \left(R(s, s') + V(s')\right) \right\} \tag{7}$$

for all $s \in S$. Note that $V(n) = 0$ since $S_n = \emptyset$.

An optimal policy is characterized as follows: When at state $s$, the available transfer options $X \subset S_s$ to successor states are revealed. The commuter transfers to a state $s' \in X$ for which $R(s, s') + V(s')$ is maximized. Thus, an optimal action $a$ at state $s$ is defined by

$$R(s, s^{a,1}) + V(s^{a,1}) \geq \ldots \geq R(s, s^{a,|S_s|}) + V(s^{a,|S_s|}), \quad (8)$$

where the successor states ranked by action $a$ are denoted by $s^{a,a(s')}$ for all $s' \in S_s$. Equation (8) gives an optimal action for a state $s$, given that the values $V(s')$ of its successor states are known. In the following, we present an algorithm for calculating the values for all states.

### D. Backward induction algorithm

The values $V(s)$ of states can be determined by means of backward induction, as shown in Algorithm 2. By executing Rec(1), the program recursively calculates values and optimal actions for all states $s$ that are reachable from the origin state 1. Initially, we only know the values of the destination state, that is, $V(n) = 0$. Thus, the first states for which the value can be calculated are the ones that precede a destination state. The algorithm then proceeds backwards until the value $V(1)$ of the origin state is calculated. Since the algorithm involves sorting,

**if** $V_H(s) \neq \emptyset$ *($V_H(s)$ has already been determined )* **then**
 Return $V_H(s)$;
**end**
**forall** $s' \in S_s$ *(successor set of s)* **do**
 $V_H(s') \leftarrow \text{Rec}(s')$;
**end**
Determine an optimal action $a$ by sorting the states $s' \in S_s$ in descending order of $R(s, s') + V_H(s')$;
$V_H(s) \leftarrow \sum_{k=1}^{|S_s|} P_a(s, s^{a,k}) \left( R(s, s^{a,k}) + V_H(s^{a,k}) \right)$;
Return $V_H(s)$;

**Algorithm 2**: A recursive function Rec($s$) for calculating an optimal policy. Initially, set $V_H(s) \leftarrow \emptyset$ for all $s \in \{1, \ldots, n-1\}$ and $V(n) \leftarrow 0$.

the complexity is bounded above by $\mathcal{O}(nm \log m + nm) = \mathcal{O}(nm(1 + \log m))$, where $m = \max_{s \in \{1,\ldots,n-1\}} |S_s|$ is the size of the largest successor set.

## IV. Numerical experiments

In the following, we present computational results obtained by Algorithms 2 and 1. The transfer probabilities between legs are determined by assuming gamma distributed ride times similarly as in [37], [38]. More precisely, given the expected ride time $t_i$ of leg $i$, we define the ride time $\tau_i$ as a random variable $\tau_i \sim \text{Gamma}(\alpha t_i, \beta, \delta t_i)$, where the Gamma$(\alpha, \beta, \delta)$ distribution is defined by the probability density function

$$f(x) = \frac{(x - \delta)^{\alpha - 1} e^{-(x-\delta)/\beta}}{\beta^\alpha \Gamma(\alpha)} \qquad \text{for } x > \delta \geq 0. \qquad (9)$$

The mean of a Gamma$(\alpha, \beta, \delta)$ distribution is $\alpha\beta + \delta$, that is, by choosing $\alpha, \beta$ and $\delta$ such that $\alpha\beta + \delta = 1$, we have $E(\tau_i) = t_i$.

We assume that at the beginning of the time horizon at $t = 0$, the locations of all vehicles are known. Thus, the schedules $((v_1^k, t_1^k), \ldots, (v_p^k, t_p^k))$ of services are converted into random schedules $((v_1^k, \tau_1^k), \ldots, (v_p^k, \tau_p^k))$ by $\tau_i^k \sim \text{Gamma}(\alpha t_i^k, \beta, \delta t_i^k)$ for $i \in \{1, \ldots, p\}$. The transfer probability from leg $i$ to $j$ is given by $P(\tau_i' \leq \tau_j)$.

