On the Complexity of Paths Avoiding Forbidden Pairs

Petr Kolman and Ondřej Pangrác

Abstract

Given a graph $G = (V, E)$, two fixed vertices $s, t \in V$ and a set F of pairs of vertices (called *forbidden pairs*), the *problem of a path avoiding forbidden pairs* is to find a path from s to t that contains at most one vertex from each pair in F. The problem is known to be NP-complete in general and a few restricted versions of the problem are known to be in P. We study the complexity of the problem for directed acyclic graphs with respect to the structure of the forbidden pairs.

We write $x \prec y$ if and only if there exists a path from x to y and we assume, without loss of generality, that for every forbidden pair $(x, y) \in F$ we have $x \prec y$. The forbidden pairs have a *halving structure* if no two pairs $(u, v), (x, y) \in F$ satisfy $v \prec x$ or $v = x$ and they have a *hierarchical structure* if no two pairs $(u, v), (x, y) \in F$ satisfy $u \prec x \prec v \prec y$. We show that the PAFP problem is NP-hard even if the forbidden pairs have the halving structure and we provide a surprisingly simple and efficient algorithm for the PAFP problem with the hierarchical structure.

1 Introduction

The *problem of a path avoiding forbidden pairs* (PAFP) is defined as follows. Given a graph $G = (V, E)$ with two fixed vertices $s, t \in V$ and a set of pairs of vertices $F \subset (V \times V)$, the task is to find a path from s to t that contains at most one vertex from each pair in F, or to recognize that such path does

[1]Department of Applied Mathematics and Institute of Theoretical Computer Science (ITI), Charles University in Prague, Malostranské náměstí 25, 118 00 Prague, Czech Republic, E-mails: `kolman,pangrac@kam.mff.cuni.cz`

not exist. The pairs in the set F are called *forbidden pairs* and the paths containing at most one vertex from each pair in F are called F-paths.

The problem arose in the seventies in connection with automatic software testing and validation [5, 6] as the impossible pairs constrained path problem. The effort was to make the testing more efficient by considering only those paths through a program that contain at most one branch of each pair of mutually unexecutable branches. Gabow et al. [3] proved that the problem is NP-complete in directed acyclic graphs.

Yinnone [7] studied the problem in directed graphs under a so-called skew-symmetry condition constraining the set of edges and the set of forbidden pairs and described a polynomial time algorithm for instances satisfying the condition.

A restricted version of the PAFP problem appears also in bioinformatics, namely in the problem of protein identification via tandem mass spectrometry. For a protein with an unknown structure, the task is to determine the sequence of its amino acids from a mass spectrum of the protein (i.e., from a set of masses corresponding to masses of prefix and suffix fragments of the protein). Chen et al. [1] propose a method that, using information about the weights of amino acids, constructs a graph from the mass spectrum; the graph that they construct is a directed acyclic graph and the biological matter of the problem yields a linear ordering $<$ of the set of vertices that is consistent with the partial ordering induced by the edge set. From the mass spectrum they also derive a set of forbidden pairs with a specific structure: every two forbidden pairs (a, b) and (c, d) satisfy that either $a < c < d < b$ or $c < a < b < d$. They are looking for a path from the first to the last vertex that avoids the forbidden pairs. For such instances they describe an algorithm that runs in time polynomial in the size of the graph.

In this paper we study the complexity of the PAFP problem (in directed acyclic graphs) with respect to the structure of the forbidden pairs. Given a directed acyclic graph $G = (V, E)$ and two vertices $x, y \in V$, we write $x \prec y$ if and only if there exists a path from x to y in G, that is, we use $\prec$ to denote the partial ordering on V induced by the structure of the graph G. It will be convenient to assume, without loss of generality, that for every forbidden pair $(x, y) \in F$ we have $x \prec y$; if $y \prec x$ we replace the pair (x, y) by (y, x) and if x and y are not comparable, we remove the pair (x, y) from F without any effect on the solvability of the problem. The forbidden pairs in F have a *halving structure* (with respect to the ordering $\prec$) if no two forbidden pairs $(u, v), (x, y) \in F$ satisfy $v \prec x$ or $v = x$. The forbidden pairs in F have a *hierarchical structure* if no two pairs $(u, v), (x, y) \in F$ satisfy

$u \prec x \prec v \prec y$. For brevity we also speak about PAFP with the halving structure and PAFP with the hierarchical structure and about instances with the halving and the hierarchical structure.

