A polynomial algorithm for minimizing travel time in consistent time-dependent networks with waits

Jérémy Omer, Michael Poss

A polynomial algorithm for minimizing travel time in consistent time-dependent networks with waits

Jérémy Omer*

Univ Rennes, INSA Rennes, CNRS, IRMAR - UMR 6625, F-35000 Rennes, France

Michael Poss†

LIRMM, University of Montpellier, CNRS, Montpellier, France

September 1, 2020

Abstract

We consider a time-dependent shortest path problem with possible waiting at some nodes of the graph and a global bound W on the total waiting time. The goal is to minimize the time traveled along the edges of the path, not including the waiting time. We prove that the problem can be solved in polynomial time when the travel time functions are piecewise linear and continuous. The algorithm relies on a recurrence relation characterized by a bound ω on the total waiting time, where $0 \leq \omega \leq W$. We show that only a small number of values $\omega_1, \omega_2, \ldots, \omega_K$ need to be considered, where K depends on the total number of breakpoints of all travel time functions.

keywords: Time-dependent networks; Shortest paths; Label-setting algorithms; Concavity; Breakpoint; Wait.

1 Introduction

We consider a variant of the time-dependent shortest-path problem (TDSP). Let $G = (V, A)$ be directed graph with n nodes and m arcs. The travel time of each arc is a continuous piecewise linear function of the departure time from its tail, which satisfies the *consistency assumption*, preventing from reaching the head of an arc earlier by departing later from its tail. The objective is to find a path from $o \in V$ to $d \in V$ that has the lowest travel time. We additionally consider that waiting is allowed at nodes belonging to the set $U \subseteq V$, but the total waiting time may not exceed a given bound W. Finally, the waiting time is not considered in the cost so only the travel time needs to be minimized.

The problem has diverse applications in transportation, starting with the limitation of emissions in urban environments, given that the emissions of a vehicle are roughly proportional to its travel time, see for instance [13] for details. It can also be used for planning salesman trips, avoiding peak hours by stopping at specific places to perform other activities, therefore avoiding to loose time

*jeremy.omer@insa-rennes.fr
†michael.poss@lirmm.fr

in traffic. In that case, W models the total amount of other activities the salesman has to realize while at rest, such as accounting, preparing presentations or reports. Finally, the problem also has applications in planning long tourism trips where waits can be seen as opportunities to visit minor attractions located on the route between the main spots whereas traffic kills the pleasure of holidays. In that case W represents the amount of time the tourists are willing to spend visiting the minor attractions.

1.1 Context and previous works

The study of the time-dependent shortest path problem (TDSP) dates back to Cooke and Halsey [6] who introduced an extension of Bellman's equations [3] to the time-dependent context. This early work has been followed by articles introducing the aforementioned *consistency assumption*. Also called FIFO (first-in first-out), this assumption leads to polynomial-time algorithms for the problem [9, 16, 14], through straightforward extensions of the algorithms for classical shortest path problem.

Allowing to wait (or park) at nodes makes the problem more complex since one must decide how long to wait at each node, in addition to choosing the path. Early variants for the TDSP with waiting include [16, 17] where it is shown that the shortest path may well be infinite. As these early papers could not handle certain kind of discontinuities, [8] proposed a new approach able to solve the problem for any kind of discontinuity. A general approach to the TDSP with waiting has been proposed by Cai et al. [4] who consider that each arc has a time-dependent cost, in addition to the time-dependent traversing time, and that waiting incurs a cost, also modeled by a time-dependent function. Their model also requires that the path reaches the destination within a given time horizon. Different dynamic programming algorithms have been proposed for this variant [4, 5, 7]. The resulting algorithms run in time polynomial in n, m, and the length of the time horizon, making them pseudo-polynomial. Related works consider alternative settings such as minimizing the excess time (over the minimum travel time) [2] or road networks with traffic lights [1].

Other variants of the TDSP with waiting have been considered by [10] and [15]. Both papers consider only traversing time functions and relax the time horizon constraint. On the one hand, if waiting is allowed only at the source node, Foschini et al. [10] show that the problem remains polynomially solvable. On the other hand, if waiting is allowed at each node, and bounds not greater than W are considered on each node, the problem becomes $\mathcal{NP}$-hard [15] and can be solved polynomially in n, m and W. Interestingly, the reduction provided in [15] does not extend to the case where the waiting times are bounded only by one global constraint. The purpose of this paper is to fill this gap: we prove that the TDSP allowing waiting at each node and bounding the waiting time only by one global constraint can be solved in polynomial time.

1.2 Definitions and notations

The travel time of an arc $e \in A$ is given by a positive continuous piecewise linear function $C_e : \mathbb{R}^+ \to \mathbb{R}_0^+$ defined by r_e pieces (and $r_e - 1$ breakpoints, not counting 0 as breakpoint), the last of which is a constant function. Each piece $s = 1, \ldots, r_e$ is an affine function $c_e^s + \rho_e^s t$ defined on the interval $[\tau_e^{s-1}, \tau_e^s]$, where $c_e^s, \rho_e^s \in \mathbb{R}$, $\tau_e^s \in \mathbb{R}^+$, $\tau_e^0 = 0$ and $\tau_e^{r_e} = +\infty$. We also denote the interval between 0 and the starting point of the last piece among all travel cost functions as $\Omega = [0, \max_{e \in A} \tau_e^{r_e - 1}]$. We assume in the paper that entering an arc e at time $t' \geq t$ leads to leaving the arc at time $t' + C_e(t') \geq t + C_e(t)$, which can be equivalently stated as follows.

Assumption 1 (Consistency assumption)**.** *Let $\rho_{min} = \min_{e,s} \rho_e^s$. We have $\rho_{min} \geq -1$.*

In what follows, we define a *path* from u to v as $p = (\nu; v_1(=u), v_2, \ldots, v_{\sigma(p)} = v)$ where ν indicates the starting time at node v_1, and $\sigma(p)$ denotes the number of nodes of p. We say that node w belongs to p if there is $i \in \{1, \ldots, \sigma(p)\}$ such that $v_i = w$. To simplify the notations used throughout, we assume that one can wait at all nodes of the graph ($U = V$), and define a *path-with-waits* as the couple (p, w), where p is a path and $\mathrm{w} = (\mathrm{w}_1, \ldots, \mathrm{w}_{\sigma(p)-1})$ is a $(\sigma(p)-1)$-tuple indicating the waiting time at each node along p. We further denote $t_{p,\mathrm{w}}(i)$ as the departure time from the i^{th} node in path-with-waits (p, w) and $t_p(i)$ as the departure time from the i^{th} node in path p. For the last node of a path p (resp. path-with-waits (p, w)), $t_p(\sigma(p))$ (resp. $t_{p,\mathrm{w}}(\sigma(p))$) denotes the arrival time at the node.

For $v \in V$, we define $\mathcal{PW}_v(\omega)$ be the set of paths-with-waits from o to v which satisfy $\sum_{i=1}^{\sigma(p)-1} \mathrm{w}_i \leq \omega$. Since we wish to minimize total travel time, the cost of (p, w) can be expressed as

$$C(p, \mathrm{w}) = \sum_{i=1}^{\sigma(p)-1} C_{v_i v_{i+1}}(t_{p,\mathrm{w}}(i)),$$

Let $T_v(\omega) = \min_{(p,\mathrm{w}) \in \mathcal{PW}_v(\omega)} C(p, \mathrm{w})$ be the minimum travel time among all paths-with-waits from o to v with total waiting time at most ω. Given $W \geq 0$, the aim of the article is to provide a polynomial-time algorithm for the optimization problem

$$T_d(W) = \min_{(p,\mathrm{w}) \in \mathcal{PW}_d(W)} C(p, \mathrm{w}). \tag{TDSPW}$$

We can assume without loss of generality that $W \in \Omega$ as it is usless to wait more than $\max_{e \in A} \tau_e^{r_e - 1}$.

We conclude the section by introducing further notations used throughout. Given $\omega < \omega'$, we denote by $p_{u \to v}(\omega, \omega')$ the shortest path (without waits) from u to v, leaving u at $T_u(\omega) + \omega'$. The total number of breakpoints of all travel time functions is denoted as $R = \sum_{e \in A}(r_e - 1)$. For $e = (u, v) \in A$ and $0 \leq \omega \leq W$, we denote the index of the right and left breakpoints of function C_e at $T_u(\omega) + \omega$ as $s_e(\omega)$ and $s_e^-(\omega)$, respectively. Formally,

$$s_e(\omega) = \min_s \{s : T_u(\omega) + \omega < \tau_e^s\} \quad \text{and} \quad s_e^-(\omega) = s_e(\omega) - 1. \tag{1}$$

To keep concise notations, we also define

$$\tau_u(\omega) = \min_{(u,v) \in A} \tau_{uv}^{s_{uv}(\omega)} \quad \text{and} \quad \tau_u^-(\omega) = \max_{(u,v) \in A} \tau_{uv}^{s_{uv}^-(\omega)}. \tag{2}$$

Next, given a one-variable function $f(x)$, we denote its left and right derivatives as $\partial_- f(x)$ and $\partial_+ f(x)$, respectively. Finally, we denote by by $SPP(\alpha, \beta)$ the complexity of solving a static shortest path problem on a digraph with α nodes and β edges, which is the same as solving a time-dependent shortest path problem without waiting on a digraph (e.g. [16]). Similarly, we let $ASPP(\alpha, \beta)$ be the complexity for solving a static shortest path problem on an acyclic digraph.

1.3 Contributions

The purpose of the article is to prove the following theorem.

Theorem 1. TDSPW *can be solved in* $\mathcal{O}(n \times R \times SPP(n, m))$.

Similarly to the method developed by Foschini et al. [10], our algorithm exploits the breakpoints of the travel time functions. Yet, our problem needs to handle the possibility of waiting at every node of the graph, unlike in [10] where it is allowed to wait only at the source node. Hence, even

when the path is given, the optimization problem considered herein involves n variables, one for each node, while the counterpart from [10] is a one-dimensional optimization problem.

