WORKING PAPER L-2006-11

WP

Daniele Pretolani, Lars Relund Nielsen &
Kim Allan Andersen

A note on "Multicriteria adaptive paths in stochastic, time-varying networks"

Logistics/SCM
Research Group

A note on "Multicriteria adaptive paths in stochastic, time-varying networks"

Daniele Pretolani*
Department of Sciences and Methods of Engineering
University of Modena and Reggio Emilia
Via Amendola 2
I-42100 Reggio Emilia
Italy

Lars Relund Nielsen†
Research Unit of Statistics and Decision Analysis
DIAS
P.O. Box 50
DK-8830 Tjele
Denmark

Kim Allan Andersen‡
Department of Business Studies
Aarhus School of Business
Fuglesangs Allé 4
DK-8210 Aarhus V
Denmark

November 17, 2006

Abstract

In a recent paper, Opasanon and Miller-Hooks study multicriteria adaptive paths in stochastic time-varying networks. They propose a label correcting algorithm for finding the full set of efficient strategies. In this note we show that their algorithm is not correct, since it is based on a property that does not hold in general.

Opasanon and Miller-Hooks also propose an algorithm for solving a parametric problem. We give a simplified algorithm which is linear in the input size.

Keywords: Multiple objective programming; shortest paths; stochastic time-dependent networks; time-adaptive strategies.

1 Introduction

In this note we consider stochastic time-varying networks (*STV networks*, also known as stochastic or random time-dependent networks) where the arcs carry multiple attributes. In particular, we address some incorrect results contained in a recent paper by Opasanon and Miller-Hooks [5].

In STV networks travel times are modelled as random variables with time-dependent distributions. STV networks were first addressed by Hall [2], who showed that the best route between two nodes is not necessarily a path, but rather a time-adaptive *strategy* that assigns optimal successor arcs to a node as a function of departure time. A detailed review of the literature on the subject can be found in the recent paper by Gao and Chabini [1], where time-adaptive route choice is considered in a more general framework, that takes into account several variants of online information.

*Corresponding author, e-mail: pretolani.daniele@unimore.it.
†e-mail: lars@relund.dk.
‡e-mail: kia@asb.dk.

The above works only consider a single objective. Nevertheless, due to the multi-objective nature of many transportation and routing problems, a single objective function is not sufficient to completely characterize most real-life problems. Several attributes such as time, cost and incident rate may be of interest. If this is the case the solution will be a set of *efficient* (Pareto optimal) strategies. Finding the efficient set is well known to be NP-hard also in deterministic networks.

Previous work on the multi-objective case has been focused on *discrete* STV networks, where travel times have integer-valued discrete distributions. Nielsen [3] and Nielsen, Andersen, and Pretolani [4] consider the bicriterion case, and propose algorithms based on the two-phase method for finding the set of efficient strategies, as well as fast heuristic methods providing reasonable approximations. Opasanon and Miller-Hooks [5] consider the problem for an arbitrary number of criteria. After pointing out some relevant properties of Pareto-optimal strategies, they propose a label correcting algorithm (*Algorithm APS*) for finding the set of efficient strategies. Moreover, since the generation of all such efficient strategies may require enormous computational effort, they devise two algorithms that find a single strategy by minimizing the expectation of a weighted sum of the criteria (referred to as "disutility"). In fact, this approach extends to $r > 2$ criteria, the *parametric problem* considered in Nielsen [3] and Nielsen et al. [4].

We show that the work of Opasanon and Miller-Hooks [5] has some technical flaws, in particular, one of the properties in their paper (Lemma 1) does not hold in general. As a result, the proof of correctness of Algorithm APS is no longer valid, indeed, we provide an example where the algorithm fails. More precisely, we show that APS may fail to generate all efficient strategies, and may generate labels not corresponding to strategies. We also point out that the model of STV network proposed by Opasanon and Miller-Hooks [5] is unnecessarily complicated and can be simplified. As a consequence, we obtain an improved algorithm for the parametric problem, whose complexity is linear in the size of the input. Note that this algorithm is essentially the same proposed in [4, 3] for the bicriterion case.

For easiness of comparison in this paper, we adopt the notation from [5]; however, we skip most of the definitions here, showing their application to a running example instead. To this purpose we adopt the representation of an STV network as a time-expanded hypergraph, see Pretolani [6]; however, we use this representation only as a graphical tool, without any theoretical development.

In the next section we introduce the STV network model adopted by Opasanon and Miller-Hooks [5], and we show a counterexample to their Lemma 1. In Section 3 we describe Algorithm ASP, showing that it fails on the counterexample. In Section 4 we show a simplified network model and an improved parametric algorithm. Final observations and motivations for further research are briefly discussed in the last section.