We note that the problem considered by Chen et al. [1] is a special case of the PAFP with both the hierarchical and the halving structure.

The main contributions of the paper are a proof of NP-hardness of the PAFP problem even if the forbidden pairs have the halving structure (Section 3) and a surprisingly simple and efficient polynomial time algorithm for the PAFP problem with the hierarchical structure (Section 4). The algorithm yields solutions also for several optimization versions of the PAFP problem: search for the shortest or the longest path with the length defined as the sum of edge or vertex weights, or the search for k shortest or k longest paths.

2 Preliminaries

We assume that the input graph $G = (V, E)$ is a directed acyclic graph that is connected and we denote by $\prec$ the partial order on V induced by G. Further, we assume that s is the unique smallest element in V and t is the unique largest element in V, with respect to the partial order $\prec$; otherwise we delete vertices and edges that are not reachable from s and vertices and edges from which t is not reachable, without any effect on the solvability of the PAFP problem in G.

In the following observation we show that the PAFP problem is NP-complete even if both $G = (V, E)$ and (V, F) are planar; the proof is a simple modification of the original proof [3] of NP-completeness of the PAFP problem.

Observation 2.1 *The PAFP problem is NP-complete even if both $G = (V, E)$ and (V, F) are planar.*

Proof: The original proof is by a transformation from the 3-SAT problem. Given a formula $C = \bigwedge_{i=1}^{n}(x_{i,1} \vee x_{i,2} \vee x_{i,3})$ where each $x_{i,j}$ is either a variable or a negation of a variable, we construct a layered graph $G = (V, E)$ as follows. For $i \in [1, n]$, the i-th level consists of three vertices $x_{i,1}, x_{i,2}, x_{i,3}$, the level zero contains the source vertex s and the level $n+1$ contains the sink vertex t, and each pair of consecutive levels forms a complete bipartite graph (formally, we define a multiset $V = \{s, t\} \cup \bigcup_{i=1}^{n}\{x_{i,1}, x_{i,2}, x_{i,3}\}$ and $E = \{(s, x_{1,1}), (s, x_{1,2}), (s, x_{1,3}), (x_{n,1}, t), (x_{n,2}, t), (x_{n,3}, t)\} \cup \bigcup_{i=1}^{n-1}\{(x_{i,1}, x_{i+1,1}),$

$(x_{i,1}, x_{i+1,2})$, $(x_{i,1}, x_{i+1,3})$, $(x_{i,2}, x_{i+1,1})$, $(x_{i,2}, x_{i+1,2})$, $(x_{i,2}, x_{i+1,3})$, $(x_{i,3}, x_{i+1,1})$, $(x_{i,3}, x_{i+1,2})$, $(x_{i,3}, x_{i+1,3})\})$. The set of the forbidden pairs consists of all pairs $(x_{i,j}, x_{k,l})$ such that $i < k$ and $(x_{i,j}, x_{k,l})$ correspond to a variable and its negation. By construction, the formula C is satisfiable if and only if there exists a path from s to t avoiding the forbidden pairs.