We address this difficulty by decomposing optimal paths-with-waits into alternating sequences of waits and paths. This decomposition is done by gradually increasing the allowed amount of waiting time, starting from $\omega_1 = 0$. Specifically, at each iteration k of our algorithm, we will define ω_k as the allowed amount of waiting time, and we compute the breakpoint $\tau_v(\omega_k)$ for each node $v \in V$. Iteration k then examines what is the value of the maximum possible wait at each node u that, when followed by a shortest path from u to some other node v, hits the breakpoint $\tau_v(\omega_k)$. Taking the minimum among all nodes of these maximum possible waits, we obtain the additional waiting time allowed at iteration k, and adding that waiting time to ω_k yields ω_{k+1}. Once ω_{k+1} is known, we compute $\{T_v(\omega_{k+1}), v \in V\}$ from $\{T_v(\omega_k), v \in V\}$ by applying a classical shortest path algorithm.

A different approach has been followed simultaneously and independently by [11, 12]. Whenever there is no waiting limit (setting W large enough), the authors of [11] decompose optimal paths-with-waits into alternating sequences of waits and paths by expanding forward and backward shortest path trees from each breakpoint and computing shortest paths on the resulting time-indexed networks. The resulting algorithm solves the problem in

$$\mathcal{O}(R \times SPP(n, m) + ASPP(nR, mR)). \tag{3}$$

They further extend their approach to handle bound W in [12] essentially by solving up to $\mathcal{O}(n \times R^2)$ variants of the problem with no wait limits, resulting in an algorithm running in

$$\mathcal{O}(n \times R^3 \times SPP(n, m) + R^2 \times ASPP(nR, mR)). \tag{4}$$

We see that our result, stated in Theorem 1, compares in the general case favorably with the result of [12], having the complexity stated in (4), while the case without waiting (obtained by setting W large enough) is solved more efficiently by the result of [11], having the complexity stated in (3).

1.4 Outline

The rest of the paper is structured as follows. In the next section, we state basic properties of the optimal paths-with-waits and the recurrence relations satisfied by $T_v(\omega)$. The section also shows that if we are able to construct a sequence $\omega_1, \omega_2, \ldots, \omega_K$ for which the following recurrence relation holds

$$T_v(\omega_{k+1}) = \min \left\{ T_v(\omega_k), \min_{(u,v)\in A} T_u(\omega_{k+1}) + C_{uv} \left(T_u(\omega_{k+1}) + \omega_{k+1} \right) \right\}, \tag{5}$$

then we obtain a polynomial-time algorithm for the problem. Section 3 further analyzes the structure of the paths-with-waits realizing $T_v(\omega)$. The first important result of the section is to show that when $\omega_1, \omega_2, \ldots, \omega_K$ are such that

$$T_u(\omega_{k+1}) + \omega_{k+1} \leq \tau_u(\omega_k), \tag{6}$$

holds for each $k \in \{1, \ldots, K-1\}$, then the discrete recurrence relation (5) holds. The section then decomposes any path-with-waits realizing $T_v(\omega_{k+1})$ into a concatenation of a path-with-waits realizing $T_u(\omega_k)$, an additional wait $\omega_{k+1} - \omega_k$ at node u, and a shortest path from u to v. Section 4 leverages that decomposition to construct dynamically a sequence $\omega_1, \omega_2, \ldots, \omega_K$ satisfying property (6). Hence, using the result from Section 3, we know that the recurrence relation (5) holds, and the latter can be solved as explained in Section 2. The main ingredient of the construction of the sequence is an algorithm that backpropagates the values $\tau_u(\omega_k)$ for each $u \in V$. Section 4 also presents the overall algorithm solving the problem.

2 Preliminaries

In what follows, we introduce a dynamic programming recursions for TDSPW. These recursions involve infinitely many states and cannot be used directly to solve TDSPW. However, we show that if we can guarantee that only a polynomial number of these states (which will be indexed by $k = 1, \ldots, K$) is needed to obtain the optimal solution, then TDSPW can be solved in polynomial time by applying K times a shortest path algorithm. The section also presents key properties of the value-functions involved in the recursions, which will be used often along the manuscript.

Let $v \in V$ and $\omega \leq W$. Our algorithm is based on the recurrence relation that relates $T_v(\omega)$ to the minimum travel time to the predecessors of v. To derive the relation, we introduce $T_v^=(\omega)$ as the minimum travel time among all paths from o to v with total waiting time *exactly* ω, not including the time waited at v. Formally,

$$T_v^=(\omega) = \min_{(p,\mathrm{w})\in\mathcal{PW}_v(\omega)} \left\{ C(p,\mathrm{w}) : \sum_{i=1}^{\sigma(p)-1} \mathrm{w}_i = \omega \right\}.$$

For this quantity, one can verify that the following recurrence holds.

$$T_v^=(\omega) = \min_{(u,v)\in A} \left\{ \min_{\omega'\leq\omega} T_u^=(\omega') + C_{uv}(T_u^=(\omega') + \omega) \right\}.$$

Using the consistency assumption, we observe that $T_u^=(\omega') + C_{uv}(T_u^=(\omega') + \omega)$ is minimum if $T_u^=(\omega')$ is minimum. Since $T_u(\omega) = \min_{\omega'\leq\omega} T_u^=(\omega')$, the above recurrence relation can be equivalently written as

$$T_v^=(\omega) = \min_{(u,v)\in A} T_u(\omega) + C_{uv}(T_u(\omega) + \omega). \tag{7}$$

As a consequence, we can distinguish two cases in the computation of $T_v(\omega)$, depending on whether it is achieved with a total wait equal to or lower than ω. If there exists $\omega' < \omega$ such that $T_v(\omega) = T_v^=(\omega')$, then

$$T_v(\omega) = T_v(\omega'). \tag{8}$$

Otherwise, $T_v(\omega) = T_v^=(\omega)$, so we can directly rewrite (7) as

$$T_v(\omega) = \min_{(u,v)\in A} T_u(\omega) + C_{uv}(T_u(\omega) + \omega). \tag{9}$$

Grouping relations (8) and (9), $T_v(\omega)$ satisfies the recurrence relation

$$T_v(\omega) = \min \left\{ \inf_{0\leq\omega'<\omega} T_v(\omega'), \min_{(u,v)\in A} T_u(\omega) + C_{uv}(T_u(\omega) + \omega) \right\}. \tag{10}$$

The bottom line of our algorithm is to build a sequence with polynomial length $0 = \omega_1, \omega_2, \ldots, \omega_K = W$ such that relation (10) can be simplified to

$$T_v(\omega_{k+1}) = \min \left\{ T_v(\omega_k), \min_{(u,v)\in A} T_u(\omega_{k+1}) + C_{uv}\left(T_u(\omega_{k+1}) + \omega_{k+1}\right) \right\}, \tag{11}$$

which can be equivalently formulated as

$$T_v(\omega_{k+1}) = \min \left\{ T_v(\omega_k), T_v^=(\omega_{k+1}) \right\}. \tag{12}$$

Indeed, we show below that for such a sequence, $\{T_u(\omega_{k+1})\}_{u\in V}$ can be computed in polynomial time from $\{T_u(\omega_k)\}_{u\in V}$ using classical shortest path algorithms.

Proposition 1. *Let $\{T_v(\omega_k), v \in V\}$ be given and $\{T_v(\omega_{k+1}), v \in V\}$ be a set of numbers that satisfy* (11) *and the initial conditions $T_o(\omega_{k+1}) = 0$. Then, $\{T_v(\omega_{k+1}), v \in V\}$ can be computed by classical shortest path algorithms in $\mathcal{O}(SPP(n, m))$.*

Proof. We create a new instance of the problem defined on the graph $G' = (V', A')$ and denote the travel time functions as C'_e for each $e \in A'$. The new graph contains a copy of each node v different from the source, connected only to o and v, namely $V' = V \cup \{v',\ \forall v \in V \setminus \{o\}\}$ and $A' = A \cup \{(o, v'),\ \forall v \in V \setminus \{o\}\} \cup \{(v', v),\ \forall v \in V \setminus \{o\}\}$. We further define the travel time functions as $C'_{ov'}(t) = 0$ and $C'_{v'v}(t) = T_v(\omega_k)$ for all $t \geq 0$ and $v \in V \setminus \{o\}$, while $C'_{uv}(t) = C_{uv}(t + \omega_{k+1})$ for each $(u, v) \in A$. Denoting $T_v(\omega_{k+1})$ as d_v for each $v \in V'$, and initializing $d_o = 0$, and $d_v = +\infty$ for each $v \in V' \setminus \{o\}$, one readily verifies that (11) can be reformulated as

$$d_v = \min_{(u,v)\in A'} d_u + C'_{uv}(d_u),$$

the classical recurrence relations for the shortest path-problem in time-dependent networks. As G' contains $2n-1$ nodes and $m+2n-2$ arcs, the problem can be solved in $SPP(2n-1, m+2n-2) \in \mathcal{O}(SPP(n, m))$ when the consistency assumption is satisfied (e.g. [16]). □

We conclude the section by introducing useful properties of $T_v(\omega)$ and TDSPW. First, we show that the consistency assumption implies that the left-derivative of $T_v(\omega)$ is bounded below by -1.

Proposition 2. *$\partial_- T_v(\omega) \geq -1$ for all $\omega > 0$ and $v \in V$. Hence, $\omega \mapsto T_v(\omega) + \omega$ is a non-decreasing function.*

Proof. Consider $\tilde{\omega} > 0$ and let $(\tilde{p}, \tilde{\mathrm{w}}) \in \mathcal{PW}_v(\tilde{\omega})$ be a path-with-waits from o to v that achieves $T_v(\tilde{\omega})$, so $C(\tilde{p}, \tilde{\mathrm{w}}) = T_v(\tilde{\omega})$. Let v_j be the last node along $\tilde{p}$ for which $\tilde{\mathrm{w}}_j > 0$ and define $\mathrm{w} \in \mathbb{R}^n$ as $\mathrm{w}_i = \tilde{\mathrm{w}}_i$ for $i \neq j$ and $\mathrm{w}_j = \tilde{\mathrm{w}}_j - \delta$ for some $0 < \delta \leq \tilde{\mathrm{w}}_j$, as well as $\omega = \sum_{i=1}^{\sigma(\tilde{p})-1} \mathrm{w}_i = \tilde{\omega} - \delta$. We claim that

$$\frac{C(\tilde{p}, \tilde{\mathrm{w}}) - C(\tilde{p}, \mathrm{w})}{\delta} \geq -1. \tag{13}$$

Therefore,

$$\begin{aligned}\frac{T_v(\tilde{\omega} + (-\delta)) - T_v(\tilde{\omega})}{(-\delta)} &= \frac{T_v(\tilde{\omega}) - T_v(\tilde{\omega} - \delta)}{\delta} = \frac{T_v(\tilde{\omega}) - T_v(\omega)}{\delta} \\ &= \frac{C(\tilde{p}, \tilde{\mathrm{w}}) - \min\limits_{(p', \mathrm{w}') \in \mathcal{PW}_v(\omega)} C(p', \mathrm{w}')}{\delta} \geq \frac{C(\tilde{p}, \tilde{\mathrm{w}}) - C(\tilde{p}, \mathrm{w})}{\delta} \geq -1.\end{aligned}$$

and the results follows by taking the limit for $\delta \to 0^+$.