2 STV networks, strategies, and dominance

Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a directed graph, referred to as the *topological network*. Arrival and departure times to/from nodes are integer in the interval $[0, I]$, which corresponds to a *peak period* $[t_0, t_0 + I\Delta t]$ discretized into time intervals of length Δt.

We consider a set $R = \{1, 2, \ldots, r\}$ of $r \geq 2$ criteria, thus with each arc we associate r probability distributions. For each arc $(i, j) \in \mathcal{A}$, $t \in [0, I]$ and $k \in R$, $C^k = \{c_{ij}^{kz_k}(t) : \ z_k = 1, \ldots, D\}$ denotes the set of possible non-negative values for criterion k when travelling along arc (i, j) at departure time t. Each value $c_{ij}^{kz_k}(t)$ occurs with probability $\rho_{ij}^{kz_k}(t)$, thus $\sum_{z_k=1,\ldots,D} \rho_{ij}^{kz_k}(t) = 1$. It is assumed that travel times are identified with criterion 1; therefore, the set $A_{ij}(t) = \{t + c_{ij}^{1z_1}(t) :$ $z_1 = 1, \ldots, D\}$ contains the possible arrival times at node j when leaving node i at time t along arc (i, j). Waiting at nodes is not permitted.

Note that the above model, adopted in [5], is unnecessarily involved; as we shall show later, it suffices to consider a single possible value (i.e., $D = 1$) for each criterion other than travel time. In the rest of this section we assume that for each arc $(i, j) \in \mathcal{A}$, $t \in [0, I]$ and criterion $k > 1$ we have a single possible value, denoted by $c_{ij}^k(t)$. Note that this is the model adopted in [6] in the case of a single cost criterion.

$(i,j),t$	$(o,a),0$	$(a,b),1$	$(a,b),2$	$(b,d),2$	$(b,d),3$	$(b,c),2$	$(b,c),3$	$(c,d),4$
C^1	$\{1,2\}$	$\{1,2\}$	$\{1\}$	$\{2\}$	$\{9\}$	$\{2\}$	$\{1\}$	$\{4\}$
$A_{ij}(t)$	$\{1,2\}$	$\{2,3\}$	$\{3\}$	$\{4\}$	$\{12\}$	$\{4\}$	$\{4\}$	$\{8\}$