We note that the constructed graph G can be made planar: add for every two consecutive layers i and $i+1$ a new vertex z_i and replace the complete bipartite graph $K_{3,3}$ on vertices $x_{i,1}, x_{i,2}, x_{i,3}, x_{i+1,1}, x_{i+1,2}, x_{i+1,3}$ by a graph with edges $\{(x_{i,1}, z_i), (x_{i,2}, z_i), (x_{i,3}, z_i), (z_i, x_{i+1,1}), (z_i, x_{i+1,2}), (z_i, x_{i+1,3})\}$. Observe that the transformation works analogously for any SAT instance. If we start with a SAT instance with the number of occurences per variable bounded by three (also an NP-complete problem [2]), we get an instance of the PAFP problem such that both (V, E) and (V, F) are planar. Thus, the PAFP is NP-complete even if $G = (V, E)$ is planar and (V, F) is planar.

□

In the next section, for technical reasons, we will work with a version of the PAFP problem (denoted !1-PAFP) in which every vertex, except for s and t, appears in exactly one forbidden pair. In the rest of this section we show that the problem remains NP-complete under this restriction.

Observation 2.2 *Let $G = (V, E)$, $F \subset V \times V$ and $s, t \in V$ be an instance of the PAFP problem. Then there exists an instance $G' = (V', E')$, $F' \subset V' \times V'$ and $s', t' \in V'$ of the !1-PAFP problem that has a solution if and only if the original instance has a solution. Further, the new instance can be constructed in time polynomial in V and F.*

Proof: The transformation from G to G' is done in two phases. In the first phase we ensure that every vertex appears in at most one forbidden pair. For every vertex $v \in V$ that belongs to more than one forbidden pair we perform step by step the following transfromation. Let $(v, u_1), \ldots, (v, u_h)$ be the forbidden pairs in which v appears as the first vertex and $(w_1, v), \ldots, (w_l, v)$ be the forbidden pairs in which v appears as the second vertex. We replace v by a directed path $x_l x_{l-1} \ldots x_1 y_h y_{h-1} \ldots y_1$ and reconnect the edges adjacent to v as follows. All edges incoming to v are directed to x_l, all edges outgoing from v are outgoing from y_1. Then, for each $i \in [1, h]$ we replace the forbidden pair (v, u_i) by (y_i, u_i) and for each $j \in [1, l]$ we replace (w_i, v) by (w_i, x_i). At this point each vertex appears in at most one forbidden pair.

In the second phase we get rid of all vertices that do not belong to any forbidden pair, except for s and t. If v is such vertex and $(u_1, v), \ldots, (u_h, v)$,

$(v, w_1), \ldots, (v, w_l)$ are edges adjacent to v, we remove v from V and from replace the edges adjacent to v by a complete bipartite graph on $\{u_1, \ldots, u_h\}$ and $\{w_1, \ldots, w_l\}$, that is, by edges $\{(u_i, w_j) \mid i \in [1, h],\ j \in [1, l]\}$. This is done for all vertices not appearing in any forbidden pair, except for s and t.

$\square$

3 Forbidden pairs with halving structure

As a tool for proving NP-completeness of the PAFP problem with the halving structure, we define a new problem called the *red-blue path problem*: Given a directed graph $G = (V, E)$ and a partitioning of its edges into two sets $E = E_R \cup E_B$ such that both $G_R = (V, E_R)$ and $G_B = (V, E_B)$ are acyclic graphs, the task is to find two node disjoint $s - t$ paths p_R and p_B in G such that $p_R \subset E_R$ and $p_B \subset E_B$ (for notational simplicity we view a path as a set of edges). The edges in E_R are called *red* and the edges in E_B are *blue*. We show that the red-blue path problem is closely related to the PAFP problem.

Proposition 3.1 *The red-blue path problem is NP-complete.*

Proof: The proof is by reduction from the !1-PAFP problem. Given an instance of the !1-PAFP problem, namely a directed graph $G = (V, E)$, $s, t \in V$ and a set of forbidden pairs $F = \{(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l)\}$ such that each vertex appears in exactly one forbidden pair (except for s and t), we construct an instance $G' = (V', E_R \cup E_B)$ of the red-blue path problem as follows. The vertex set G' will be the same as for G, that is, $V' = V$. The set of red edges will consists of edges $E_R = \{(s, x_1), (s, y_1), (x_l, t), (y_l, t)\} \cup \bigcup_{i=1}^{l-1}\{(x_i, x_{i+1}), (x_i, y_{i+1}), (y_i, x_{i+1}), (y_i, y_{i+1})\}$ and the set of blue edges will coincide with the set of original edges in G, that is, $E_B = E$.