Let us now prove the claim (13). By definition, $t_{\tilde{p},\mathrm{w}}(j) = t_{\tilde{p},\tilde{\mathrm{w}}}(j) - \delta$ and, since $\rho_{min} \geq -1$, $t_{\tilde{p},\mathrm{w}}(\ell) \leq t_{\tilde{p},\tilde{\mathrm{w}}}(\ell)$ for each $j \leq \ell \leq \sigma(\tilde{p})$. In particular, $t_{\tilde{p},\mathrm{w}}(\sigma(\tilde{p})) \leq t_{\tilde{p},\tilde{\mathrm{w}}}(\sigma(\tilde{p}))$, and the claim follows from the relation $C(p', \mathrm{w}') = t_{p',\mathrm{w}'}(\sigma(p')) - \sum_{i=1}^{\sigma(p')-1} \mathrm{w}'_i$ valid for any path-with-waits (p', w'). □

Remark 1. *The function $T_v(\omega)$ is non-increasing for each $v \in V$. In contrast, function $T_v^{=}(\omega)$ may be increasing, as can be seen in the two-nodes example where $V = \{o, d\}$ and $A = \{(o, d)\}$ and $C_{od}(t) = \min(t, W)$. In that example, $T_v^{=}(\omega) = \omega$ for each $0 \leq \omega \leq W$.*

Next we show that we can restrict our attention to acyclic solutions.

Proposition 3. TDSPW *admits an acyclic optimal solution.*

Proof. Consider a solution (p, w) such that $p = (\nu; o = v_1, \ldots, v_{\sigma(p)} = d)$ contains a cycle $c = (\nu_c; v_i, v_{i+1}, \ldots, v_j)$, where $v_i = v_j$. Let $C(c)$ be the travel time of cycle c in (p, w). By non-negativity of arc travel times, we know that $C(c) \geq 0$. As a consequence, if $j = \sigma(p)$, we get a solution with smaller or equal cost by removing c and the associated waiting times from (p, w). If $j < \sigma(p)$, let $\mu_k = \sum_{\ell=1}^{k} \mathrm{w}_\ell$ and $(p(k), \mathrm{w}(k))$ is the subpath-with-waits of (p, w) that goes from o to v_k, for all $k \in \{2, \ldots, \sigma(p)\}$. Assumption 1 ensures that $\partial_+ C_{uv} \geq -1, \forall (u, v) \in A$, so

$$\begin{aligned} C\left(p(j+1), \mathrm{w}(j+1)\right) &= C\left(p(i), \mathrm{w}(i)\right) + C(c) + C_{v_j v_{j+1}}\left(C(p(i), \mathrm{w}(i)) + C(c) + \mu_j\right) \\ &\geq C(p(i), \mathrm{w}(i))) + C_{v_j v_{j+1}}\left(C(p(i), \mathrm{w}(i))) + \mu_j\right) \\ &= C\left(p(i), \mathrm{w}(i)\right) + C_{v_i v_{j+1}}\left(C(p(i), \mathrm{w}(i))) + \mu_i + (\mu_j - \mu_i)\right), \end{aligned}$$

where the last expression is the travel time from o to v_{j+1} along (p, w) if we remove c and and wait an additional $\mu_j - \mu_i$ at $v_i = v_j$. This shows that c can be removed without increasing the total travel time nor the total waiting time along the path-with-waits. □

3 Structure of optimal paths

The sequence $\omega_1, \ldots, \omega_K$ will be constructed so that for all $k = 1, \ldots, K$ there is some arc (u, v) such that a breakpoint of C_{uv} is reached at $T_u(\omega_k) + \omega_k$. The practical algorithm that identifies this sequence, provided in the next section, is based on the structure of the paths-with-waits that realize the minimum travel time $T_v(\omega)$. For this, if $(p, \mathrm{w}) \in \mathcal{PW}_v(\omega)$, we define $(p(i), \mathrm{w}(i))$ and $(\overline{p}(i), \overline{\mathrm{w}}(i))$ as the subpaths-with-waits of (p, w) respectively ending and starting at the i-th node of p. In this section, we provide key properties of the paths-with-waits that realize $T_v(\omega)$ for each $v \in V$ and $\omega \geq 0$. Specifically, we identify that these paths-with-waits can be split into two subpaths, the second of which contains no more waiting. This property will be essential to compute the values $\omega_1, \ldots, \omega_K$ by back-propagation in the following section.

Definition 1. *Let $v \in V$, $0 \leq \omega \leq W$ and $(p, \mathrm{w}) \in \mathcal{PW}_v(\omega)$ where $p = (\nu; v_1 := o, v_2, \ldots, v_{\sigma(p)} = v)$ and $\mathrm{w} = (\mathrm{w}_1, \ldots, \mathrm{w}_{\sigma(p)-1})$. If there is $i \leq \sigma(p) - 1$ such that $\sum_{k=1}^{i} \mathrm{w}_k = \omega$ and $\sum_{k=1}^{i-1} \mathrm{w}_k = \omega' < \omega$, then $(\overline{p}(i), \overline{\mathrm{w}}(i))$ is the* saturated subpath of (p, w). *By extension the nodes $v_i, \ldots, v_{\sigma(p)} = v$ are said to be* saturated *in (p, w) and v_i is the* first saturated node *in (p, w).*

From this definition, one can notice that the first saturated node of a path-with-waits is also the last one where the wait is positive. The introduction of this saturated subpath allows for a more precise characterization of optimal paths.

Proposition 4. *Let $v \in V$, $0 \leq \omega \leq W$ such that $T_v(\omega) = T_v^{=}(\omega)$. Then, there exists $(p, \mathrm{w}) \in \mathcal{PW}_v(\omega)$ such that*

- *$C(p, \mathrm{w}) = T_v(\omega)$, $\sum_{\ell=1}^{\sigma(p)-1} \mathrm{w}_\ell = \omega$, and*
- *$C(p(j), \mathrm{w}(j)) = T_{v_j}(\omega)$ for every saturated node v_j in (p, w).*

In particular, $T_{v_j}(\omega) = T_{v_j}^{=}(\omega)$ for every saturated node v_j after the first saturated node.

Proof. Let $(p, \mathrm{w}) \in \mathcal{PW}_v(\omega)$ such that $C(p, \mathrm{w}) = T_v(\omega)$ and $\sum_{\ell=1}^{\sigma(p)-1} \mathrm{w}_\ell = \omega$. Assume that there is v_j, a saturated node in (p, w), such that $C(p(j), \mathrm{w}(j)) > T_{v_j}(\omega)$. There is $(p^j, \mathrm{w}^j) \in \mathcal{PW}_{v_j}(\omega)$ such that $C(p^j, \mathrm{w}^j) = T_{v_j}(\omega)$ and $\sum_{\ell=1}^{\sigma(p^j)-1} \mathrm{w}_\ell^j = \omega^j \leq \omega$. We can then build a path-with-waits (p', w') from o to v with total wait ω, by appending $(\overline{p}(j), \overline{\mathrm{w}}(j))$ to (p^j, w^j) and waiting an additional

$\omega - \omega^j$ at v_j. Node v_j is saturated in both (p, w) and (p', w'), so the departure time from v_j is $C(p(j), \mathrm{w}(j))+\omega$ in (p, w) and $T_{v_j}(\omega)+\omega$ in (p', w'). And since (p, w) and (p', w') are identical from v_j to v, the consistency assumption and $C(p(j), \mathrm{w}(j))+\omega > T_{v_j}(\omega)+\omega$ yields $C(p, \mathrm{w}) \geq C(p', \mathrm{w}')$. From $C(p, \mathrm{w}) = T_v(\omega)$, we deduce that $C(p', j') = T_v(\omega)$. We thus obtain the desired path-with-waits by applying the above approach recursively on the saturated nodes until $C(p(j), \mathrm{w}(j)) = T_{v_j}(\omega)$ for every saturated node v_j.

The last assertion follows from the fact that the total wait along $(p(j), \mathrm{w}(j))$ is exactly equal to ω.

□

Remark 2. *We observe that if i is the first saturated node in (p, w), there is no wait from v_{i+1} to v in (p, w). If $C(p, \mathrm{w}) = T_v(\omega)$, this means that $\overline{p}(i)$ is the shortest path (without wait) from v_i to v among those departing from v_i at $T_{v_i}(\omega) + \omega$.*

Remark 3. *In the proof of Proposition 4, we use the consistency assumption to show that $C(p(j), \mathrm{w}(j))+\omega > T_{v_j}(\omega)+\omega$ implies $C(p, \mathrm{w}) \geq C(p', \mathrm{w}')$. Actually, it can happen that $C(p, \mathrm{w}) = C(p', \mathrm{w}')$ only if there is an arc $(v_\ell, v_{\ell+1})$ in p such that $\ell \geq j$ and $C_{\ell,\ell+1}$ has a slope equal to -1 at $C(p(\ell), \mathrm{w}(\ell))+\omega$. As a consequence, if there is no such arc in the saturated subpath of (p, w), we can show that $C(p, \mathrm{w}) = T_v(\omega)$ directly implies that $C(p(j), \mathrm{w}(j)) = T_{v_j}(\omega)$ for every saturated node v_j.*

Next we prove a property which, if satisfied by sequence $\omega_1, \ldots, \omega_K$, directly implies that (12) holds.

Proposition 5. *Let $\omega_{k+1} \geq \omega_k$ be such that for all $u \in V$,*

$$T_u(\omega_{k+1}) + \omega_{k+1} \leq \tau_u(\omega_k). \tag{14}$$

Then for all $v \in V$, $\omega \mapsto T_v(\omega)$ is a continuous piecewise linear and concave function on $[\omega_k, \omega_{k+1}]$.