Table 1: Travel times and arrival times

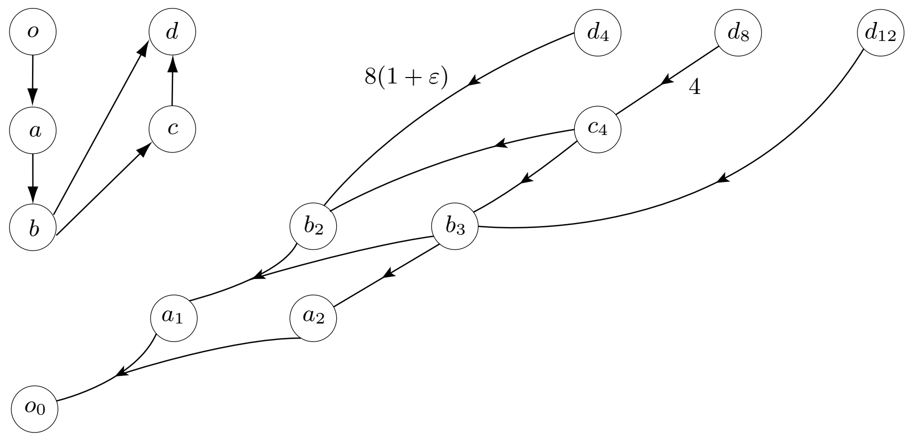


Figure 1: Topological network and time-expanded hypergraph

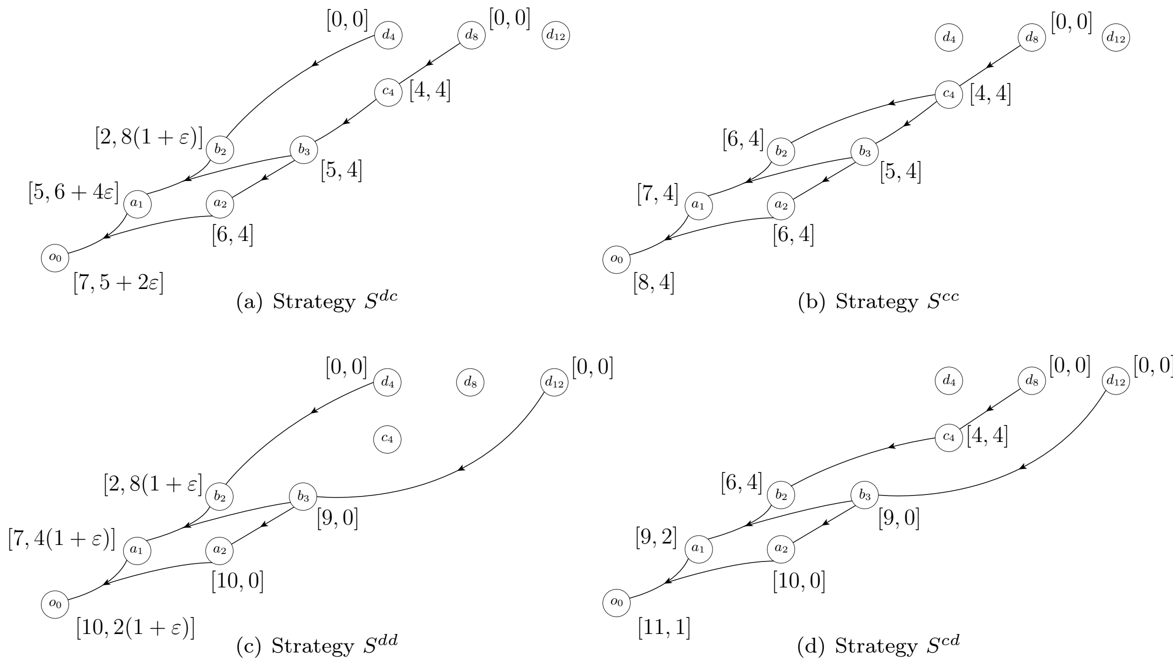
Example 1 Consider the topological network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ shown in the top left corner of Figure 1. We assume that a traveller leaves the origin node o at time zero towards the destination node d, thus we only consider departure times corresponding to the traveller's possible arrival times at intermediate nodes. For each arc (i,j) and relevant time t, the set C^1 of travel times and the set $A_{ij}(t)$ of arrival times are given in Table 1.

We only have two non-deterministic travel times, namely arc (o,a) at departure time 0 and arc (a,b) at departure time 1; in both cases we assume that arrival times have the same probability 1/2, that is $\rho_{oa}^{1,1} = \rho_{oa}^{1,2} = \rho_{ab}^{1,1} = \rho_{ab}^{1,2} = 1/2$. Note that routing decisions are needed (actually, possible) only at node b.

We represent the STV network by means of a *time-expanded hypergraph*, as shown in Figure 1. For each node $i \in G$ and time t we introduce a hypergraph node i_t; for each arc (i,j) and departure time t we introduce a *hyperarc* $e_{ij}(t)$ that connects node i_t to the set $\{j_\theta : \theta \in A_{ij}(t)\}$ of hypergraph nodes corresponding to possible arrival times at j. For example, hyperarc $e_{ab}(1)$ connects node a_1 to the node set $\{b_2, b_3\}$, since $A_{ab}(1) = \{2,3\}$.

We assume $r = 2$, and we refer to criterion 2 as *cost*; the cost is zero for each arc and departure time, except for the two cases shown in Figure 1, namely: arc (c,d) at departure time 4, with cost $c_{cd}^2(4) = 4$, and arc (b,d) at departure time 2, with cost $c_{bd}^2(2) = 8(1+\varepsilon)$, where we assume $0 < \varepsilon < 1$. □

According to the routing model proposed by Hall [2] a time-adaptive strategy assigns to each node a successor arc as a function of departure time. In a discrete STV network, a strategy is defined by choosing a single successor arc for each node $i \neq d$ and time t. Each strategy associates with each node i, time t and $k \in R$ the expected value of criterion k for travelling from i to the destination at departure time t. Given a strategy, the corresponding expected values can be formally defined by means of a set of recursive equations, see e.g. Pretolani [6]. In practice, the computation of these values consists in a labelling process such as the one described by Opasanon and Miller-Hooks [5]. We illustrate this process by applying it to the strategies in our running example.



(a) Strategy S^{dc} (b) Strategy S^{cc} (c) Strategy S^{dd} (d) Strategy S^{cd}

Figure 2: Strategies and corresponding time-cost pairs.

Example 1 (continued) In order to define a strategy, we must choose a successor for node b at time 2 and at time 3; for the other nodes, only one successor is available. Since two choices are possible at node b, namely going to the destination d or to the intermediate node c, we can define four possible strategies. We denote these strategies by S^{dd}, S^{cd}, S^{dc} and S^{cc}, where u and v in S^{uv} denote the successor of b at time 2 and 3, respectively. The four strategies are shown in Figure 2; each one is represented by the corresponding *hyperpath*, containing the hyperarcs corresponding to the successor arcs.

