The Next-to-Shortest Path Problem on Directed Graphs with Positive Edge Weights

Bang Ye Wu
Department of Computer Science and Information Engineering, National Chung Cheng University, ChiaYi, Taiwan 621, R.O.C.

Hung-Lung Wang
Institute of Information and Decision Sciences, National Taipei University of Business, Taipei, Taiwan 100, R.O.C.

Given an edge-weighted graph *G* and two distinct vertices *s* and *t* of *G*, the next-to-shortest path problem asks for a path from *s* to *t* of minimum length among all paths from *s* to *t* except the shortest ones. In this article, we consider the version where *G* is directed and all edge weights are positive. Some properties of the requested path are derived when *G* is an arbitrary digraph. In addition, if *G* is planar, an $O(n^3)$-time algorithm is proposed, where *n* is the number of vertices of *G*. © 2015 Wiley Periodicals, Inc. NETWORKS, Vol. 000(00), 000–000 2015

Keywords: algorithm; time complexity; planar graph; digraph; directed acyclic graph; *k* shortest paths; next-to-shortest path

1. INTRODUCTION

In this article, we are concerned with the next-to-shortest path problem, defined as follows. Given a graph G, let an edge with endpoints u and v be denoted by uv. We call $\ell(uv)$ the weight of uv. A walk in G is a list $v_0, e_1, v_1, \ldots, e_k, v_k$ of vertices and edges such that, for $1 \le i \le k, e_i = v_{i-1}v_i$. A path in G is a walk with no repeated vertices, and a path may consist of only a single vertex and no edge. Specifically, a u, v-walk is a walk such that $v_0 = u$ and $v_k = v$, and a u, v-path is defined analogously. The length of a path is the sum of the weights of all edges in the path. A u, v-path is shortest if its length is minimum among all u, v-paths. For any two vertices u and v, the distance between u and v, denoted by $d(u, v)$, is the length of a shortest u, v-path. A next-to-shortest u, v-path is a u, v-path whose length is minimum among the u, v-paths of length greater than $d(u, v)$. A formal definition of the next-to-shortest path problem is given in Problem 1.

Problem 1 (the next-to-shortest path problem). The input and output of the next-to-shortest path problem are specified as follows.

- Input: a quadruple (G, s, t, ℓ), where G is a graph, s and t are distinct vertices of G, and ℓ is a real-valued function defined on the set of edges of G.
- Output: a next-to-shortest s, t-path.

For an instance (G, s, t, ℓ), the graph G is called the underlying graph, vertices s and t are the source and destination, respectively. The next-to-shortest path problem was studied by Lalgudi and Papaefthymiou [13] on digraphs. They showed that the problem is NP-hard but can be efficiently solved if paths are replaced by walks in the problem definition. The next-to-shortest path problem on special graph classes was also studied [3, 15]. For undirected graphs with positive edge weights, the first polynomial-time algorithm was presented by Krasikov and Noble [12] with time complexity $O(n^3m)$, in which n and m are the number of vertices and edges, respectively. The time complexity has been improved several times [11, 14, 22], and the currently best result is a linear time algorithm, assuming that the distances from s and t to all other vertices are given [22]. Hence, for undirected graphs with positive edge weights, the next-to-shortest path problem can be solved as efficiently as the single-source shortest-path problem. The next-to-shortest path problem becomes much more difficult with respect to the following two generalizations: either zero-weight edges are allowed, or the underlying graph is directed. Zhang and Nagamochi [26] showed that the version for undirected graphs with nonnegative edge weights admits an $O(n^9m)$-time algorithm. Independently, Wu et al. showed in a manuscript [23] that this version can be solved in linear time if the distances from s and t to all other vertices are given. If the underlying graph is directed, no polynomial-time algorithm

Received November 2013; accepted November 2014
Correspondence to: H.-L. Wang; e-mail: hlwang@ntub.edu.tw
Contract grant sponsor: NSC 100–2221-E-194-036-MY3 and NSC 101-2221-E-194-025-MY3 (to B.Y.W), Contract grant sponsor: NSC 101-2221-E-141-002 (National Science Council, Taiwan, R.O.C.; to H.-L.W)
DOI 10.1002/net.21598
Published online in Wiley Online Library (wileyonlinelibrary.com).

is known unless the graph is acyclic. In such a case, a linear time algorithm can be derived immediately from Lalgudi and Papaefthymiou [13].

In this article, we focus on the next-to-shortest path problem where the underlying graph is directed and edge weights are positive. For an instance (G, s, t, ℓ) with G being undirected, an important property used to construct a next-to-shortest s, t-path is that the requested path contains at most one outward subpath, which is a path not in the union of all shortest s, t-paths except its endpoints [22, 23]. Unfortunately, as shown later by an example, this property does not hold for digraphs. Nevertheless, we show that the outward subpaths of a next-to-shortest s, t-path pass through only a specific set of vertices, and this property is useful in designing algorithms. By further exploring the properties of a next-to-shortest s, t-path on a planar digraph, we design an $O(n^3)$-time algorithm for the planar case.