Proof. See detailed proof in Appendix A. □

Corollary 1. *Let $\omega_{k+1} \geq \omega_k$ be such that (14) holds. Then for all $v \in V$, (12) holds, i.e.,*

$$T_v(\omega_{k+1}) = \min\left\{T_v(\omega_k), T_v^{=}(\omega_{k+1})\right\}.$$

Proof. By concavity of T_v on $[\omega_k, \omega_{k+1}]$, T_v is either constant on $[\omega_k, \omega_{k+1}]$ or there is $\bar{\omega} \in [\omega_k, \omega_{k+1}]$ such that T_v is decreasing on $[\bar{\omega}, \omega_{k+1}]$. In the latter case, we get that $T_v(\omega) = T_v^{=}(\omega)$ for all $\omega \in [\bar{\omega}, \omega_{k+1}]$. In particular, $T_v(\omega_{k+1}) = T_v^{=}(\omega_{k+1})$. To summarize, $T_v(\omega_{k+1}) = \min\{T_v(\omega_k), T_v^{=}(\omega_{k+1})\}$.

□

The above result allows to get more specific in the characterization of the first saturated node of the path exhibited in Proposition 4 when $\omega = \omega_{k+1}$ for some integer k.

Corollary 2. *Let $0 \leq \omega_k < \omega_{k+1} \leq W$ such that (14) holds and consider $v \in V$ such that $T_v(\omega_{k+1}) = T_v^{=}(\omega_{k+1})$. Then, there exists $(p, \mathrm{w}) \in \mathcal{PW}_v(\omega_{k+1})$ such that*

- *$C(p, \mathrm{w}) = T_v(\omega_{k+1})$ and $\sum_{\ell=1}^{\sigma(p)-1} \mathrm{w}_\ell = \omega_{k+1}$;*
- *$C(p(i), \mathrm{w}(i)) = T_{v_i}(\omega_k)$, where v_i is the first saturated node in (p, w);*
- *and $C(p(j), \mathrm{w}(j)) = T_{v_j}^{=}(\omega_{k+1})$ for every other saturated node v_j.*

Proof. Proposition 4 guarantees that there is $(p, \mathrm{w}) \in \mathcal{PW}_v(\omega)$ such that $C(p, \mathrm{w}) = T_v(\omega_{k+1})$, $\sum_{\ell=1}^{\sigma(p)-1} \mathrm{w}_\ell = \omega_{k+1}$, and $C(p(j), \mathrm{w}(j)) = T_{v_j}(\omega_{k+1})$ for every saturated node v_j. Denoting the first saturated node as v_i, we know that the total wait up to it in (p, w), $\omega^i = \sum_{\ell=1}^{i-1} \mathrm{w}_\ell$, is less than ω_{k+1}. As a consequence, $C(p(i), \mathrm{w}(i)) = T_{v_i}(\omega_{k+1})$ yields $T_{v_i}(\omega) = T_{v_i}(\omega_{k+1})$ for all $\omega \in [\omega^i, \omega_{k+1}]$, so that $\partial_+ T_{v_i}(\omega) = 0$ for $\omega \in [\omega^i, \omega_{k+1}[$. By concavity of T_{v_i} on $[\omega_k, \omega_{k+1}]$, $\partial_+ T_{v_i}(\omega) = 0$ for $\omega \in [\omega_k, \omega_{k+1}[$, so we conclude that $T_{v_i}(\omega) = T_{v_i}(\omega_{k+1})$ for all $\omega \in [\omega_k, \omega_{k+1}]$, and therefore, $C(p(i), \mathrm{w}(i)) = T_{v_i}(\omega_k)$. □

The above two results immediately yield the concluding result of the section, which is essential to the justification of the polynomial algorithm described in next section. Indeed, the path-with-waits realizing $T_v(\omega_{k+1})$ can now be expressed as the concatenation of a path-with-waits up to some node u for a total wait not greater than ω_k, and a path-without-waits leaving u at $T_u(\omega_k) + \omega_{k+1}$. In particular, this means that given $T_u(\omega_k)$ for all $u \in V$, we will be able to focus on such paths during the search for ω_{k+1}.

Theorem 2. *Let $0 \leq \omega_k < \omega_{k+1} \leq W$ such that (14) holds and consider $v \in V$. Then,*

$$T_v(\omega_{k+1}) = \min \left\{ T_v(\omega_k), \min_{u \in V \setminus \{v\}} \left\{ T_u(\omega_k) + C\left(p_{u \to v}(\omega_k, \omega_{k+1})\right) \right\} \right\}.$$

Observe that $C\left(p_{v \to v}(\omega_k, \omega_{k+1})\right) = 0, \forall v \in V$, so the above result can be equivalently written as

$$T_v(\omega_{k+1}) = \min_{u \in V} \left\{ T_u(\omega_k) + C\left(p_{u \to v}(\omega_k, \omega_{k+1})\right) \right\}.$$

4 Building the sequence

We describe next how to construct the sequence $\omega_1, \ldots, \omega_K = W$, starting from $\omega_1 = 0$ and increasing the value of ω_k along the iterations. From Proposition 5, we wish to define the next iterate ω_{k+1} as the largest value (not greater than W) such that for all $v \in V$, condition (14) is satisfied, i.e.,

$$T_v(\omega_{k+1}) + \omega_{k+1} \leq \tau_v(\omega_k).$$

Stated otherwise, ω_{k+1} is the next waiting value such that a breakpoint of an edge cost function is reached by $T_v(\omega_{k+1}) + \omega_{k+1}$ for some $v \in V$. This leads us to define the sequence $\omega_1, \ldots, \omega_K = W$ as in the following theorem. With this definition, we guarantee that (14) is satisfied for all $v \in V$ and that the sequence contains at most as many elements as the total number of breakpoints in the edge cost functions.

Theorem 3. *Let $\omega_1 = 0$ and consider the sequence defined recursively as*

$$\omega_{k+1} = \min \left\{ W, \min_{v \in V} \left\{ \max_{\omega \in \Omega} \{\omega : T_v(\omega) + \omega \leq \tau_v(\omega_k)\} \right\} \right\} \tag{15}$$

Then, there is $K \in \{1, \ldots, R\}$ such that $\omega_K = W$, and $\forall k \in \{1, \ldots, K-1\}$

$$T_v(\omega_{k+1}) + \omega_{k+1} \leq \tau_v(\omega_k), \forall v \in V.$$

Proof. Let $k \in \{1, \ldots, R-1\}$. Let $v^* \in \arg\min_{v \in V} \{\max_{\omega \in \Omega} \{\omega : T_v(\omega) + \omega \leq \tau_v(\omega_k)\}\}$ and $\omega_{v^*} = \max_{\omega \in \Omega} \{\omega : T_{v^*}(\omega) + \omega \leq \tau_{v^*}(\omega_k)\}$. If $\omega_{v^*} \geq W$, then (15) yields $\omega_{k+1} = W$ with $k+1 \leq R$. Otherwise, we get $\omega_{k+1} = \omega_{v^*}$, so $T_{v^*}(\omega_{k+1}) + \omega_{k+1} = \tau_{v^*}(\omega_k)$. By definition of $\tau_{v^*}(\omega_{k+1})$, we know

that $\tau_{v^*}(\omega_{k+1}) > T_{v^*}(\omega_{k+1}) + \omega_{k+1}$, so $\tau_{v^*}(\omega_{k+1}) > \tau_{v^*}(\omega_k)$. Moreover, for any ω, $\tau_v(\omega)$ is the breakpoint of a cost function of an arc leaving from v. As a consequence, $\tau_{v^*}(\omega_k)$ and $\tau_{v^*}(\omega_{k+1})$ correspond to two different breakpoints.

Since the function $\omega \mapsto T_v(\omega)+\omega$ is non-decreasing for all $v \in V$, we deduce that each breakpoint of each arc cost function can correspond to at most one element of the sequence $\omega_1, \ldots, \omega_K$, hence $K \leq R$.

Finally, the recursive definition of ω_{k+1} directly yields

$$T_v(\omega_{k+1}) + \omega_{k+1} \leq \tau_v(\omega_k), \forall v \in V.$$

□

We now wish to build the sequence $\omega_1, \ldots, \omega_K$ by using the recursive relation (15). Assuming that $\omega_1, \ldots, \omega_k$ are known for some $k \geq 1$, it is not clear though how to use (15) to actually compute ω_{k+1}, because we do not know the value of $T_v(\omega)$ for $\omega > \omega_k$. The contribution of this section is to construct a back-propagation algorithm that will be essential in computing these values. Ideally, such a back-propagation algorithm would construct reverse shortest paths from any given node v at time t based on the cost function C_e for every $e \in A$; for instance, if (u, v) is an edge in such a shortest path, the algorithm would relate t' and t through $t = t' + C_{uv}(t')$. An important difficulty faced by such an algorithm is that when back-propagating some time t along an arc (u, v), we do not know in which piece of the cost function C_{uv} the arc has been entered. Somewhat surprisingly, we show in the section that for each arc, the only piece that matters is $s_e(\omega_k)$ (see (1)), defined on the interval $[\tau_e^{s_e(\omega_k)-1}, \tau_e^{s_e(\omega_k)}]$. This allows to back-propagate the values of $\tau_v(\omega_k)$ along particulars paths that reach v at $\tau_v(\omega_k)$. Those are defined as follows.

Definition 2. *Let $u \in V, v \in V$ and $\omega \in [0, W]$. We denote as $\mathcal{P}^{\text{back}}_{u\to v}(\omega)$ the set of paths such that $p = (\nu; v_1, \ldots, v_{\sigma(p)}) \in \mathcal{P}^{\text{back}}_{u\to v}(\omega)$ if and only if:*

- *$v_1 = u$, $v_{\sigma(p)} = v$,*
- *p reaches v at $\tau_v(\omega)$, and*
- *$t_p(v_i) \in [T_{v_i}(\omega) + \omega, \tau_{v_i}(\omega)]$, for $i = 1, \ldots, \sigma(p) - 1$.*

The departing time from u of the shortest path among $\mathcal{P}^{\text{back}}_{u\to v}(\omega)$ is then denoted as $t_{u\to v}(\omega)$, i.e.,

$$t_{u\to v}(\omega) = \begin{cases} \tau_v(\omega) - \min\{C(p) : p \in \mathcal{P}^{\text{back}}_{u\to v}(\omega)\}, & \text{if } \mathcal{P}^{\text{back}}_{u\to v}(\omega) \neq \emptyset, \\ +\infty, & \text{otherwise.} \end{cases}$$

Out of completeness in the proofs, we also set $t_{v\to v}(\omega) = \tau_v(\omega)$. The above also implies that $\mathcal{P}^{\text{back}}_{u\to v}(\omega) = \emptyset$ when $\tau_v(\omega) = +\infty$ (so $t_{u\to v}(\omega) = +\infty$ in that case).