Each strategy assigns to each hyperpath node i_t a label $\lambda_i(t) = [\lambda_i^1(t), \lambda_i^2(t)]$, where $\lambda_i^1(t)$ is the expected travel time and $\lambda_i^2(t)$ is the expected cost for traveling from node i at departure time t to the destination. For each destination node d_t the label is $[0, 0]$. If (i, j) is the successor arc of node i at departure time t, then the label of node i_t is obtained from the labels at nodes $\{j_\theta :\ \theta \in A_{ij}(t)\}$. For instance, consider strategy S^{dc}; here label $[4, 4]$ for c_4 is obtained from $[0, 0]$ at d_8, since both travel time and cost are 4 for arc (c, d) at time 4; the label $[5, (6 + 4\varepsilon)]$ for a_1 is obtained from labels $[2, 8(1 + \varepsilon)]$ and $[5, 4]$ (nodes b_2 and b_3), since the expected travel time is $(1 + 2)/2 + (2 + 5)/2 = 5$, while the expected cost is $8(1 + \varepsilon)/2 + 4/2 = (6 + 4\varepsilon)$.

Figure 2 reports the labels assigned to hyperpath nodes by each strategy. The four circles in Figure 3 plot the labels assigned to o_0 by the four strategies for $\varepsilon = 0.5$. □

Since $0 < \varepsilon < 1$, the labels $\lambda_o(0)$ given by the four strategies turn out to be non-dominated. However, if the labels $\lambda_a(1)$ are considered, S^{dd} is dominated by S^{cc}. Therefore, a traveller leaving o at time zero and follows strategy S^{dd} has a nonzero probability (actually, probability 1/2) of arriving at a at time one and afterwards following a dominated substrategy. Note also that no other strategy yields the same label $[\lambda_o^1(0), \lambda_o^2(0)]$ as S^{dd}. We may therefore state

Theorem 1 *There may exist an efficient strategy S satisfying:*

1. *S contains non-efficient substrategies.*
2. *There does not exist any strategy $S' \neq S$ with the same label as S containing only efficient substrategies.*

Note that similar results are given in [3] for the bicriterion case, where expected as well as maximum possible values are considered. Theorem 1 shows that a well known property of deterministic bicriterion shortest paths does not extend to adaptive routing in STV networks and thus contradicts Lemma 1 in Opasanon and Miller-Hooks [5]. As a consequence, the proof of correctness of algorithm APS (Proposition 6 in [5]) is not valid, since it exploits Lemma 1. In the next section we show that APS may actually fail.

3 The labelling algorithm

In this section we show that Algorithm APS can fail when applied to our running example. We only provide a short summary of the algorithm here and refer the reader to Opasanon and Miller-Hooks [5] for a complete description.

In general terms, APS is a label correcting algorithm that explores the topological network in a backward fashion, starting from the destination d. For each node i and departure time t, APS maintains an efficient set of *labels*, where each label is a vector containing the expected values of the r criteria for one strategy. A set SE of *scan eligible* nodes is maintained throughout the algorithm; at each iteration, a node j is removed from SE, and each arc $(i,j) \in \mathcal{A}$ is processed. In this phase, in order to generate a label for i at time t, each possible combination of the labels of node j at times in $A_{ij}(t)$ is considered. Whenever new labels are added to label sets, dominated labels are dropped. Upon termination, the strategies corresponding to the final set of labels can be retrieved by means of a suitable predecessor data structure.

Example 1 (continued) We consider a version of APS where the SE list is a FIFO queue. We assume that when the destination d is selected, arc (c,d) is processed before arc (b,d), thus node c is processed before node b. As before, labels correspond to time-cost pairs $[\lambda_i^1(t), \lambda_i^2(t)]$; we denote by $L_i(t)$ the set of labels for i at time t. APS calculates labels in the following way.

1. Node d selected. Arc (c,d): $L_c(4) = \{[4,4]\}$; Arc (b,d): $L_b(2) = \{[2, 8(1+\varepsilon)]\}$, $L_b(3) = \{[9,0]\}$. $SE = \{c, b\}$.