1.1. Related Work

The next-to-shortest path problem is closely related to the 2-disjoint paths problem, defined as follows. Two paths are disjoint if they have no vertex in common and internally disjoint if they share no vertex except the endpoints. For a given graph and k source-to-destination pairs, the k-disjoint paths problem, abbreviated as k-VDP, is to determine if there exist disjoint paths connecting the given k sources to their corresponding destinations. When the number of pairs is fixed, k-VDP is polynomial-time solvable on undirected graphs [16, 19] but is NP-complete on digraphs [7]. On dags (directed acyclic graphs), it is known that k-VDP can be solved in polynomial time for fixed k [7]. Recently, Tholey [20] gave a linear time algorithm for 2-VDP on dags. On planar digraphs, k-VDP is also polynomial-time solvable for fixed k [17], and furthermore, it is fixed-parameter tractable [5]. Similarly, there exists a linear time algorithm for 2-VDP on planar digraphs [21].

We mention two problems, whose formulations generalize that of the next-to-shortest path problem. Although these two problems are more general, the results do not induce an efficient algorithm for the problem we are concerned with. The first is the problem of determining k Shortest Paths (kSP), and the second is that of computing shortest paths avoiding forbidden subpaths (AFS).

In kSP, besides the quadruple (G, s, t, ℓ) as in the next-to-shortest path problem, a positive integer k is given, and it asks for the k shortest s, t-paths in nondecreasing order of length. For G being directed with nonnegative edge weights, the best known result is proposed by Yen [24, 25], with worst-case time complexity $O(kn(m + n \log n))$. Recently, Hershberger et al. [9, 10] gave some results for kSP from both theoretical and practical aspects. When paths are replaced by walks in the problem definition, the problem becomes less difficult, and the best known result to date is provided by Eppstein [6]. A next-to-shortest s, t-path is in fact the $(r + 1)$th shortest s, t-path, where r is the number of shortest s, t-paths. As r may be exponential in terms of n, Yen's algorithm does not induce a polynomial-time algorithm for the next-to-shortest path problem.

In AFS, besides (G, s, t, ℓ), an additional set S of paths is given, and a shortest path with no subpath being an element in S is sought. By taking all shortest s, t-paths as the set S, the solution of AFS is indeed the solution of the next-to-shortest path problem. However, AFS is NP-hard, even when S consists of paths with two edges [18]. For the shortest walk AFSs problem, some efficient algorithms were proposed. See Ahmed and Lubiw [1] for a brief survey.

The rest of this article is organized as follows. In Section 2, we give some notation and definitions used in this article, as well as some basic properties. In Sections 3, we elaborate the next-to-shortest path problem on digraphs, and the case of planar digraphs is discussed in Section 4. Concluding remarks are given in Section 5.

2. PRELIMINARIES

In this section, we define the notation and the terminology. Fundamental properties of a next-to-shortest path in a digraph are also summarized. All the analyses related to computational complexity are based on the uniform-cost RAM model [2]. In this model, a number is able to be stored in a single computer word, and arithmetic operations on two numbers can be done in constant time. For an instance (G, s, t, ℓ), the vertex set and edge set of G are denoted by V and E, respectively, and we let $n = |V|$ and $m = |E|$. For a subgraph H of G, the vertex set and edge set of H is denoted by $V(H)$ and $E(H)$, respectively. For any $U \subseteq V(H)$, the subgraph induced by U on H is denoted by $H[U]$.

For vertices x and y on path P, let P_{xy} denote the subpath from x to y. When $x = y$, the path P_{xy} is a vertex without any edge. By "$P : u \underset{H}{\rightsquigarrow} v$," we denote that the path P is a u, v-path in graph H. If P is a u, v-walk and Q is a v, w-walk, then P and Q can be concatenated to obtain a u, w-walk. Such an operation is called a concatenation of walks P and Q, and the result is denoted by $P \circ Q$.

In a graph G, let $d_G(u, v)$ denote the distance between u and v in G. If there is no u, v-path in G, then $d_G(u, v) = \infty$. The subscript is omitted for succinctness if the distance is measured in the underlying graph of a given instance. For convenience, let $d_s(v) = d(s, v)$ and $d_t(v) = d(v, t)$, where s is the source and t is the destination.

For two distinct vertices u and v of a dag G, if there is a u, v-path in G, then u is said to be an ancestor of v, and v is a descendent of u. Notice that in this definition, a vertex is not an ancestor of itself. Given an instance (G, s, t, ℓ) with G being directed, let D be the union of all shortest s, t-paths. If all edges have positive weights, the subgraph D is a dag. We define two binary relations $\prec_{D'}$ and $\preceq_{D'}$ on $V(D')$, in which D' is a subgraph of D, and

- $u \prec_{D'} v$ iff u is an ancestor of v in D';
- $u \preceq_{D'} v$ iff $u \prec_{D'} v$ or $u = v$.