It follows from the construction that if there exists a path p between s and t in G avoiding the forbidden pairs in F, then there exist the two node disjoint paths $p_B \subset E_B$ and $p_R \subset E_R$ in G' (specifically, $p_B = p$), and vice versa. The reduction is clearly a polynomial-time reduction and the proof is completed.

$\square$

Theorem 3.2 *The PAFP problem with the halving structure is NP-complete.*

Proof: The proof is by reduction from the red-blue path problem. Given an instance $G = (V, E_R \cup E_B)$ and $s, t \in V$ of the red-blue path problem, we construct an instance $G' = (V', E')$, $s', t' \in V'$ and $F \subset V' \times V'$ of the PAFP problem with the halving structure. The vertex set of the graph G' consists of two copies of the vertex set of the graph G; let V_1 denote the first copy and V_2 the second copy and let v_1 denote the copy of $v \in V$ in V_1 and v_2 the copy of $v \in v$ in V_2. The core of G' constitute two graphs (V_1, E_R) and (V_2, E_B) connected by an edge (t_1, s_2), that is, $V' = V_1 \cup V_2$, $E' = E_1 \cup E_2 \cup \{(t_1, s_2)\}$, $s' = s_1$ and $t' = t_2$. The set of forbidden pairs consists of all pairs (v_1, v_2) except for the pairs (s_1, s_2) and (t_1, t_2), that is, $F = \bigcup_{v \in V \setminus \{s,t\}} \{(v_1, v_2)\}$.

By construction, if there exists a path p in G' from s' to t', then it has a form $p = s_1 \cdots t_1 s_1 \cdots t_2$ and the first part $s_1 \cdots t_1$ uses only the edges from E_R and the second part $s_2 \cdots t_2$ uses only the edges from E_B and, moreover the forbidden pairs ensure that if the first parts uses a vertex v_1 then the other part does not use the vertex v_2, for each $v \in V \setminus \{s, t\}$, and vice versa. Thus, if we interpret the first part of the path p as the red path in G and the second part as the blue path in G, then they have the desired property. On the other hand, given a red path and a node disjoint blue path in G, they a concatenation of their copies in G' yields a solution for the PAFP problem in G'. Since the reduction is clearly a polynomial-time reduction, the proof is completed.

$\square$

Although the PAFP problem with the halving structure is NP-complete, there exists a natural version of the PAFP with the halving structure that is solvable in polynomial time. This version was already briefly discussed in Introduction and the instances of it have both the halving and the hierarchical structure (and some other restricting properties). Let V be a linearly ordered set consisting of $2k+2$ vertices $s < x_1 < x_2 < \ldots < x_k < y_k < \ldots < y_2 < y_1 < t$, $G = (V, E)$ be a directed graph such that the partial order $\prec$ induced by G is consistent with $<$ (i.e., $a < b$ for each edge $(a, b) \in E$) and $F = \{(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)\}$ be a set of forbidden pairs. As mentioned in Introduction, such instances are important in the problem of protein identification and Chen et al. [1] proposed a polynomial time algorithm for solving them. In the rest of this section we enlarge the set of instances of the PAFP with the halving structure for which a polynomial time algorithm is known.

Let σ be a permutation of $1, 2, \ldots, k$. We say that an instance $G = (V, E)$, F (with $|V| = 2k+2$) belongs to the class PAFP(σ) if there exists a linear ordering of V consistent with $\prec$ such that, after a suitable relabeling of the vertices, $s < x_1 < x_2 < \ldots < x_k < y_{\sigma(1)} < y_{\sigma(2)} < \ldots y_{\sigma(k)} < t$ and $F = \{(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)\}$. For a permutation σ of $1, 2, \ldots, k$ we denote by $\overline{\sigma}$ the reversed permutation (i.e., $\overline{\sigma}(i) = \sigma(k+1-i)$, for each $i \in [1, k]$).