2. Node c selected. Arc (b,c): $L_b(2) = \{[2, 8(1+\varepsilon)], [6,4]\}$, $L_b(3) = \{[9,0],[5,4]\}$. $SE = \{b\}$.

3. Node b selected. Arc (a,b): $L_a(2) = \{[10,0],[6,4]\}$. For $L_a(1)$ we have $|L_b(2)| \cdot |L_b(3)| = 4$ possible combinations of labels, as described below:

	$\lambda_b(2)$	$\lambda_b(3)$	$\lambda_a(1)$
1	$[2, 8(1+\varepsilon)]$	$[9,0]$	$[7, 4(1+\varepsilon)]$
2	$[2, 8(1+\varepsilon)]$	$[5,4]$	$[5, 6+4\varepsilon)]$
3	$[6,4]$	$[9,0]$	$[9,2]$
4	$[6,4]$	$[5,4]$	$[7,4]$

Since $0 < \varepsilon < 1$, label 4 dominates label 1, and the resulting set of efficient labels is $L_a(1) = \{[5, 6+4\varepsilon)], [9,2], [7,4]\}$. $SE = \{a\}$.

4. Node a selected. Arc (o,a): at departure time 0 we have $|L_a(1)| \cdot |L_a(2)| = 6$ possible combinations:

	$\lambda_a(1)$	$\lambda_a(2)$	$\lambda_o(0)$
1	$[5, 6+4\varepsilon]$	$[10,0]$	$[9, 3+2\varepsilon)]$
2	$[5, 6+4\varepsilon]$	$[6,4]$	$[7, 5+2\varepsilon)]$
3	$[9,2]$	$[10,0]$	$[11,1]$
4	$[9,2]$	$[6,4]$	$[9,3]$
5	$[7,4]$	$[10,0]$	$[10,2]$
6	$[7,4]$	$[6,4]$	$[8,4]$

Since label 4 dominates label 1, the resulting set of efficient labels (sorted in increasing order of travel time) is $L_o(0) = \{[7, 5+2\varepsilon)], [8,4], [9,3], [10,2], [11,1]\}$. $SE = \{\}$.

The five crosses in Figure 3 plot the efficient labels in set $L_o(0)$. □

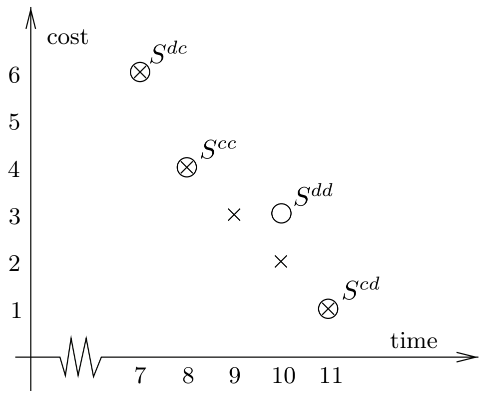


Figure 3: Strategies and efficient labels ($\varepsilon = 1/2$).

Comparing the results of Algorithm APS to Figure 2 leads to two observations.

1. Algorithm APS fails to generate the label $[10, 2(1+\varepsilon)]$, corresponding to strategy S^{dd}. This can be easily explained since the label corresponding to the dominated sub-strategy of S^{dd} for node a at time 1 is discarded.

2. Algorithm APS returns two labels in $L_o(0)$, namely $[9, 3]$ and $[10, 2]$, that do not correspond to any strategy. This fact can be explained by observing that these labels are obtained by combining *conflicting* substrategies. Consider label $[10, 2]$ obtained by combining $[7, 4] \in L_a(1)$ and $[10, 0] \in L_a(2)$. As shown in Figure 4, $[7, 4]$ is obtained from strategy S^{cc}, that assigns to node b the successor arc (b, c), while $[10, 0]$ can be obtained only by choosing arc (b, d) at time 3. Thus the substrategies corresponding to labels $[7, 4]$ and $[10, 0]$ conflict at node b at departure time 3 and cannot be combined in a strategy. Similar observations can be done for label $[9, 3]$.

We conclude that Algorithm APS is not correct for two reasons: first, because it drops dominated substrategies, contradicting Theorem 1; and second, because it combines conflicting substrategies.

4 A simplified model, and a faster parametric algorithm

The STV network model adopted by Opasanon and Miller-Hooks [5] specifies a distribution of possible values for each criterion, arc (i, j) and departure time t. However, this model is unnecessarily detailed, since for each criterion $k > 1$ it suffices to know the *expectation*

$$c_{ij}^k(t) = \sum_{z_k=1}^{D} c_{ij}^{kz_k}(t) \cdot \rho_{ij}^{kz_k}(t).$$

Clearly, the distribution is necessary for travel times; note however that $c_{ij}^k(t)$ can be defined also for $k = 1$. In order to prove that the simplified model is correct, it suffices to show that the formula used to compute new labels for criteria other than travel time (see Step 3 of Algorithm