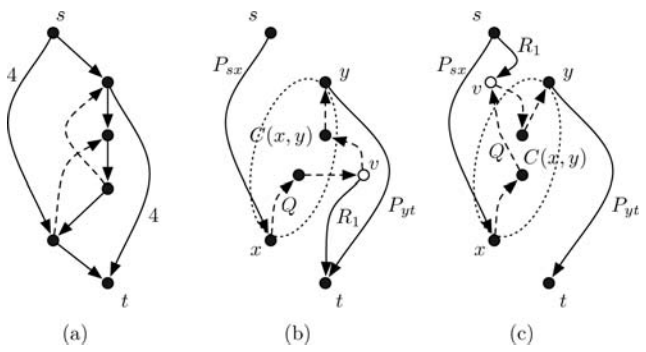

FIG. 1. Outward subpaths and mixed subpaths in a digraph with given s and t. (a) An example of a next-to-shortest path with two outward subpaths, where edges without indicated numbers are of length one. The solid lines are edges in D and the dashed lines are outward paths. The graph has only one next-to-shortest path which contains the two outward subpaths. (b) The case where a mixed subpath contains a vertex v in D but not an ancestor of x. (c) The case where a mixed subpath contains a vertex v in D but not a descendant of y.

For $D' = D$, we simplify $\prec_D$ and $\preceq_D$ to $\prec$ and $\preceq$, respectively. Obviously, for any vertices u and v in $V(D)$, a u, v-path in D is a shortest u, v-path in G. An outward path, as mentioned in Section 1, is a path such that both endpoints are in $V(D)$, and all edges are from the set $E\backslash E(D)$. We list some simple properties, which hold immediately from the definitions, in the following.

Property 1. *For any two vertices s and t of a digraph, an s, t-path containing an outward subpath is longer than a shortest s, t-path.*

Property 2. *In a dag, any path leading to a vertex v contains only ancestors of v. Similarly, any path leading from v contains only descendants of v.*

Property 3. *In a dag, if u is not an ancestor of v, any path starting from u contains no ancestor of v.*

Property 4. *In a dag, any u, v-path and v, x-path are internally disjoint.*

In the following, we simply use the term "next-to-shortest path" to refer to a next-to-shortest s, t-path when s and t are the source and destination in the instance being considered.

3. PROPERTIES OF A NEXT-TO-SHORTEST PATH ON DIGRAPHS WITH POSITIVE EDGE WEIGHTS

In this section, we consider the next-to-shortest path problem on a general digraph with positive edge weights. As mentioned in Section 1, the reason why the undirected version can be solved efficiently is that a next-to-shortest path contains exactly one outward subpath. However, in a digraph with positive edge weights, this property does not hold. In Figure 1a, we show such an example.

Although, in a digraph, a next-to-shortest path may contain more than one outward subpath, a similar concept as in a undirected graph can be obtained by considering the maximal mixed subpath, which is defined as follows.

Definition 1. *For a given instance, a mixed path is a path starting and ending with outward subpaths. The maximal mixed subpath of a path is a mixed subpath not properly contained in another mixed subpath.*

Notice that a mixed subpath contains at least one outward subpath and may be exactly an outward subpath. The next property is trivial since an s, t-path without any mixed subpath is a shortest s, t-path.

Property 5. *A next-to-shortest path contains exactly one maximal mixed subpath.*

Based on Property 5, a straightforward approach for solving the next-to-shortest path problem is to find, for all x and y, a shortest path $P = P_{sx} \circ P_{xy} \circ P_{yt}$, in which P_{sx} and P_{yt} are two disjoint paths in D, and P_{xy} is a mixed path. If the next-to-shortest path problem can be "decomposed" as computing two disjoint paths P_{sx} and P_{yt} in D with a shortest path P_{xy} in a specific subgraph of G, then a polynomial-time algorithm can be derived. The reason is that if edge weights are positive, then D is a dag, and the 2-VDP problem on D can be solved in linear time [20]. In the following, we show that the next-to-shortest path problem can be decomposed in some circumstances.

Definition 2. *For two vertices x and y in V(D), let* $C(x, y) = \{v \in V(D) : y \prec v \prec x\} \cup \{x, y\}$. *Define* $H_{xy} = G[(V\backslash V(D)) \cup C(x, y)]$.

Notice that $C(x, y) = \{x, y\}$ *if y is not an ancestor of x in D.*

Lemma 1. *If P is a next-to-shortest path with maximal mixed subpath Q, then* $V(Q) \cap V(D) \subseteq C(x, y)$, *where* $Q = P_{xy}$.

Proof. Suppose to the contrary that $V(Q) \cap V(D) \not\subseteq C(x, y)$. First, we consider the case where there is a vertex v in $(V(Q) \cap V(D))\backslash C(x, y)$ such that $v \not\preceq x$. Assume that v is the first such vertex when traveling along P, and R_1 is any v, t-path in D (Fig. 1b). We claim that $R = P_{sv} \circ R_1$ is a better solution, which contradicts the optimality of P. By definition, P_{sv} contains at least one outward subpath, and, therefore, $\ell(R) > d(s, t)$. Since the vertices in D passed through by P_{sv} are all ancestors of x, and R_1 contains no ancestor of x, Property 2 implies that R is a path.