Let us denote

$$V(\omega_k) = \arg\min_{v\in V} \left\{ \max_{\omega\in\Omega} \{\omega : T_v(\omega) + \omega \leq \tau_v(\omega_k)\} \right\}.$$

Before moving to the description of the actual back-propagation algorithm, we show below that ω_{k+1} can indeed be computed by searching for shortest paths in $\mathcal{P}^{\text{back}}_{u\to v}(\omega_k)$ for $u \in V$ and $v \in V(\omega_k)$.

Proposition 6. *Let $\omega_1, \ldots, \omega_K$ be constructed as in Theorem 3. Then, for each $k \in \{1, \ldots, K-1\}$ and $v \in V(\omega_k)$,*

$$\omega_{k+1} = \min \left\{ W, \max_{u\in V} \{t_{u\to v}(\omega_k) - T_u(\omega_k)\} \right\}. \tag{16}$$

Proof. See detailed proof in Appendix B. □

Algorithm 1 details the back-propagation that will be used to search for shortest paths in $\mathcal{P}^{\text{back}}_{u\to v}(\omega_k)$. The algorithm requires that $T_u(\omega_k)$ and $\tau_u(\omega_k)$ be already computed for all $u \in V$. Then for each input node v, it back-propagates the departure time $\tau_v(\omega_k)$ along paths of $\mathcal{P}^{\text{back}}_{u\to v}(\omega_k)$ for each $u \in V$ with the aim of computing $t_{u\to v}(\omega_k)$. One additional difficulty is that Proposition 6 is only concerned with nodes that belong to $V(\omega_k)$, which are not known at this stage. The focus of what follows will thus be to show that Algorithm 1 returns $\max_{u\in V}\{t_{u\to v}(\omega_k) - T_u(\omega_k)\}$ for at least one input node $v \in V(\omega_k)$, and that it returns a larger or equal value for every other node (see Lemma 1). As a consequence, Proposition 6 guarantees that ω_{k+1} will be found by considering the smallest value returned by Algorithm 1 among every node.

For more insight into the back-propagation, first observe that by definition of $\tau_u(\omega_k)$ and $\tau^-_u(\omega_k)$ (see (2)), we know that $\tau^-_u(\omega_k) \leq T_u(\omega_k) + \omega_k < \tau_u(\omega_k)$ for all $u \in V$. As a consequence, if $p \in \mathcal{P}^{\text{back}}_{u\to v}(\omega_k)$, then $t_p(v_j) \in [\tau^-_{v_j}(\omega_k), \tau_{v_j}(\omega_k)]$ for all $v_j \in p$. If (u, w) is an edge of a path in $p \in \mathcal{P}^{\text{back}}_{u\to v}(\omega_k)$, we can then relate $t_p(u)$ and $t_p(w)$ through the formula

$$t_p(w) = t_p(u) + C_{uw}(t_p(u)) = t_p(u) + \rho^{s_{uw}(\omega_k)}_{uw} t_p(u) + c^{s_{uw}(\omega_k)}_{uw}. \tag{17}$$

This formula justifies that steps 6–16 of Algorithm 1 describe a backward label-setting algorithm. In these steps the departure times at $u \in V$, t_u, is updated until the shortest path in $\mathcal{P}^{\text{back}}_{u\to v}(\omega_k)$ is found or until we can prove that the back-propagation of $\tau_v(\omega_k)$ is not useful. The non-marked node with largest departure time, w, is selected and marked at steps 7–10, among which steps 8 and 9 ensure that $t_w \in [T_w(\omega_k) + \omega_k, \tau_w(\omega_k)]$. As we will show in Lemma 2 presented in Appendix C, if the conditions checked at these steps are both wrong, then $t_w = t_{w\to v}(\omega_k)$. Formula (17) is then used to update t_u for every arc (u, w) at steps 11-16, wherein steps 15-16 address the special case $\rho^{s_{uw}(\omega_k)}_{uw} = -1$. When this happens, (17) becomes $t_p(w) = c^{s_{uw}(\omega_k)}_{uw}$, so the departure time at w does not depend on the departure time at u. If $c^{s_{uw}(\omega_k)}_{uw} > t_w$, this means that w can never be reached early enough to get to v at $\tau_v(\omega_k)$ by taking arc (u, w). We can thus skip this iteration in the for loop. If $c^{s_{uw}(\omega_k)}_{uw} \leq t_w$, we show in Lemma 2 presented in Appendix C that the back-propagation of $\tau_v(\omega_k)$ can be stopped here to treat another node.

The execution of the back-propagation loop is described formally in the following proposition. This will be essential to prove the overall validity of Algorithm 1.

We are now ready to prove that the overall approach is valid, that is, $\omega_{k+1} = \min\{W, \min_{v\in V} \omega_v\}$, where ω_v is the value returned by Algorithm 1 when back-propagating $\tau_v(\omega_k)$.

Lemma 1. *For all $v \in V$, denote as ω_v the value returned by Algorithm 1 when back-propagating $\tau_v(\omega_k)$. There is $v^* \in \arg\min_{v\in V} \{\max_{\omega\in\Omega}\{\omega : T_v(\omega) + \omega \leq \tau_v(\omega_k)\}\}$ such that*

1. *$\omega_{v^*} = \max_{u\in V}\{t_{u\to v^*}(\omega_k) - T_u(\omega_k)\}$, and*
2. *$\omega_v \geq \omega_{v^*}, \forall v \in V$.*

Proof. See detailed proof in Appendix C. □

Corollary 3. *Let $\omega_1, \ldots, \omega_k$ be defined recursively as in Theorem 3. For all $v \in V$, denote as ω_v the value returned by Algorithm 1 when back-propagating $\tau_v(\omega_k)$. Then,*

$$\min_{v\in V}\{\omega_v\} = \min_{v\in V}\{\max_{u\in V}\{t_{u\to v}(\omega_k) - T_u(\omega_k)\}\}.$$

Proof. The result is the direct consequence of Lemma 1 and Proposition 6. □

input: $\omega_k, v \in V, \tau_u(\omega_k), T_u(\omega_k), \forall u \in V,$

1 $V' \leftarrow V$
2 $S \leftarrow \emptyset$ // set of marked nodes
3 $t_u \leftarrow -\infty, \forall u \in V'$
4 $t_v \leftarrow \tau_v(\omega_k)$
5 **if** $\tau_v(\omega_k) = +\infty$ **then** $\omega_v \leftarrow +\infty$, **go to** step 18
6 **while** $S \neq V'$ **do**
7 select w in $\arg\max_{u \in V' \setminus S} \{t_u\}$
8 **if** $t_w > \tau_w(\omega_k)$ **then** $\omega_v \leftarrow +\infty$, **go to** step 18 // $\tau_w(\omega_k)$ is reached before $\tau_v(\omega_k)$
9 **if** $t_w < T_w(\omega_k) + \omega_k$ **then** $V' \leftarrow V' \setminus \{w\}$, **go to** step 7 // impossible to backtrack to w late enough
10 $S \leftarrow S \cup \{w\}$ // mark node w
11 **for** $(u, w) \in A$ // back-propagation loop of t_w to its predecessors
12 **do**
13 **if** $\rho_{uw}^{s_{uw}(\omega_k)} > -1$ **then** $t_u \leftarrow \max\left\{t_u, \frac{t_w - c_{uw}^{s_{uw}(\omega_k)}}{1+\rho_{uw}^{s_{uw}(\omega_k)}}\right\}$
14 **else**
15 **if** $c_{uw}^{s_{uw}(\omega_k)} \leq t_w$ **then** $\omega_v \leftarrow +\infty$, **go to** step 18 // $\tau_u(\omega_k)$ is reached before $\tau_v(\omega_k)$
16 **else continue** // t_w will not be reached with this arc, skip arc (u, w)
17 $\omega_v \leftarrow \max_{u \in V'} \{t_u - T_u(\omega_k)\}$
18 **return** ω_v

Algorithm 1: Back-propagation of $\tau_v(\omega_k)$.

The above shows that we can solve TDSPW by executing Algorithm 2. The construction of the sequence $\omega_1, \ldots, \omega_K$ is done by repeated executions of Algorithm 1 (step 4) so Corollary 3 ensures that the sequence satisfies the recursive definition of Theorem 3. According to Theorem 3 and Corollary 1, we then know that Proposition 1 applies, so the computation of $T_v(\omega_k), \forall v \in V$, can be done by executing a shortest path algorithm. This yields the main result of the article, stated in Theorem 4.

initialization: $k = 1, \omega_1 = 0, \omega_0 = -1, T_v(\omega_0) = +\infty \; \forall v \in V \setminus \{o\}, T_o(\omega_0) = 0$
1 **repeat**
2 | **for** $v \in V$ **do** compute $T_v(\omega_k)$ using a shortest path algorithm
3 | **for** $v \in V$ **do** compute $\tau_v(\omega_k)$
4 | **for** $v \in V$ **do** $\omega_v \leftarrow$ result of the back-propagation of $\tau_v(\omega_k)$ with Algorithm 1
5 | $\omega_{k+1} = \min\left(W, \min_{v \in V} \omega_v\right)$
6 | $k \leftarrow k + 1$
7 **until** $\omega_k = W$
8 **return** $T_v(W), \forall v \in V$

Algorithm 2: Computing $T_d(W)$.

Theorem 4. *Algorithm 2 solves* TDSPW *in* $\mathcal{O}(n \times R \times SPP(n, m))$.

Proof. The validity of the algorithm has already been justified in the discussion preceding the statement of the theorem. So we only prove the complexity result.

Corollary 3 shows that the sequence $\omega_1, \ldots, \omega_K$ is constructed as expected, so Theorem 3 justifies that K is at most equal to the total number of breakpoints in the arc cost functions. At each iteration, a shortest path algorithm is called once to compute $T_v(\omega_k), \forall v \in V$, and Algorithm 1 needs to be called n times. Furthermore, Algorithm 1 is essentially a time-dependent shortest path problem without waiting on a digraph, which can be solved in $SPP(n, m)$. □

5 Acknowledgements

We thank the referees for their constructive comments that have helped improving the presentation of the paper.