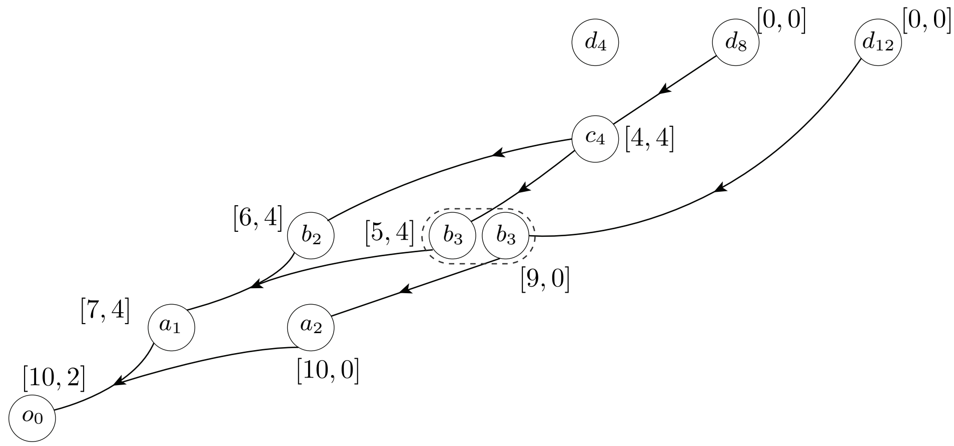


Figure 4: Conflicting substrategies at node b and time 3.

APS, Section 4 in [5]) can be simplified as follows:

$$\begin{aligned}
\eta_i^k(t) &= \textstyle\sum_{(z_1,x)\in Q} \sum_{z_k=1}^{D} \left[c_{ij}^{kz_k}(t) + \lambda_{jx}^k\big(t + c_{ij}^{1z_1}(t)\big) \right] \cdot \rho_{ij}^{1z_1}(t) \cdot \rho_{ij}^{kz_k}(t) \\
&= \textstyle\sum_{(z_1,x)\in Q} \rho_{ij}^{1z_1}(t) \cdot \left[\sum_{z_k=1}^{D} c_{ij}^{kz_k}(t) \cdot \rho_{ij}^{kz_k}(t) \;+\; \sum_{z_k=1}^{D} \lambda_{jx}^k\big(t + c_{ij}^{1z_1}(t)\big) \cdot \rho_{ij}^{kz_k}(t) \right] \\
&= \textstyle\sum_{(z_1,x)\in Q} \rho_{ij}^{1z_1}(t) \cdot \left[c_{ij}^k(t) \;+\; \lambda_{jx}^k\big(t + c_{ij}^{1z_1}(t)\big) \right] \\
&= c_{ij}^k(t) \;+\; \textstyle\sum_{(z_1,x)\in Q} \rho_{ij}^{1z_1}(t) \lambda_{jx}^k\big(t + c_{ij}^{1z_1}(t)\big)
\end{aligned}$$

Note that this simplification is purely algebraic, and does not depend on the choice of the labels to combine. Since the new label is obtained as a function of the expectation $c_{ij}^k(t)$ (and of the travel time distribution) the model assigning a single expected value (as originally defined in [6] for the single criterion case) can be adopted without loss of generality.

Consider now the parametric problem (Section 4.2 in [5]), where we must find a strategy minimizing the expectation of a weighted sum of the criteria with respect to weights $w = \{w^k : 1 \le k \le r\}$, $\sum_{k=1,\ldots,r} w^k = 1$. For each arc (i, j) and departure time t we define the *parametric cost* c_{ij}^w as follows:

$$c_{ij}^w(t) = \sum_{k=1}^{r} w^k \cdot c_{ij}^k(t). \tag{1}$$

Recall that Algorithm ALEDS II computes label $U_i(t)$ for each node i and time t; the formula used to compute these labels (Equation (2), Lemma 2 in [5]) can be rewritten as follows:

$$\begin{aligned}
\nu_i(t) &= \textstyle\sum_{k=1}^{r} w^k \cdot \left[\sum_{z_k=1}^{D} c_{ij}^{kz_k}(t) \cdot \rho_{ij}^{kz_k}(t) \right] \;+\; \sum_{z_1=1}^{D} U_j\big(t + c_{ij}^{1z_1}(t)\big) \cdot \rho_{ij}^{1z_1}(t) \\
&= \textstyle\sum_{k=1}^{r} w^k \cdot c_{ij}^k(t) \;+\; \sum_{z_1=1}^{D} U_j\big(t + c_{ij}^{1z_1}(t)\big) \cdot \rho_{ij}^{1z_1}(t) \\
&= c_{ij}^w(t) \;+\; \textstyle\sum_{z_1=1}^{D} U_j\big(t + c_{ij}^{1z_1}(t)\big) \cdot \rho_{ij}^{1z_1}(t).
\end{aligned}$$