The other case is that each vertex v in $(V(Q) \cap V(D))\backslash C(x, y)$ satisfies $v \prec x$ and $y \not\prec v$. Let v be one of those vertices with minimum distance from s, and let R_1 be any s, v-path in D (Fig. 1c). We claim that $R = R_1 \circ P_{vt}$ is a better solution. By definition, P_{vt} contains at least one outward subpath and, therefore, $\ell(R) > d(s, t)$. Since R_1 passes through only ancestors of v and P_{vt} contains no ancestor of v, we have that R is a path. ∎

By Lemma 1, P_{xy} is a path in H_{xy}. Notice that P_{xy} may not be a shortest x, y-path in H_{xy}, and thus, it is not feasible to compute P_{xy} via computing the shortest x, y-path in H_{xy}. However, there is an easier case that can be solved via Lemma 1. The details are provided in Corollary 1.

Corollary 1. *If P is a next-to-shortest path with maximal mixed subpath P_{xy} such that $y \not\prec x$, then P_{xy} is an outward path. Furthermore, P_{xy} is a shortest x, y-path in H_{xy}.*

When $y \not\prec x$, any s, x-path and y, t-path in D are disjoint, and therefore, we have the next result.

Corollary 2. *Suppose that P is a next-to-shortest path with maximal mixed subpath P_{xy} such that $y \not\prec x$. If $Q_1 : s \underset{D}{\rightsquigarrow} x$ and $Q_2 : y \underset{D}{\rightsquigarrow} t$ are disjoint and Q_0 is a shortest x, y-path in H_{xy}, then $Q_1 \circ Q_0 \circ Q_2$ is a next-to-shortest path.*

4. AN $O(N^3)$-TIME ALGORITHM ON PLANAR DIGRAPHS WITH POSITIVE EDGE WEIGHTS

As mentioned in Section 3, a next-to-shortest path P is of the form $s \underset{D}{\rightsquigarrow} x \underset{G}{\rightsquigarrow} y \underset{D}{\rightsquigarrow} t$, where P_{xy} is a maximal mixed subpath. The idea to solve the next-to-shortest path problem on planar digraphs with positive edge weights is to enumerate all possible vertex pairs (x, y) and for each of them, compute a shortest path of the form $s \underset{D}{\rightsquigarrow} x \underset{G}{\rightsquigarrow} y \underset{D}{\rightsquigarrow} t$. For the case where $y \not\prec x$, Corollary 2 leads to the following result.

Lemma 2. *Given (G, s, t, ℓ), if G is planar and there exists a next-to-shortest path P with maximal mixed subpath P_{xy} such that $y \not\prec x$, a next-to-shortest path can be computed in $O(n^2)$ time.*

Proof. By Corollary 2, our goal is to find two vertices $u, v \in V(D)$ with $v \not\prec u$ such that $d_s(u) + d_t(v) + d_{H_{uv}}(u, v)$ is minimized. As H_{uv} can be computed in $O(n)$ time for given u and v and the single-source shortest-path problem can be solved in $O(n)$ time [8], the requested vertices can be computed in $O(n^3)$ time by evaluating $d_s(u) + d_t(v) + d_{H_{uv}}(u, v)$ for all $u, v \in V(D)$ with $v \not\preceq u$. Nevertheless, we show that the requested vertices can be computed more efficiently by computing $\min_{v \in V_u}(d_s(u) + d_t(v) + d_{H_{uv}}(u, v))$ for all $u \in V(D)$, where $V_u = \{v \in V(D) : v \not\preceq u\}$, and then choosing the pair of vertices for which the minimum is achieved.

For any $u \in V(D)$, let G_u be the subgraph obtained by removing all edges of D and all ancestors of u from G, and let $\phi_u : V_u \to \mathbb{R}^+$ be defined as

$$\phi_u(v) = d_s(u) + d_t(v) + d_{G_u}(u, v).$$

For any $u, v \in V(D)$ with $v \not\prec u$, as H_{uv} is a subgraph of G_u, we have

$$\phi_u(v) \le d_s(u) + d_t(v) + d_{H_{uv}}(u, v). \tag{1}$$

We claim that for $v^* \in \operatorname{argmin}_{v \in V_u} \phi_u(v)$, the equation

$$\phi_u(v^*) = d_s(u) + d_t(v^*) + d_{H_{uv^*}}(u, v^*) \tag{2}$$

holds, and then by (1), one can derive that $v^* \in \operatorname{argmin}_{v \in V_u}(d_s(u) + d_t(v) + d_{H_{uv}}(u, v))$.