The values of the $\alpha, \beta$ and $\delta$ parameters are $(1, 0.25, 0.75)$, which corresponds to the coefficient of variation $\sigma/\sqrt{\mu} = 0.25$ [37].

The algorithm was implemented in Mathematica and the tests were performed on a 2.2 GHz Dual Core Intel PC.

### A. Description of instances

The Helsinki tram network consists of ten tram lines, operated by 50 trams, and 154 stops. The tram schedules of Helsinki Region Transport available at [39] and the instances are available at `http://math.tkk.fi/~lehame/dju`.

In each instance, the origin and destination nodes $v_o$ and $v_d$ are chosen randomly from the set of 154 tram stops, see Figure 5. The departure time is set equal to 9:00 and the length of the time horizon is defined by $T = 1.6d(v_o, v_d)$, where $d(v_o, v_d)$ is the expected duration of the shortest path from $v_o$ to $v_d$ in the tram network. The maximum walking distance equals $d_w^{\max} = 0.25$, that is, all pairs of nodes, the distance between which is less than 0.25 km, are connected by walking legs. The speed of each walking leg equals $\nu_w = 5$ km/h and the duration of walking legs is modeled similarly as the duration of service legs, see Equation (9).

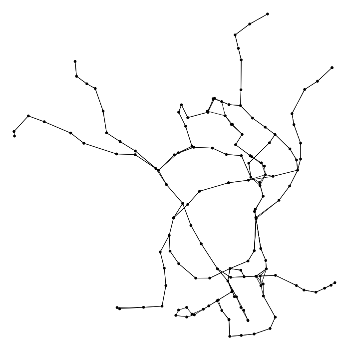

Fig. 5. The tram network of Helsinki consisting of ten tram lines and 154 stops.

### B. Results

TABLE I
COMPUTATIONAL RESULTS. THE TABLE SHOWS THE PROBABILITY $V(1)$ OF REACHING THE DESTINATION LEG $n$ FROM THE ORIGIN LEG 1, THE EXPECTED NUMBER OF SUCCESSFUL PATHS $h_1$ FROM 1 TO $n$ AND THE CORRESPONDING COMPUTATION TIMES IN SECONDS.

| Instance | Probability | | Exp. number of paths | |
|---|---|---|---|---|
| | CPU time (s) | $V(1)$ | CPU time (s) | $h_1$ |
| b1 | 0.23 | 0.97 | 0.15 | 106.88 |
| b2 | 0.06 | 1. | 0.04 | 42.31 |
| b3 | 0.06 | 1. | 0.04 | 20.72 |
| b4 | 0.06 | 0.46 | 0.04 | 1.25 |
| b5 | 0.31 | 0.97 | 0.22 | 7.05 |
| b6 | 0.1 | 0.91 | 0.07 | 2.67 |
| b7 | 0.09 | 0.85 | 0.06 | 12.72 |
| b8 | 0.05 | 0.7 | 0.03 | 0.7 |
| b9 | 0.12 | 0.99 | 0.08 | 25.67 |
| b10 | 0.08 | 0.79 | 0.06 | 83.34 |

The computational results of ten instances are shown in Table I. The first column shows the name of the instance. The third and fifth columns show the probability $V(1)$ of reaching the destination and the expected number of successful paths $h_1$. The second and fourth columns show the corresponding computation times.

By looking at the computation times in the Table I, we see that all instances were solved within a fraction of a second. In addition, calculating the expected number of successful paths is slightly faster than calculating probability, which involves sorting.

Since the expected number of successful paths represents the number of alternatives, even a large value of $h_1$ does not guarantee that the destination will be reached in time (b10). On the other hand, small values of $h_1$ seem to correspond to small probabilities (b4 and b8). For instances b2 and b3 with $V(1) = 1$, the expected number of successful paths gives additional information: Instance b2 provides more alternative routes for the commuter than b3.