With the above simplification, Lemma 2 in [5] implies that for a given strategy S, the values $U_i(t)$ can be defined recursively as:

$$U_i(t) = c_{ij}^w(t) \;+\; \sum_{z_1=1}^{D} U_j\big(t + c_{ij}^{1z_1}(t)\big) \cdot \rho_{ij}^{1z_1}(t) \tag{2}$$

where (i, j) is the successor of node i at time t in S. Equation (2) is a particular case of the definition of cost given in [6]. We conclude that the parametric problem reduces to finding an optimal strategy with respect to the parametric cost (1). This allows us to devise an improved algorithm for the parametric problem, whose computational complexity is discussed below.

Recall that we have r criteria and $I+1$ time instances. Let us denote by $V = |\mathcal{N}|$ and $E = |\mathcal{A}|$ the number of nodes and arcs in the topological network. Let P denote the maximum number of possible travel times for each arc and departure time, and let H be the sum, over all arcs and times, of the number of possible travel times. We may assume $H = \Theta(P \cdot I \cdot E)$, even if in some cases H may be significantly smaller than $(P \cdot I \cdot E)$. For each criterion $k > 1$ we have one possible value for each arc and time, thus the description of the input STV network has length $O(H + r \cdot E \cdot I)$, i.e., $O(E \cdot I \cdot (r + P))$.

First of all, note that the expected travel times $\lambda^1_{ij}(t)$ are not part of the input data, but can be computed in $O(H)$ time. Given a weight vector w, the parametric weights $c^w_{ij}(t)$ can be computed in $O(r \cdot E \cdot I)$ time. The best strategy for a single cost criterion can be found in linear time in the size of the time-expanded hypergraph, which is $O(H)$ [6]. We thus have the following result.

Proposition 1 *The parametric problem can be solved in linear time in the input size, that is, in $O(E \cdot I \cdot (r + P))$ time.*

It is interesting to compare this result to the complexity of algorithm ALEDS II, namely $O(V^3 \cdot I^2 \cdot P \cdot r)$. Assuming $E = \Theta(V^2)$, as implicitly done in [5], our method improves on ALEDS II of a factor $V \cdot I \cdot (P \cdot r)/(P + r)$, i.e. $O(V \cdot I \cdot \min\{P, r\})$.

As a final remark, it is worth noting that some efficient strategies may not be found by solving the parametric problem, regardless of the chosen weights. In fact, it is well known that only *supported* efficient solutions can be found with this approach. In our running example, S^{dd} is an unsupported strategy which cannot be found by solving the parametric problem. The interested reader is referred to [4] for further details about finding unsupported points.

5 Final remarks

The results provided in the previous sections show that Algorithm APS should be modified in (at least) two ways:

1. Labels obtained by combining conflicting sub-strategies must be ignored.
2. Dominated labels cannot be discarded.

Although a correct labelling algorithm may be obtained in this way, its practical efficiency remains questionable. On the other hand, alternative approaches seem to show some weaknesses as well; this is the case, for example, of the two-phase method proposed in Nielsen et al. [4]. These results seem to indicate that bicriterion adaptive routing in STV networks is quite hard, so that approximation algorithms may be a reasonable direction for further research.

Interestingly, we can look at the issue of conflicting substrategies in the labelling approach from a different perspective. Consider again Figure 4, showing the computation of label $[10, 2]$; this figure can be interpreted in terms of a traveller choosing the successor of node b at time 3 as follows:

- if the departure time from a was 1 go to node c.
- if the departure time from a was 2 go to the destination node d.

The reader can easily verify that the pair $[10, 2]$ correctly gives the expected travel time and cost obtained with this route choice. However, this route choice is *not* allowed in the time-adaptive model, since the successor at node b and time 3 is not unique, but depends on the departure time from a. Such route choice would be allowed in a more general routing model, that we may define as a *history-adaptive* route choice, where the successor arc for each node and time is chosen as a function of the previous events, i.e. realizations of travel times. In general, the labels computed by Algorithm APS may be interpreted in terms of *history-adaptive strategies.* Since the history-adaptive model is more flexible, we may expect some efficient time-adaptive strategies to be dominated by history-adaptive strategies; indeed, this is the case for strategy S^{dd}, dominated by the history-adaptive strategy in Figure 3, i.e. by label $[10, 2]$.

Here we do not state any formal definition or property of history-adaptive route choice; as a matter of fact, at this time we do not know whether such a model may be suitable in some application and is computationally tractable; we believe however that this may be an interesting subject for further research.