Suppose to the contrary that equation (2) does not hold. It follows that there exists a vertex v^* belonging to $\operatorname{argmin}_{v \in V_u} \phi_u(v)$ such that $\phi_u(v^*) < d_s(u) + d_t(v^*) + d_{H_{uv^*}}(u, v^*)$. Therefore, the path $P : u \underset{G_u}{\rightsquigarrow} v^*$ with $\ell(P) = d_{G_u}(u, v^*)$ passes through a vertex w in $V_u \backslash \{v^*\}$, and thus

$$d_{G_u}(u, v^*) = d_{G_u}(u, w) + d_{G_u}(w, v^*). \tag{3}$$

Recall that in G_u, all edges of D have been removed. If $w \prec v^*$, then $d(w, v^*) < d_{G_u}(w, v^*)$, and we have

$$\begin{aligned} & d_s(u) + d_t(w) + d_{G_u}(u, w) \\ &\quad = d_s(u) + d(w, v^*) + d_t(v^*) + d_{G_u}(u, w) \\ &\quad < d_s(u) + d_{G_u}(w, v^*) + d_t(v^*) + d_{G_u}(u, w) \\ &\quad \underset{(3)}{=} d_s(u) + d_t(v^*) + d_{G_u}(u, v^*). \end{aligned}$$

Otherwise, $w \not\prec v^*$, and thus $d_t(w) < d(w, v^*) + d_t(v^*)$. We have

$$\begin{aligned} & d_s(u) + d_t(w) + d_{G_u}(u, w) \\ &\quad < d_s(u) + d(w, v^*) + d_t(v^*) + d_{G_u}(u, w) \\ &\quad \le d_s(u) + d_{G_u}(w, v^*) + d_t(v^*) + d_{G_u}(u, w) \\ &\quad \underset{(3)}{=} d_s(u) + d_t(v^*) + d_{G_u}(u, v^*). \end{aligned}$$

Both cases contradict that $\phi_u(v^*)$ is minimum. Consequently, equation (2) holds.

The pair (u, v) that minimizes $\phi_u(v)$ can be computed in $O(n^2)$ time as follows. In a preprocessing stage, we compute $d_s(v)$ and $d_t(v)$ for all $v \in V(G)$ by solving the single-source shortest-path problem on G. As G is planar, the preprocessing can be done in $O(n)$ time [8]. For each $u \in V(D)$, we can construct G_u and compute the distances from u to all vertices in V_u on G_u in $O(n)$ time. Thus, the overall time complexity is $O(n^2)$. ■

Next, we consider the case where $y \prec x$. For any $y \prec x$, we show in the following that if there exists a next-to-shortest path with a maximal mixed subpath from x to y, then one can compute a shortest such path in $O(n)$ time. Similarly, the strategy is to decompose the problem into a 2-VDP and a shortest path problem. In the discussion below, we assume that G is a plane graph. The embedding is represented by adjacency lists satisfying that all neighbors of a vertex appear in clockwise order. Such adjacency lists can be computed in $O(n)$ time using PQ-trees [4]. The vertices and edges are also treated as points and polygonal curves, respectively, in a two-dimensional plane. This assumption makes the notion of a region enclosed by some curves clear.

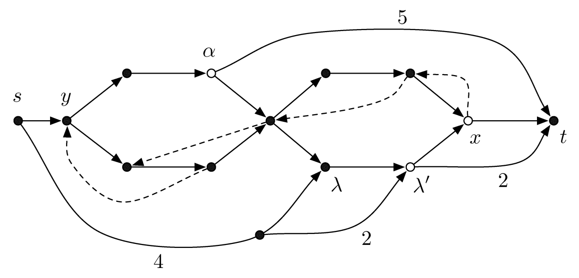

FIG. 2. Valid vertex pairs and extreme vertex pairs with respect to (x, y). The subgraph D consists of all the vertices and solid edges. The set of entrances of $C(x, y)$ is $\{y, \lambda, \lambda'\}$, and the set of exits of $C(x, y)$ is $\{\alpha, \lambda', x\}$. An entrance can simultaneously be an exit, e.g. λ'. Valid vertex pairs with respect to (x, y) are (λ, α) and (λ', α). There is exactly one extreme vertex pair (λ', α) with respect to (x, y).

For any $x, y \in V(D)$ with $y \prec x$, the boundary B of $G[C(x,y)]$ is defined as two x, y-paths B_1 and B_2 enclosing all the vertices and edges of $G[C(x,y)]$. Notice that B_1 and B_2 may not be internally disjoint. With the embedding given in [4], the boundary of $C(x, y)$ can be computed in $O(n)$ time by applying breadth-first search from y and then backtrack from x. For any two vertices u and v with $u \prec_B v$ in B, we denote by B_{uv} a u, v-path in B_1 or in B_2. Notice that B_{uv} may be simultaneously a subpath in B_1 and B_2. Vertex v is said to be an entrance of $C(x, y)$ if there is an edge uv satisfying $u \in V(D)\backslash C(x, y)$. Similarly, vertex v is said to be an exit of $C(x, y)$ if there is an edge vw such that $w \in V(D)\backslash C(x, y)$. A vertex pair (λ, α) is said to be valid with respect to (x, y) if λ and α are an entrance and an exit of $C(x, y)$, respectively, and there are disjoint paths P_1 and P_2 such that

$$P_1 : s \underset{D}{\rightsquigarrow} \lambda \underset{B}{\rightsquigarrow} x \quad \text{and} \quad P_2 : y \underset{B}{\rightsquigarrow} \alpha \underset{D}{\rightsquigarrow} t.$$

An illustration is given in Figure 2. We say that (λ, α) is an entrance-exit pair on an s, t-path P if, when traveling along P, λ, and α are the first and the last vertices in $C(x, y)$, respectively.