A Proof of Proposition 5

We prove the result by contradiction, so assume that there is $v \in V$ such that $\omega \mapsto T_v(\omega)$ is not concave on $[\omega_k, \omega_{k+1}]$. We have already observed that T_v is a continuous piecewise linear function, so it is not concave if and only if there is $\bar{\omega} \in]\omega_k, \omega_{k+1}[$ such that T_v is not differentiable at $\bar{\omega}$ and

$$\partial_- T_v(\bar{\omega}) < \partial_+ T_v(\bar{\omega}) \tag{18}$$

In particular, (18) involves that $\partial_- T_v(\bar{\omega}) < 0$, so $T_v^=(\bar{\omega}) = T_v(\bar{\omega})$. According to Proposition 4, this means that we can build $(p, \mathrm{w}) \in \mathcal{PW}_v(\bar{\omega})$ such that $C(p, \mathrm{w}) = T_v(\bar{\omega})$, $\sum_{\ell=1}^{\sigma(p)-1} \mathrm{w}_\ell = \bar{\omega}$, and, denoting $p = (\nu; v_1(= o), v_2, \ldots, v_{\sigma(p)} = v)$, $C(p(j), \mathrm{w}(j)) = T_{v_j}(\bar{\omega})$ for every saturated node v_j in (p, w). Let v_i be the first saturated node in (p, w). To exhibit the contradiction, we study the function

$$\begin{aligned} f : [-\mathrm{w}_i, W - \bar{\omega}] &\to \mathbb{R}^+ \\ \epsilon &\mapsto C\left(p, \mathrm{w} + \epsilon\delta(i)\right), \end{aligned}$$

where $\delta(i)$ is the i-th vector of the canonical basis of $\mathbb{R}^{\sigma(p)}$ ($\delta_i(i) = 1$ and $\delta_j(i) = 0$, for $j \neq i$). Stated otherwise, $f(\epsilon)$ is the cost of $(p, \mathrm{w} + \epsilon\delta(i)))$, the path-with-waits from o to v obtained from (p, w) by waiting an additional ϵ at node v_i. Since the arc cost functions are piecewise linear and continuous, so is f. As a consequence, we can compute the left and right partial derivative of f at 0 as

$$\begin{cases} \partial_- f(0) = \lim\limits_{\epsilon \to 0-} \dfrac{C(p, \mathrm{w} + \epsilon\delta(i)) - C(p, \mathrm{w})}{\epsilon} = \lim\limits_{\epsilon \to 0-} \dfrac{C(p, \mathrm{w} + \epsilon\delta(i)) - T_v(\bar{\omega})}{\epsilon} \\ \partial_+ f(0) = \lim\limits_{\epsilon \to 0+} \dfrac{C(p, \mathrm{w} + \epsilon\delta(i)) - C(p, \mathrm{w})}{\epsilon} = \lim\limits_{\epsilon \to 0+} \dfrac{C(p, \mathrm{w} + \epsilon\delta(i)) - T_v(\bar{\omega})}{\epsilon} \end{cases}$$

Observing that $(p, \mathrm{w} + \epsilon\delta(i))$ is a path-with-waits from o to v with total wait $\bar{\omega} + \epsilon$, we get $C(p, \mathrm{w} + \epsilon\delta(i)) \geq T_v(\bar{\omega} + \epsilon)$ which yields

$$\begin{cases} \partial_- f(0) \leq \lim\limits_{\epsilon \to 0-} \dfrac{T_v(\bar{\omega} + \epsilon) - T_v(\bar{\omega})}{\epsilon} = \partial_- T_v(\bar{\omega}) \\ \partial_+ f(0) \geq \lim\limits_{\epsilon \to 0+} \dfrac{T_v(\bar{\omega} + \epsilon) - T_v(\bar{\omega})}{\epsilon} = \partial_+ T_v(\bar{\omega}) \end{cases}$$

Together with (18), we get

$$\partial_- f(0) \leq \partial_- T_v(\bar{\omega}) < \partial_+ T_v(\bar{\omega}) \leq \partial_+ f(0),$$

which implies in particular that f is non-differentiable at 0.

By definition, $f(\epsilon)$ is the sum of composites of the piecewise-linear arc cost functions $C_{v_j v_{j+1}}, j = 1, \ldots, \sigma(p) - 1$. For $j = 1, \ldots, i$ the term related to $C_{v_j v_{j+1}}$ is taken at $C(p(j), \mathrm{w}(j)) + \sum_{k=1}^{j-1} \mathrm{w}_k$, which does not depend on ϵ. For $j = i+1, \ldots, \sigma(p) - 1$ the term related to $C_{v_j v_{j+1}}$ is taken at $C(p(j), \mathrm{w}(j) + \epsilon\delta(i)) + \bar{\omega} + \epsilon$. Let v_j be a saturated node ($j \in \{i, \ldots, \sigma(p)\}$): using that $C(p(j), \mathrm{w}(j)) = T_{v_j}(\bar{\omega})$ and $\omega_k \leq \bar{\omega} \leq \omega_{k+1}$, we get $\tau^-_{v_j}(\omega_k) \leq C(p(j), \mathrm{w}(j)) + \bar{\omega} \leq \tau_{v_j}(\omega_k)$. Recalling that $C_{v_j v_{j+1}}$ is differentiable on $]\tau^-_{v_j}(\omega_k), \tau_{v_j}(\omega_k)[$, for all $(v_j, v_{j+1}) \in A$, we get that f is non-differentiable at $\epsilon = 0$ only if there is a saturated node v_j such that the piece where $C_{v_j v_{j+1}}$ is evaluated changes at $\epsilon = 0$, i.e.,

i. $T_{v_j}(\bar{\omega}) + \bar{\omega} = \tau^-_{v_j}(\omega_k)$ and $\exists \alpha > 0 : C(p(j), \mathrm{w}(j) - \nu\delta(i)) + \bar{\omega} - \nu < \tau^-_{v_j}(\omega_k)$ for all $0 < \nu < \alpha$; or

ii. $T_{v_j}(\bar{\omega}) + \bar{\omega} = \tau_{v_j}(\omega_k)$ and $\exists \alpha > 0 : C(p(j), \mathrm{w}(j) + \nu\delta(i)) + \bar{\omega} + \nu > \tau_{v_j}(\omega_k)$ for all $0 < \nu < \alpha$.

If $T_{v_j}(\bar{\omega}) + \bar{\omega} = \tau^-_{v_j}(\omega_k)$, $\tau^-_{v_j}(\omega_k) \leq T_{v_j}(\omega_k) + \omega_k$ yields $T_{v_j}(\bar{\omega}) + \bar{\omega} = T_{v_j}(\omega_k) + \omega_k$, hence $\partial_- T_{v_j}(\bar{\omega}) = -1$. Function T_{v_j} is piecewise linear, so there is $\alpha > 0$ such that for all $0 < \nu < \alpha$, $T_{v_j}(\bar{\omega} - \nu) = T_{v_j}(\bar{\omega}) + \nu$. Using that $C(p(j), \mathrm{w}(j) - \nu\delta(i)) \geq T_{v_j}(\bar{\omega} - \nu)$, we obtain

$$C(p(j), \mathrm{w}(j) - \nu\delta(i)) + \bar{\omega} - \nu \geq T_{v_j}(\bar{\omega}) + \bar{\omega} \geq \tau^-_{v_j}(\omega_k),$$

so item i. is never true.

If $T_{v_j}(\bar{\omega}) + \bar{\omega} = \tau_{v_j}(\omega_k)$, then $T_{v_j}(\bar{\omega}) + \bar{\omega} = T_{v_j}(\omega_{k+1}) + \omega_{k+1}$ by (14). A path-with-waits from o to v_j that realizes $T_{v_j}(\omega_{k+1})$ with total wait ω_{k+1} will then arrive at $T_{v_j}(\omega_{k+1}) + \omega_{k+1} = T_{v_j}(\bar{\omega}) + \bar{\omega}$. So if we follow p from there without additional waits, v will also be reached at $T_v(\bar{\omega}) + \bar{\omega}$, just like in (p, w). Denoting this new path-with-waits as (p', w'), we get that $C(p', \mathrm{w}') = T_v(\bar{\omega}) + \bar{\omega} - \omega_{k+1}$. Finally, $C(p', \mathrm{w}') \geq T_v(\omega_{k+1})$, hence $T_v(\omega_{k+1}) + \omega_{k+1} \leq T_v(\bar{\omega}) + \bar{\omega}$. This is only possible if $\partial_+ T_v(\omega) = -1$ for all $\omega \in [\bar{\omega}, \omega_{k+1}]$, which is in contradiction with $\partial_- T_v(\bar{\omega}) < \partial_+ T_v(\bar{\omega})$. We conclude that item ii. is never true either, hence f is differentiable at 0: a contradiction.

B Proof of Proposition 6

Let $k \in \{1, \dots, K-1\}$, and $v \in \arg\min_{u \in V} \{\max_{\omega \in \Omega}\{\omega : T_u(\omega) + \omega \leq \tau_u(\omega_k)\}\}$. If $\tau_v(\omega_k) = +\infty$, then both (15) and (16) yield $\omega_{k+1} = W$, proving the desired equality.