## V. Conclusions

We consider a dynamic journey planning problem in scheduled transportation networks. The problem is modeled as a Markov decision process. We present an algorithm for producing an optimal policy for traveling from a given origin to a given destination, assuming that the remaining path to the destination can be modified at any time during the journey. We also provide a straightforward method for maximizing the reliability of a journey. Numerical results suggest that the proposed methods may be useful for practical purposes due to their computational simplicity.

## References


[1] K. N. Androutsopoulos and K. G. Zografos, "Solving the multi-criteria time-dependent routing and scheduling problem in a multimodal fixed scheduled network," *European Journal of Operational Research*, vol. 192, pp. 18–28, 2009.

[2] K. G. Zografos and K. N. Androutsopoulos, "Algorithms for itinerary planning in multimodal transportation networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, 2008.

[3] Z. R. Peng and R. Huang, "Design and development of interactive trip planning for web-based transit information systems," *Transp. Res. Part C*, vol. 8, no. 1, pp. 409–425, 2000.

[4] P. Modesti and A. Siomachen, "A utility measure for finding multiobjective shortest paths in urban multimodal transportation networks," *Eur. J. Oper. Res.*, vol. 111, no. 3, pp. 495–508, 1998.

[5] R. Huang and Z. R. Peng, "An integration of network data model and routing algorithms for online transit trip planning," in *Proc. 80th Annu. Meeting Transp. Res. Board*. Washington, DC, 2001.

[6] ——, "Schedule-based path finding algorithms for online transit trip planning," in *Proc. 81st Annu. Meeting Transp. Res. Board*. Washington, DC, 2002.

[7] M. Horn, "An extended model and procedural framework for planning multi-modal passenger journeys," *Transp. Res. Part B*, vol. 37, no. 7, pp. 641–660, 2003.

[8] C. O. Tong and A. J. Richardson, "A computer model for finding the timedependent minimum path in transit systems with fixed schedules," *J. Adv. Transp.*, vol. 18, no. 2, pp. 145–161, 1984.

[9] C. O. Tong and S. C. Wong, "A stochastic transit assignment model using a dynamic schedule-based network," *Transp. Res. Part B*, vol. 33, no. 2, pp. 107–121, 1999.

[10] A. Ziliaskopoulos and W. Wardell, "An intermodal optimum path algorithm for multimodal networks with dynamic arc travel times and switching delays," *Eur. J. Oper. Res.*, vol. 125, no. 3, pp. 486–502, 2000.

[11] J. Brub, J. Potvin, and J. Vaucher, "Time-dependent shortest paths through fixed sequence of nodes: Application to a travel planning problem," *Comput. Oper. Res.*, vol. 33, no. 6, pp. 1838–1856, 2006.

[12] K. L. Cooke and E. Halsey, "The shortest route through a network with time-dependent intermodal transit times," *J. Math. Anal. Appl.*, vol. 14, pp. 493–498, 1966.

[13] X. Cai, T. Kloks, and C. K. Wong, "Time-varying shortest path problems with constraints," *Networks*, vol. 29, no. 3, pp. 141–149, 1997.

[14] I. Chabini, "Discrete dynamic shortest path problems in transportation applications," *Transp. Res. Rec.*, vol. 1645, pp. 170–175, 1998.

[15] M. Kostreva and M. Wiecek, "Time dependency in multiple objective dynamic programming," *J. Math. Anal. Appl.*, vol. 173, no. 1, pp. 289–307, 1993.

[16] H. Hamacher, S. Ruzika, and S. Tjandra, "Algorithms for time-dependent bicriteria shortest path problems," *Discrete Optim.*, vol. 3, no. 3, pp. 238–254, 2006.

[17] J. Bander and C. C. White, "A new route optimization algorithm for rapid decision support," *Proc. Vehicle Navigat. Inf. Syst. Conf. Part 2*, vol. P-253, pp. 709–728, 1991.