Lemma 3. *If P is a next-to-shortest path with maximal mixed subpath P_{xy} and (λ, α) is an entrance-exit pair on P, then (λ, α) is valid with respect to (x, y).*

Proof. By definition, we have that $P_{s\lambda}$ and $B_{y\alpha}$ are disjoint and so are $B_{\lambda x}$ and $P_{\alpha t}$. We claim that the paths $P_{s\lambda} \circ B_{\lambda x}$ and $B_{y\alpha} \circ P_{\alpha t}$ are disjoint by showing the following:

- $P_{s\lambda}$ and $P_{\alpha t}$ are disjoint.
  These two paths are disjoint as P is of the form $s \underset{D}{\rightsquigarrow} \lambda \underset{D}{\rightsquigarrow} x \underset{G}{\rightsquigarrow} y \underset{D}{\rightsquigarrow} \alpha \underset{D}{\rightsquigarrow} t$.
- $B_{\lambda x}$ and $B_{y\alpha}$ are disjoint.
  As $P_{\lambda x}$ and $P_{y\alpha}$ are disjoint and are enclosed within the region $G[C(x,y)]$, it follows that $V(B_{\lambda x}) \cap V(B_{y\alpha}) = \emptyset$. ■

Based on Lemma 3, the following corollary can be derived.

Corollary 3. *Let P and (λ, α) be defined as in Lemma 3. There is a next-to-shortest path P' such that $P' = P_{s\lambda} \circ B_{\lambda x} \circ P_{xy} \circ B_{y\alpha} \circ P_{\alpha t}$.*

Proof. We prove this corollary by showing that P' is a path with the same length as P. To show that P' is a path, we claim that $V(B_{\lambda x}) \cap V(P_{xy}) = \{x\}$ and $V(P_{xy}) \cap V(B_{y\alpha}) = \{y\}$. Suppose to the contrary that there is a vertex v satisfying $v \in V(B_{\lambda x}) \cap V(P_{xy})$ and $v \neq x$. One may construct a path $P'' = P_{s\lambda} \circ B_{\lambda v} \circ P_{vy} \circ B_{y\alpha} \circ P_{\alpha t}$, which is shorter than P. Similarly, $V(P_{xy}) \cap V(B_{y\alpha}) = \{y\}$ holds.

Next, we show that the lengths of P and P' are equal. One can easily show that $\ell(B_{\lambda x}) = \ell(P_{\lambda x})$ and $\ell(B_{y\alpha}) = \ell(P_{y\alpha})$. The reason is that paths in D with the same endpoints are of the same length. Therefore, P' is a next-to-shortest path. ■

Based on Corollary 3, we may focus on computing a next-to-shortest path of the form of P'. In other words, for any $y \prec x$, we intend to compute a shortest path P of the form $s \underset{D}{\rightsquigarrow} \lambda \underset{B}{\rightsquigarrow} x \underset{G}{\rightsquigarrow} y \underset{B}{\rightsquigarrow} \alpha \underset{D}{\rightsquigarrow} t$, where λ and α are an entrance and an exit of $C(x, y)$, respectively. The set of candidates of λ and α can be further reduced by considering only so-called "extreme" vertex pairs defined as follows. For $y \prec x$, a vertex pair (λ, α) is said to be extreme with respect to (x, y) if λ and α are an entrance and an exit of $C(x, y)$, respectively, and there is no entrance λ' and exit α' of $C(x, y)$ such that $\lambda \prec_B \lambda'$ or $\alpha' \prec_B \alpha$, where B is the boundary of $C(x, y)$. See Figure 2 for an illustration. Notice that for any $x, y \in V(D)$, there are at most four extreme vertex pairs. We show in Lemma 4 that there is a next-to-shortest path P' passing through an extreme vertex pair with respect to (x, y), where P'_{xy} is the maximal mixed subpath.

Lemma 4. *Let P be a next-to-shortest path with maximal mixed subpath P_{xy} such that $y \prec x$. There is a next-to-shortest path P' passing through an extreme vertex pair (λ, α) with respect to (x, y), and P_{xy} is the maximal mixed subpath of P'. In particular, P' is of the form $s \underset{D}{\rightsquigarrow} \lambda \underset{B}{\rightsquigarrow} x \underset{G}{\rightsquigarrow} y \underset{B}{\rightsquigarrow} \alpha \underset{D}{\rightsquigarrow} t$, where B is the boundary of $C(x, y)$.*