References

[1] S. Gao and I. Chabini. Optimal routing policy problems in stochastic time-dependent networks. *Transportation Research Part B*, 40:93–122, 2006. doi:10.1016/j.trb.2005.02.001.

[2] R.W. Hall. The fastest path through a network with random time-dependent travel times. *Transportation Science*, 20(3):182–188, 1986.

[3] L.R. Nielsen. *Route Choice in Stochastic Time-Dependent Networks*. PhD thesis, Department of Operations Research, University of Aarhus, 2004. URL `http://www.imf.au.dk/publs?id=499`.

[4] L.R. Nielsen, K.A. Andersen, and D. Pretolani. Bicriterion shortest hyperpaths in random time-dependent networks. *IMA Journal of Management Mathematics*, 14(3):271–303, 2003. doi:10.1093/imaman/14.3.271.

[5] S. Opasanon and E. Miller-Hooks. Multicriteria adaptive paths in stochastic, time-varying networks. *European Journal of Operational Research*, 173:72–91, 2006. doi:10.1016/j.ejor.2004.12.003.

[6] D. Pretolani. A directed hypergraph model for random time-dependent shortest paths. *European Journal of Operational Research*, 123(2):315–324, 2000. doi:10.1016/S0377-2217(99)00259-3.

Working Papers from Logistics/SCM Research Group

L-2006-11 Daniele Pretolani, Lars Relund Nielsen & Kim Allan Andersen: A note on "Multicriteria adaptive paths in stochastic, time-varying networks".

L-2006-10 Lars Relund Nielsen, Kim Allan Andersen & Daniele Pretolani: Bicriterion a priori route choice in stochastic time-dependent networks.

L-2006-09 Christian Larsen & Gudrun P. Kiesmüller: Developing a closed-form cost expression for an *(R,s,nQ)* policy where the demand process is compound generalized Erlang.

L-2006-08 Eduardo Uchoa, Ricardo Fukasawa, Jens Lysgaard, Artur Pessoa, Marcus Poggi de Aragão, Diogo Andrade: Robust Branch-Cut-and-Price for the Capacitated Minimum Spanning Tree Problem over a Large Extended Formulation.

L-2006-07 Geir Brønmo, Bjørn Nygreen & Jens Lysgaard: Column generation approaches to ship scheduling with flexible cargo sizes.

L-2006-06 Adam N. Letchford, Jens Lysgaard & Richard W. Eglese: A Branch-and-Cut Algorithm for the Capacitated Open Vehicle Routing Problem.

L-2006-05 Ole Mortensen & Olga W. Lemoine: Business integration between manufacturing and transport-logistics firms.

L-2006-04 Christian H. Christiansen & Jens Lysgaard: A column generation approach to the capacitated vehicle routing problem with stochastic demands.

L-2006-03 Christian Larsen: Computation of order and volume fill rates for a base stock inventory control system with heterogeneous demand to investigate which customer class gets the best service.

L-2006-02 Søren Glud Johansen & Anders Thorstenson: Note: Optimal base-stock policy for the inventory system with periodic review, backorders and sequential lead times.

L-2006-01 Christian Larsen & Anders Thorstenson: A comparison between the order and the volume fill rates for a base-stock inventory control system under a compound renewal demand process.

L-2005-02 Michael M. Sørensen: Polyhedral computations for the simple graph partitioning problem.

L-2005-01 Ole Mortensen: Transportkoncepter og IT-støtte: et undersøgelsesoplæg og nogle foreløbige resultater.

L-2004-05 Lars Relund Nielsen, Daniele Pretolani & Kim Allan Andersen: *K* shortest paths in stochastic time-dependent networks.

L-2004-04 Lars Relund Nielsen, Daniele Pretolani & Kim Allan Andersen: Finding the K shortest hyperpaths using reoptimization.

L-2004-03 Søren Glud Johansen & Anders Thorstenson: The (r,q) policy for the lost-sales inventory system when more than one order may be outstanding.

L-2004-02 Erland Hejn Nielsen: Streams of events and performance of queuing systems: The basic anatomy of arrival/departure processes, when focus is set on autocorrelation.

L-2004-01 Jens Lysgaard: Reachability cuts for the vehicle routing problem with time windows.

Handelshøjskolen i Århus

Aarhus
School of Business

ISBN 87-7882-156-8

Department of Business Studies

Aarhus School of Business
Fuglesangs Allé 4
DK-8210 Aarhus V - Denmark

Tel. +45 89 48 66 88
Fax +45 86 15 01 88

www.asb.dk