Thus, we suppose next that $\tau_v(\omega_k) < +\infty$ and define

$$\widetilde{\omega} = \max_{\omega \in \Omega}\{\omega : T_v(\omega) + \omega \leq \tau_v(\omega_k)\}. \tag{19}$$

Theorem 3 states that we are in the conditions of application of Corollaries 1 and 2. Using Corollary 1, we get

$$T_v(\widetilde{\omega}) = \min\left\{T_v(\omega_k), T_v^{=}(\widetilde{\omega})\right\}.$$

1. If $T_v(\widetilde{\omega}) = T_v(\omega_k)$, we use that $t_{v\to v}(\omega_k) = \tau_v(\omega_k) = T_v(\widetilde{\omega}) + \widetilde{\omega}$ (by definition of v and $t_{v\to v}$) to observe that
$$\widetilde{\omega} = t_{v\to v}(\omega_k) - T_v(\omega_k).$$

2. Otherwise, $T_v(\widetilde{\omega}) = T_v^{=}(\widetilde{\omega})$, so by Corollary 2, there are $v_i \in V$ and a path p from v_i to v such that $t_p(v_i) = T_{v_i}(\omega_k) + \widetilde{\omega}$, $t_p(v_j) = T_{v_j}(\widetilde{\omega}) + \widetilde{\omega}, \forall v_j \in p, v_j \notin \{v_i, v\}$ and $T_{v_i}(\omega_k) + C(p) = T_v(\widetilde{\omega})$. By definition of $\widetilde{\omega}$, $T_w(\widetilde{\omega}) + \widetilde{\omega} \leq \tau_w(\omega_k), \forall w \in p$. Since $\omega \mapsto T_w(\omega) + \omega$ is non-decreasing for all $w \in V$, we see that p reaches v at $\tau_v(\omega_k)$ and every intermediary node v_j in the time interval $[T_{v_j}(\omega_k) + \omega_k, \tau_{v_j}(\omega_k)]$. Stated otherwise, $p \in \mathcal{P}^{\text{back}}_{v_j \to v}(\omega_k)$. This means in particular that
$$\tau_v(\omega_k) - \min\{T_u(\omega_k) + C(p) : u \in V, p \in \mathcal{P}^{\text{back}}_{u\to v}(\omega_k)\} \geq \tau_v(\omega_k) - T_v(\widetilde{\omega}) = \widetilde{\omega}.$$

By definition of $t_{u\to v}(\omega_k)$, we get that in both cases above:

$$\widetilde{\omega} \leq \max_{u \in V}\{t_{u\to v}(\omega_k) - T_u(\omega_k)\}. \tag{20}$$

To show the reverse equality, let $u \in V$ and $p \in \mathcal{P}^{\text{back}}_{u\to v}(\omega_k)$, and let $\omega = \tau_v(\omega_k) - (T_u(\omega_k) + C(p))$ (notice $\omega \geq \omega_k$ by definition of $\mathcal{P}^{\text{back}}_{u\to v}(\omega_k)$). Then $T_u(\omega_k) + C(p)$ is the the cost of a path-with-waits from o to v that reaches u at $T_u(\omega_k)$ and v at $\tau_v(\omega_k)$ with a total wait equal to ω. We thus get that $T_v(\omega) + \omega \leq \tau_v(\omega_k)$. Using (19), this implies that $\omega \leq \widetilde{\omega}$, and since the above is valid for any u and any $p \in \mathcal{P}^{\text{back}}_{u\to v}(\omega_k)$:

$$\begin{aligned}\widetilde{\omega} &\geq \tau_v(\omega_k) - \min\left\{(T_u(\omega_k) + C(p)) : u \in V,\, p \in \mathcal{P}^{\text{back}}_{u\to v}(\omega_k)\right\} \\ &= \max_{u \in V}\{t_{u\to v}(\omega_k) - T_u(\omega_k)\}.\end{aligned} \tag{21}$$

Combining (20) and (21) and yields

$$\widetilde{\omega} = \max_{u \in V}\{t_{u\to v}(\omega_k) - T_u(\omega_k)\}.$$

Finally, we can rely on (19) to see that the above actually proves that

$$\max_{\omega \in \Omega}\{\omega : T_v(\omega) + \omega \leq \tau_v(\omega_k)\} = \max_{u \in V}\{t_{u\to v}(\omega_k) - T_u(\omega_k)\},$$

proving the result whenever $\tau_v(\omega_k) < +\infty$.

C Proof of Lemma 1

Before proving the lemma, we introduce the following technical result that will be instrumental in proving Lemma 1.

Lemma 2. *Consider the node w selected at an execution of step 7 of Algorithm 1, i.e., $w \in \arg\max_{u \in V' \setminus S} \{t_u\}$. We have that*

1. *if $T_w(\omega_k) + \omega_k \leq t_w \leq \tau_w(\omega_k)$, $t_w = t_{w \to v}(\omega_k)$;*
2. *if $t_w < T_w(\omega_k) + \omega_k$, then $\mathcal{P}^{\text{back}}_{w \to v}(\omega_k) = \emptyset$;*
3. *if $t_w > \tau_w(\omega_k)$, then $\tau_w(\omega_k) < \tau_v(\omega_k)$ and*

$$\max_{\omega \in \Omega} \{\omega : T_w(\omega) + \omega \leq \tau_w(\omega_k)\} \leq \max_{\omega \in \Omega} \{\omega : T_v(\omega) + \omega \leq \tau_v(\omega_k)\}.$$

4. *if there is $(u, w) \in A$ such that $\rho_{uw}^{s_{uw}(\omega_k)} = -1$ and $c_{uw}^{s_{uw}(\omega_k)} \leq t_w$, then $\tau_u(\omega_k) < \tau_v(\omega_k)$ and*

$$\max_{\omega \in \Omega} \{\omega : T_u(\omega) + \omega \leq \tau_u(\omega_k)\} \leq \max_{\omega \in \Omega} \{\omega : T_v(\omega) + \omega \leq \tau_v(\omega_k)\}.$$

Proof. One can verify that the first three items are satisfied if the conditions checked at steps 8 and 9 have been false since the beginning of the execution of Algorithm 1. Indeed, in such case, the algorithm is a plain back-propagation for the computation of shortest paths with affine arc costs.

The complete proof of items 1. and 2. is by induction on the number of executions of step 7. At the first execution of this step, $w = v$, so $t_w = \tau_v(\omega_k) = t_{v \to v}(\omega_k)$ by definition. Hence items 1. and 2. are true.

We now consider a later execution of step 7, where $w \in \arg\max_{u \in V' \setminus S} \{t_u\}$, assuming that the two items hold at each previous iteration. For every node u selected at a previous execution of step 7, the definition of the algorithm involves that $u \in S$ if $T_u(\omega_k) + \omega_k \leq t_u \leq \tau_u(\omega_k)$ and $u \in V \setminus V'$ if $t_u < T_u(\omega_k) + \omega_k$. This means that for all marked nodes $u \in S$, $t_u = t_{u \to v}(\omega_k)$, and that no path in $\mathcal{P}^{\text{back}}_{u \to v}(\omega_k)$ goes through a node of $V \setminus V'$. Moreover, $t_u > \tau_u(\omega_k)$ did not happen, otherwise the algorithm would have been terminated.

To show item 1., observe that we would have obtained the same value for t_w if the back-propagation had been run on the subgraph induced by V'. Moreover, the nodes of $V \setminus V'$ are not involved in the paths of $\mathcal{P}^{\text{back}}_{w \to v}(\omega_k)$, so $t_{w \to v}(\omega_k)$ can be computed by considering the subgraph induced by V'. In this subgraph, the conditions checked at steps 8-9 would have been false at every previous iteration, so if $T_w(\omega_k) + \omega_k \leq t_w \leq \tau_w(\omega_k)$, t_w is the result of a classical back-propagation. As a consequence, $t_w = t_{w \to v}(\omega_k)$.

The proof of item 2. is by contradiction. Assume that $t_w < T_w(\omega_k) + \omega_k$ and that there is $p \in \mathcal{P}^{\text{back}}_{w \to v}(\omega_k) \neq \emptyset$. Then let $p = (t_p; v_1 (= w), v_2, \ldots, v_{\sigma(p)} (= v))$ be the shortest path in $\mathcal{P}^{\text{back}}_{w \to v}(\omega_k)$. If every node in p is marked, then let $i = 1$. Otherwise, let v_i be the first non-marked node in p, i.e., $v_i \notin S$ and $v_j \in S$ for $j \in \{i+1, \ldots, \sigma(p)\}$. Since p is a shortest path, then $t_p(v_j) = t_{v_j \to v}(\omega_k)$ for every marked node $v_j \neq w$ of p. Moreover, the back-propagation along the arcs of v_{i+1} ensures that $t_{v_i} \geq t_p(v_i)$. By definition, $w \in \arg\max_{u \in V' \setminus S} \{t_u\}$, so $t_w \geq t_{v_i} \geq t_p(v_i)$. And given that p goes through w before v_i, $t_p(w) \leq t_p(v_i)$. As a consequence, $t_p(w) \leq t_w < T_w(\omega_k) + \omega_k$, which is in contradiction with the definition of $\mathcal{P}^{\text{back}}_{w \to v}(\omega_k)$.

To prove item 3., assume that $t_w > \tau_w(\omega_k)$ at some execution of step 7. Let p be the path from w to v constructed by the back-propagation. First we have, by non-negativity of arc travel times, $t_w \leq \tau_v(\omega_k)$, so

$$\tau_w(\omega_k) < \tau_v(\omega_k).$$

Then, denote as v^+ the successor of w in p. Let $\omega' = \max_\omega\{T_w(\omega) + \omega \leq \tau_w(\omega_k)\}$ and let $(p^w, \mathrm{w}^w) \in \mathcal{PW}_w(\omega')$ be a path-with-wait from o to w that reaches w at $\tau_w(\omega_k) < t_w$ with a total wait equal to ω'. Taking arc (w, v^+) immediately after p^w (no wait at w), v^+ is reached at

$$t^+ = \tau_w(\omega_k) + \rho_{wv^+}^{s_{wv^+}(\omega_k)}\tau_w(\omega_k) + c_{wv^+}^{s_{wv^+}(\omega_k)}.$$

By $\tau_w(\omega_k) < t_w$, we know that v^+ is reached earlier in this path than in p, i.e., $t^+ \leq t_{v^+}$. If we then wait $t_{v^+} - t^+$ at v^+, we can take the end of p from v^+ to reach v at $\tau_v(\omega_k)$. Now, if (p', w') is the path-with-wait from o to v constructed above, we get that (p', w') reaches v at $\tau_v(\omega_k)$ with a total wait

$$\omega_{(p',\mathrm{w}')} = \omega' + t_{v^+} - t^+ \geq \omega'. \tag{22}$$

Stated otherwise, we have $\tau_v(\omega_k) = C(p', \mathrm{w}') + \omega_{(p',\mathrm{w}')}$ where $C(p', \mathrm{w}') \geq T_v(\omega_{(p',\mathrm{w}')})$, so

$$T_v(\omega_{(p',\mathrm{w}')}) + \omega_{(p',\mathrm{w}')} \leq \tau_v(\omega_k).$$

This yields that

$$\max_{\omega\in\Omega}\{\omega : T_v(\omega) + \omega \leq \tau_v(\omega_k)\} \geq \omega_{(p',\mathrm{w}')} \geq \omega' = \max_{\omega\in\Omega}\{\omega : T_w(\omega) + \omega \leq \tau_w(\omega_k)\}.$$