Proof. Let $P = P_{s\lambda'} \circ P_{\lambda' x} \circ P_{xy} \circ P_{y\alpha'} \circ P_{\alpha' t}$ with (λ', α') being an entrance-exit pair on P. By Corollary 3, there is a next-to-shortest path $P'' = P_{s\lambda'} \circ B_{\lambda' x} \circ P_{xy} \circ B_{y\alpha'} \circ P_{\alpha' t}$. Let (λ, α) be the extreme vertex pair with respect to (x, y) satisfying $\lambda' \preceq_B \lambda$ and $\alpha \preceq_B \alpha'$. We claim that there is a path $P' : s \underset{D}{\rightsquigarrow} \lambda \underset{B}{\rightsquigarrow} x \underset{P}{\rightsquigarrow} y \underset{B}{\rightsquigarrow} \alpha \underset{D}{\rightsquigarrow} t$, which is a next-to-shortest path. First, as $B_{\lambda x}$ and $B_{y\alpha}$ are subpaths of $B_{\lambda' x}$ and $B_{y\alpha'}$, respectively, $B_{\lambda x} \circ P_{xy} \circ B_{y\alpha}$ is a path. Second, by the definition of $C(x, y)$, any descendant of α not in $C(x, y)$ cannot be an ancestor of λ, and therefore any s, λ-path and α, t-path passing through only vertices in $D\backslash C(x, y)$ must be disjoint. ■

We are now ready to handle the case where $y \prec x$ by decomposing it into a 2-VDP and a shortest path problem. By Lemma 4, there is an optimal solution P of the form $s \underset{D}{\rightsquigarrow} \lambda \underset{B}{\rightsquigarrow} x \underset{G}{\rightsquigarrow} y \underset{B}{\rightsquigarrow} \alpha \underset{D}{\rightsquigarrow} t$, and the subpaths $P_{s\lambda}$ and $P_{\alpha t}$ are two disjoint paths in $D^{\lambda\alpha}_{xy}$, which is the subgraph induced by

$$(V(D)\backslash C(x, y)) \cup \{\lambda, \alpha\}.$$

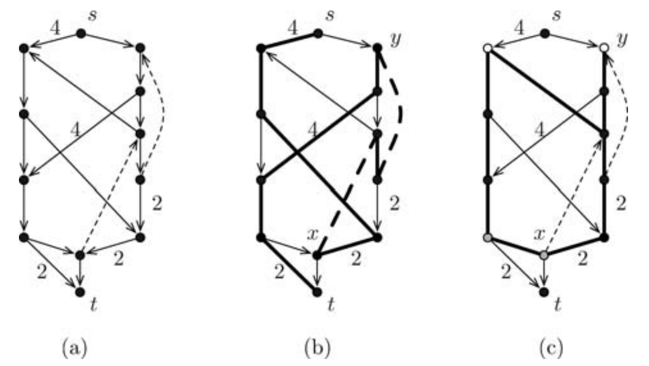

FIG. 3. Taking arbitrary y, x-paths as the boundary is infeasible. (a) A nonplanar digraph G with source s and destination t. Solid edges belong to D. (b) The only next-to-shortest path P, on which edges are thickened. The subpath P_{xy} is a maximal mixed subpath. (c) A "boundary" B of $C(x, y)$. The edges of B are thickened. The white vertices are the entrances of $C(x, y)$, given B as the boundary, while the grey ones are the exits. Each of the four entrance-exit pairs is not valid with respect to (x, y).

By Lemma 1 and Lemma 4, one may derive that the maximal mixed subpath P_{xy} is a shortest x, y-path in $H_{xy}^{\lambda\alpha}$, which is the subgraph induced by

$$(V \setminus V(D)) \cup (C(x, y) \setminus V(B_{\lambda x}) \setminus V(B_{y\alpha})) \cup \{x, y\}.$$

As a result, we conclude this section by the following theorem.

Theorem 1. *When G is a planar digraph with positive edge weights, the next-to-shortest path problem can be solved in $O(n^3)$ time.*

Proof. By Corollary 2, the case where $y \not\prec x$ can be solved in $O(n^2)$ time. For the case where $y \prec x$, the solution can be found by computing disjoint s, λ-path and α, t-path in $D_{xy}^{\lambda\alpha}$ and a shortest x, y-path in $H_{xy}^{\lambda\alpha}$. Assuming that $D_{xy}^{\lambda\alpha}$ and $H_{xy}^{\lambda\alpha}$ are given, both problems can be solved in $O(n)$ time as $D_{xy}^{\lambda\alpha}$ is a dag and $H_{xy}^{\lambda\alpha}$ is planar [8, 20].

For given $x, y \in V(D)$ with $y \prec x$, there are at most four extreme vertex pairs (λ, α) with respect to (x, y), which can be computed by identifying the boundary B of $C(x, y)$ and determining all the entrances and exits. The procedure given above can be done in $O(n)$ time with the embedding given in [4], and both $D_{xy}^{\lambda\alpha}$ and $H_{xy}^{\lambda\alpha}$ can then be computed in $O(n)$ time. As a result, the overall time complexity is $O(n^3)$ as there are $O(n^2)$ such possible (x, y) pairs. ∎

5. CONCLUDING REMARKS

In this article, we show that the next-to-shortest path problem on planar digraphs with positive edge weights can be solved in polynomial time. An important open problem is the time complexity for the problem on general digraphs with positive edge weights. We did not manage to exploit the properties presented in Section 3 for obtaining a polynomial-time algorithm to solve the next-to-shortest path problem on arbitrary digraphs with positive edge weights. For planar digraphs, we show that the mixed subpath cannot intersect the sub-boundaries as defined in Lemma 4. For nonplanar digraphs, the difficulty in applying Lemma 4 is that it is nontrivial to determine the "requested boundary" of $C(x, y)$ as in the planar case. Taking two arbitrary y, x-paths as the boundary is not feasible, as shown in Figure 3; however, enumerating all such paths is inefficient as there may be exponentially many y, x-paths in $G[C(x, y)]$. It is also interesting to know if the problem is polynomial-time solvable on a larger class of graphs than planar graphs.