[18] M. C. Tan, C. O. Tong, S. C. Wong, and J. Xu, "An algorithm for finding reasonable paths in transit networks," *J. Adv. Transp.*, vol. 41, no. 3, pp. 285–305, 2007.

[19] C. Mulley and J. D. Nelson, *Research in Transportation Economics*, vol. 25, no. 1, pp. 39 – 45, 2009.

[20] R. W. Hall, "The fastest path through a network with random time-dependent travel times," *Transportation science*, vol. 20, no. 3, pp. 182–188, 1986.

[21] J. L. Bander and C. C. White III, "A heuristic search approach for a nonstationary shortest path problem with terminal costs," *Transportation Science*, vol. 36, pp. 218–230, 2002.

[22] L. Fu and L. Rillet, "Expected shortest paths in dynamic and stochastic traffic networks," *Transportation Research*, vol. 32, no. 7, pp. 499–516, 1998.

[23] E. D. Miller-Hooks and H. S. Mahmassani, "Least expected time paths in stochastic, time-varying transportation networks," *Transportation Science*, vol. 34, pp. 198–215, 2000.

[24] C. Davies and P. Lingras, "Genetic algorithms for rerouting shortest paths in dynamic and stochastic networks," *European Journal of Operational Research*, vol. 144, pp. 27–38, 2003.

[25] S. Kim, M. Lewis, and C. White III, "Optimal vehicle routing with real-time traffic information," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 2, pp. 178–188, 2005.

[26] S. Kim, M. Lewis, and C. C. White III, "State space reduction for nonstationary stochastic shortest path problems with real-time traffic congestion information," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 3, pp. 273–284, 2005.

[27] A. Azaron and F. Kianfar, "Dynamic shortest path in stochastic dynamic networks: Ship routing problem," *European Journal of Operational Research*, vol. 144, pp. 138–156, 2003.

[28] M. C. Ferris and A. Ruszczynski, "Robust path choice in networks with failures," *Networks*, vol. 35, pp. 181–194, 2000.

[29] B. Thomas and C. White, "The dynamic shortest path problem with anticipation," *European Journal of Operational Research*, vol. 176, no. 2, pp. 836–854, 2007.

[30] S. Waller and A. Ziliaskopoulos, "On the online shortest path problem with limited arc cost dependencies," *Networks*, vol. 40, no. 4, pp. 216–227, 2002.

[31] H. E. Psaraftis and J. N. Tsitsiklis, "Dynamic shortest paths in acyclic networks with markovian arc costs," *Operations Research*, vol. 41, no. 1, pp. 91–101, 1993.

[32] G. H. Polychronopoulos and J. N. Tsitsiklis, "Stochastic shortest path problems with recourse," *Networks*, vol. 27, pp. 133–143, 1996.

[33] L. Häme and H. Hakula, "Dynamic journeying under uncertainty," 2012, manuscript submitted to EJOR.

[34] Helsinki Region Traffic. (2010, Oct.) Jouko neighbourhood routes and service routes. [Online]. Available: http://www.hsl.fi/EN/passengersguide/ServiceroutesandJoukoroutes/

[35] S. Axler, *Linear Algebra Done Right*. Springer-Verlag, ISBN 0-387-98258-2, 1996.

[36] R. Bellman, "A markovian decision process," *Journal of Mathematics and Mechanics*, no. 6, 1957.

[37] R. A. Russell and T. L. Urban, "Vehicle routing with soft time windows and erlang travel times," *Journal of the Operational Research Society*, vol. 59, pp. 1220–1228, 2007.

[38] Y. S. Chiang, , and P. O. Roberts, "A note on transit time and reliability for regular-route trucking," *Transport Research Part B*, vol. 14s, no. 1-2, pp. 59–65, 1980.

[39] Helsinki Region Traffic. (2010, Oct) Kalkati.net XML representation of helsinki regional transport authority (HSL) timetable data. [Online]. Available: http://developer.reittiopas.fi/