To prove item 4., first observe that if we are back-propagating the value of t_w, this means that $T_w(\omega_k) + \omega_k \leq t_w \leq \tau_w(\omega_k)$, hence $t_w = t_{w\to v}(\omega_k)$. Now let $(u, w) \in A$ such that $\rho_{uw}^{s_{uw}(\omega_k)} = -1$ and $c_{uw}^{s_{uw}(\omega_k)} \leq t_w$. Let $\omega_u = \max_{\omega\in\Omega}\{\omega : T_u(\omega) + \omega \leq \tau_u(\omega_k)\}$. There is a path-with-wait $(p_u, \mathrm{w}_u) \in \mathcal{PW}_u(\omega_u)$ such that $C(p_u, \mathrm{w}_u) = T_u(\omega_u)$. For any path-with-wait (p, w) that goes though arc (u, w), if $\tau_u^-(\omega_k) \leq t_{p,\mathrm{w}}(u) \leq \tau_u(\omega_k)$, then by definition of (u, w), $t_{p,\mathrm{w}}(w) = c_{uw}^{s_{uw}(\omega_k)} \leq t_w$. As a consequence, we can extend (p_u, w_u) by

- departing from u at $T_u(\omega_u) + \omega_u = \tau_u(\omega_k)$,
- waiting an additional $\omega' = t_w - c_{uw}^{s_{uw}(\omega_k)}$ at w,
- and taking the shortest path in $\mathcal{P}_{w\to v}^{\mathrm{back}}(\omega_k)$ to reach v at $\tau_v(\omega_k)$.

Denoting this path-with-wait as (p_v, w_v), we see that it accumulates a total wait $\omega_v = \omega_u + \omega' \geq \omega_u$. Given that (p_v, w_v) reaches v at $\tau_v(\omega_k)$, we have $T_v(\omega_v) + \omega_v \leq C(p_v, \mathrm{w}_v) + \omega_v = \tau_v(\omega_k)$. As a conclusion,

$$\max_{\omega\in\Omega}\{\omega : T_v(\omega) + \omega \leq \tau_v(\omega_k)\} \geq \omega_v \geq \omega_u = \max_{\omega\in\Omega}\{\omega : T_u(\omega) + \omega \leq \tau_u(\omega_k)\}.$$

□

We are now ready to prove Lemma 1.

Proof of Lemma 1. To prove item 1., let $v^* \in \arg\min_{v\in V(\omega_k)}\{\tau_v(\omega_k)\}$. We then consider the back-propagation of $\tau_{v^*}(\omega_k)$ by Algorithm 1. By contraposition of item 3. of Lemma 2, we know that

for all w selected at step 7 of Algorithm 1, $t_w \leq \tau_w(\omega_k)$. As a consequence, items 1. and 2. of Lemma 2 guarantee that at step 17 of Algorithm 1

$$\max_{u \in V'} \{t_u - T_u(\omega_k)\} = \max_{u \in V} \{t_{u \to v^*}(\omega_k) - T_u(\omega_k)\}.$$

We now prove that $\omega_v \geq \omega_{v^*}, \forall v \in V$. We thus consider $v \in V$ such that $\omega_v < +\infty$. By definition of ω_{v^*}, $T_v(\omega_{v^*}) + \omega_{v^*} \leq \tau_v(\omega_k)$ for each $v \in V$, so the application of Corollary 1 yields

$$T_v(\omega_{v^*}) = \min\{T_v(\omega_k), T_v^{=}(\omega_{v^*})\}.$$

At step 4 of Algorithm 1, we initialize t_v to $\tau_v(\omega_k)$, and this value is not modified in the rest of the algorithm, hence $\omega_v \geq \tau_v(\omega_k) - T_v(\omega_k)$. As a consequence,

$$T_v(\omega_{v^*}) = T_v(\omega_k) \implies \omega_v \geq \tau_v(\omega_k) - T_v(\omega_{v^*}) \geq \omega_{v^*}.$$

Now, assume that $T_v(\omega_{v^*}) = T_v^{=}(\omega_{v^*})$, and let (p, w) be a path-with-waits constructed as in Corollary 2 so that

- $C(p, \mathrm{w}) = T_v(\omega_{v^*})$ and $\sum_{\ell=1}^{\sigma(p)-1} \mathrm{w}_\ell = \omega_{v^*}$;
- $C(p(i), \mathrm{w}(i)) = T_{v_i}(\omega_k)$, where v_i is the first saturated node in (p, w);
- and $C(p(j), \mathrm{w}(j)) = T_{v_j}(\omega_{v^*})$ for every other saturated node v_j.

In particular, (p, w) connects v_i to v without waits (after v_i) with a cost equal to $T_v(\omega_{v^*}) - T_{v_i}(\omega_k)$. We claim that the subpath of p leaving from v_i, $\bar{p}(i)$, is in $\mathcal{P}^{\mathrm{back}}_{v_i \to v}(\omega_k)$. By definition of $t_{v_i \to v}(\omega_k)$, $\tau_v(\omega_k) - t_{v_i \to v}(\omega_k)$ is the minimum cost of a path in $\mathcal{P}^{\mathrm{back}}_{v_i \to v}(\omega_k)$, so the claim yields

$$\tau_v(\omega_k) - t_{v_i \to v}(\omega_k) \leq T_v(\omega_{v^*}) - T_{v_i}(\omega_k).$$

Moreover, if $\omega_v < +\infty$, then Lemma 2 yields that $t_u = t_{u \to v}(\omega_k)$ for all $v \in V'$ and $\mathcal{P}^{\mathrm{back}}_{u \to v}(\omega_k) = \emptyset$ for $u \in V \setminus V'$. As a consequence, $\omega_v = \max_{u \in V} \{t_{u \to v}(\omega_k) - T_u(\omega_k)\}$. In particular, this means that

$$\omega_v \geq t_{v_i \to v}(\omega_k) - T_{v_i}(\omega_k) \geq \tau_v(\omega_k) - T_v(\omega_{v^*}) \geq \omega_{v^*},$$

where the last inequality follows from $\tau_v(\omega_k) \geq T_v(\omega_{v^*}) + \omega_{v^*}$.

To prove the claim, observe that (p, w) reaches v at $T_v(\omega_{v^*}) + \omega_{v^*}$, so if we back-propagate $T_v(\omega_{v^*}) + \omega_{v^*}$ from v to v_i along $\bar{p}(i)$, we reach every intermediary node v_j at $C(p(j), \mathrm{w}(j)) = T_{v_j}(\omega_{v^*}) + \omega_{v^*}$. Using that $\tau_v(\omega_k) \geq T_v(\omega_{v^*}) + \omega_{v^*}$, we observe that if we back-propagate $\tau_v(\omega_k)$ from v to v_i along $\bar{p}(i)$, we necessarily reach the intermediary nodes v_j later than when back-propagating $T_v(\omega_{v^*}) + \omega_{v^*}$, i.e., later than $T_{v_j}(\omega_{v^*}) + \omega_{v^*}$. Moreover, if one intermediary node is reached later than $\tau_{v_j}(\omega_k)$, then Algorithm 1 necessarily stops at step 8, which is not possible if $\omega_v < +\infty$. By $T_{v_j}(\omega_{v^*}) + \omega_{v^*} \geq T_{v_j}(\omega_k) + \omega_k$, we get that every intermediary node v_j is also reached in the time interval $[T_{v_j}(\omega_k) + \omega_k, \tau_{v_j}(\omega_k)]$ when back-propagating $\tau_{\bar{v}}(\omega_k)$ from v to v_i along $\bar{p}(i)$. □

References

[1] R. K. Ahuja, J. B. Orlin, S. Pallottino, and M. Grazia Scutellà. Minimum time and minimum cost-path problems in street networks with periodic traffic lights. *Transportation Science*, 36(3):326–336, 2002.

[2] R. K. Ahuja, J. B. Orlin, S. Pallottino, and M. G. Scutella. Dynamic shortest paths minimizing travel times and costs. *Networks*, 41(4):197–205, 2003.

[3] R. Bellman. On a routing problem. *Quarterly of applied mathematics*, 16(1):87–90, 1958.

[4] X. Cai, T. Kloks, and C.-K. Wong. Time-varying shortest path problems with constraints. *Networks*, 29(3):141–150, 1997.

[5] I. Chabini and B. Dean. Shortest path problems in discrete-time dynamic networks: complexity, algorithms and implementations. *Unpublished manuscript*, 1999.

[6] K. L. Cooke and E. Halsey. The shortest route through a network with time-dependent internodal transit times. *Journal of Mathematical Analysis and Applications*, 14(3):493 – 498, 1966.

[7] B. C. Dean. Algorithms for minimum-cost paths in time-dependent networks with waiting policies. *Networks*, 44(1):41–46, 2004.

[8] M. Dell'Amico, M. Iori, and D. Pretolani. Shortest paths in piecewise continuous time-dependent networks. *Operations Research Letters*, 36(6):688–691, 2008.

[9] S. E. Dreyfus. An appraisal of some shortest-path algorithms. *Operations Research*, 17(3):395–412, 1969.

[10] L. Foschini, J. Hershberger, and S. Suri. On the complexity of time-dependent shortest paths. *Algorithmica*, 68(4):1075–1097, 2014.

[11] E. He, N. Boland, G. Nemhauser, and M. Savelsbergh. Computational complexity of time-dependent shortest path problems. *Optimization Online (Feb. 2019)*, 12, 2019.

[12] E. He, N. Boland, G. Nemhauser, and M. Savelsbergh. Dynamic discretization discovery algorithms for time-dependent shortest path problems. *Optimization Online*, 7082, 2019.

[13] O. Jabali, T. Van Woensel, and A. De Kok. Analysis of travel times and co2 emissions in time-dependent vehicle routing. *Production and Operations Management*, 21(6):1060–1074, 2012.

[14] D. E. Kaufman and R. L. Smith. Fastest paths in time-dependent networks for intelligent vehicle-highway systems application. *I V H S Journal*, 1(1):1–11, 1993.

[15] J. Omer and M. Poss. Time-dependent shortest path with discounted waits. *Networks*, 74(3):287–301, 2019.

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

[17] A. Orda and R. Rom. Minimum weight paths in time-dependent networks. *Networks*, 21(3):295–319, 1991.