ACKNOWLEDGMENT

The authors would like to thank Professor Douglas Shier and the anonymous reviewers, whose helpful comments improved the presentation of this paper.

REFERENCES

[1] M. Ahmed and A. Lubiw, Shortest paths avoiding forbidden subpaths, Networks 61 (2013), 322–334.

[2] A.V. Aho, J.E. Hopcroft, and J.D. Ullman, The design and analysis of computer algorithms, Addison-Wesley, Reading, Massachusetts, 1974.

[3] S.C. Barman, S. Mondal, and M. Pal, An efficient algorithm to find next-to-shortest path on trapezoid graphs, Adv Appl Math Anal 2 (2007), 97–107.

[4] N. Chiba and T. Nishizeki, A linear algorithm for embedding planar graphs using PQ-trees, J Comput Syst Sci 30 (1985), 54–76.

[5] M. Cygan, D. Marx, M. Pilipczuk, and M. Pilipczuk, The planar directed k-vertex-disjoint paths problem is fixed-parameter tractable, IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS), Berkeley, CA, 2013, pp. 197–206.

[6] D. Eppstein, Finding the k shortest paths, SIAM J Comput 28 (1998), 625–673.

[7] S. Fortune, J. Hopcroft, and J. Wyllie, The directed subgraph homeomorphism problem, Theor Comput Sci 10 (1980), 111–121.

[8] M.R. Henzinger, P. Klein, S. Rao, and S. Subramanian, Faster shortest-path algorithms for planar graphs, J Comput Syst Sci 53 (1997), 2–23.

[9] J. Hershberger, M. Maxel, and S. Suri, Finding the k shortest simple paths: A new algorithm and its implementation, ACM Trans Algorithms 3 (2007), Article 45.

[10] J. Hershberger, S. Suri, and A.M. Bhosle, On the difficulty of some shortest path problems, ACM Trans Algorithms 3 (2007), Article 5.

[11] K.-H. Kao, J.-M. Chang, Y.-L. Wang, and J. S.-T. Juan, A quadratic algorithm for finding next-to-shortest paths in graphs, Algorithmica 61 (2011), 402–418.

[12] I. Krasikov and S.D. Noble, Finding next-to-shortest paths in a graph, Inform Process Lett 92 (2004), 117–119.

[13] K.N. Lalgudi and M.C. Papaefthymiou, Computing strictly-second shortest paths, Inform Process Lett 63 (1997), 177–181.

[14] S. Li, G. Sun, and G. Chen, Improved algorithm for finding next-to-shortest paths. Inform Process Lett 99 (2006), 192–194.

[15] S. Mondal and M. Pal, A sequential algorithm to solve next-to-shortest path problem on circular-arc graphs, J Phys Sci 10 (2006), 201–217.

[16] N. Robertson and P.D. Seymour, Graph minors. XIII. The disjoint paths problem, J Comb Theory B 63 (1995), 65–110.

[17] A. Schrijver, Finding *k* disjoint paths in a directed planar graph, SIAM J Comput 23 (1994), 780–788.

[18] S. Szeider, Finding paths in graphs avoiding forbidden transitions, Discrete Appl Math 126 (2003), 261–273.

[19] T. Tholey, Solving the 2-disjoint paths problem in nearly linear time, Theor Comput Syst 39 (2006), 51–78.

[20] T. Tholey, Linear time algorithms for two disjoint paths problems on directed acyclic graphs, Theor Comput Sci 465 (2012), 35–48.

[21] G. Woeginger, A simple solution to the two paths problem in planar graphs, Inform Process Lett 36 (1990), 191–192.

[22] B.Y. Wu, A simpler and more efficient algorithm for the next-to-shortest path problem, Algorithmica 65 (2013), 467–479.

[23] B.Y. Wu, J.-L. Guo, and Y.-L. Wang, A linear time algorithm for the next-to-shortest path problem on undirected graphs with nonnegative edge lengths, 2012, arXiv:1203.5235 [cs.DS].

[24] J.Y. Yen, Finding the *K* shortest loopless paths in a network, Manage Sci 17 (1971), 712–716.

[25] J.Y. Yen, Another algorithm for finding the *K* shortest loopless network paths, Proceedings of 41st Mtg. Operations Research Society of America, New Orleans, 1972, Vol. 20, p. B/185.

[26] C. Zhang and H. Nagamochi, The next-to-shortest path in undirected graphs with nonnegative weights. Proceedings of 18th Computing: The Australasian Theory Symp (CATS), Melbourne, Australia, 2012, pp. 13–19.