In the following theorem, by an *oracle* for a problem we mean anything that provides correct solutions for instances of the problem in constant time.

Proposition 3.3 *Let S be a class of permutations and let A_S be an oracle for instances from $\bigcup_{\sigma \in S}$ PAFP(σ). For every $\sigma \in S$, every instance from PAFP($\overline{\sigma}$) can be solved in polynomial time with the oracle A_S.*

Proof: Consider a permutation $\sigma \in S$ and an instance from PAFP($\overline{\sigma}$), that is, a graph $G = (V, E)$ and a set of forbidden pairs F with the property that there exists a linear ordering on V consistent with $\prec$ such that after a suitable relabeling the vertices from V satisfy $s < x_1 < x_2 < \ldots < x_k < y_{\sigma(1)} < y_{\sigma(2)} < \ldots < y_{\sigma(k)} < t$ and the set of the forbidden pairs has the form $F = \{(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)\}$.

If $(s, t) \in E$, the problem is trivial. Otherwise we perform a series of queries to the A_S-oracle for instances derived from G and F. If any of the answers is positive, then there exists an F-path in G, and if none of the answers is positive, then there is no F-path in G.

We start by constructing a set of edges E' by a modification of E: we delete all edges starting in s, all edges ending in t and all edges of the form (x_i, y_j), and then we reverse the direction of the edges induced by the vertices $V_y = \{y_1, y_2, \ldots, y_k\}$. Note that the graph (V, E') with the set of forbidden pairs F is an instance from PAFP(σ). If there exists an edge $(x_i, t) \in E$, for some i, we construct a new graph $G_t = (V, E_t)$ by setting $E_t = E' \cup \bigcup_{(x_i,t) \in E} \{(x_i, t)\} \cup \bigcup_{(s,x_i) \in E} \{(s, x_i)\}$ and ask the A_S-oracle for the existence of an F-path in G_t. Analogously, if there exists an edge (s, y_i), for some i, then we construct $G_s = (V, E_s)$ by setting $E_s = E' \cup \bigcup_{(s,y_j) \in E} \{(y_j, t)\} \cup \bigcup_{(y_j,t) \in E} \{(s, y_j)\}$ and ask the A_S-oracle for the existence of an F-path in G_s.

Finally, for each edge of the form $(x_i, y_j) \in E$ we construct a graph G_{ij} by setting $E_{ij} = E' \cup \bigcup_{(s,x'_i) \in E} \{(s, x'_i)\} \cup \{(y_j, t)\} \cup \bigcup_{(y'_j,t) \in E} \{(x_i, y'_j)\}$ and ask the A_S-oracle for the existence of an F-path in $G_{ij} = (V, E_{ij})$.

If none of the particular problems has a solution then there is no F-path from s to t in the graph G.

$\square$

Note that the version of the problem considered by Chen et al. [1] belongs to the class PAFP($\bar{\iota}$) where ι is the identity permutation. Thus, we get the following corollary.

Corollary 3.4 *There exists a polynomial time algorithm for problems from PAFP(ι).*

4 Forbidden pairs with hierarchical structure

In this section we describe a polynomial time algorithm for the PAFP problem with the hierarchical structure. The input of the algorithm is a directed acyclic graph $G = (V, E)$, a set of forbidden pairs $F \subset V \times V$ with the hierarchical structure (i.e., no two pairs $(a, b), (c, d) \in F$ satisfy $a \prec c \prec b \prec d$) and a weight function $w : E \to \mathbb{R}$ on the edges. The goal is to find an path from s to t that avoids the forbidden pairs from F and has the maximum weight (the weight of a path is the sum of the edge weights of the path). We recall our assumption that s is the unique minimal element and t is the unique maximal element in V, with respect to the partial order $\prec$.

On a high level, the algorithm repeatedly shrinks the input instance by removing vertices, edges and forbidden pairs while guaranteeing that one can obtain a solution for the original instance from a solution for the new (smaller) instance. For this purpose, the algorithm maintains a label $\rho(e)$ for every edge in the current graph; the labels are strings and initially, the label of an edge e is $\rho(e) = e$. The label of an edge e encompasses information about a path in the original graph that is now represented by the edge e; the details are described bellow.

The algorithm iteratively applies the following three reduction rules. For notational simplicity, we use G and F to denote the current graph and the current set of forbidden pairs.

Rule R1 – contraction of a vertex.

Step by step, for every vertex $x \in V \setminus \{s, t\}$ that does not appear in any forbidden pair in F, do the following. Remove x from V, and for every pair of edges $(u, x), (x, y) \in E$ add a new edge (u, y) to E. The label of the new edge is a concatenation of the labels of the two old edges, that is, $\rho(u, y) = \rho(u, x)\rho(x, y)$. Similarly, we define the weight of the new edge as the sum of the weights of the two old edges, that is, $w(u, y) = w(u, x) + w(x, y)$. Remove all edges adjacent to x.

If two parallel edges (u, v) emerge (i.e., an edge (u, v) was present in G before the reduction), we compare their weights and keep only the edge with the larger weight.

Rule R2 – removal of an edge.

For every edge $e \in E \cap F$, remove e from E.

Rule R3 – removal of a forbidden pair.

For every $(x, y) \in F$ such that $x \not\prec y$ (i.e., there is no directed path from x to y), remove (x, y) from F.

We are ready to describe the algorithm.

Input: a directed acyclic graph $G = (V, E)$, two distinct vertices $s, t \in V$, a set of forbidden pairs $F \subset V \times V$ with the hierarchical structure, an edge weight function $w : E \to \mathbb{R}$.

Alg.: **for each** edge $(x, y) \in E$ set $\rho(x, y) = (x, y)$.
while $|V| > 2$ **do**
apply **R1**
apply **R2**
apply **R3**
if $E = \emptyset$ output "No F-path from s to t exists."
else output "The maximum weight of an F-path from s to t is $w(s, t)$ and the corresponding path is $\rho(s, t)$."

To establish the correctness of the algorithm, we will prove two things. First, after an application of any of the rules, there is way to obtain a solution for the original instance from the solution for the new (smaller) instance. Second, if none of the rules is applicable, then the graph consists of the two vertices s and t only (with or without an edge between them) and the problem is trivial. Before proving these two properties formally in lemmas, it will be convenient to introduce an additional notation. *A label* $\rho(P)$ of a path P consisting of k edges $e_1, e_2, \ldots, e_k$ is a concatenation of the k labels $\rho(e_i)$, that is, $\rho(P) = \rho(e_1)\rho(e_2)\ldots\rho(e_k)$. Observe that in the original graph with edge labels $\rho(e) = e$, the label $\rho(P)$ of a path P is exactly the description of the path.

Lemma 4.1 *Let $G = (V, E)$ be a directed acyclic graph and F a set of forbidden pairs with the hierarchical structure. Let G' be the graph obtained from G by an application of the reduction rule* **R1** *(***R2***, or* **R3***, resp.). Then* *for any F-path P' in G' there is an F-path P in G such that $\rho(P) = \rho(P')$ and $w(P) = w(P')$. On the other hand, if P is an F-path in G between s and t then there exists an F-path P' in G' between s and t such that $w(P) \leq w(P')$.*

Proof: We start by arguing about the rule **R1**. Without loss of generality we assume that the rule was applied to a vertex x only.

Consider an F-path P' in G' that uses only edges appearing also in the graph G. Then P' is also a path in G and the first claim in the lemma is trivial. Similarly, if P is a path in G that does not use the vertex x, then P is a path in G' too and the second claim in the lemma is trivial.

Now, consider an F-path P' in G' that contains a new edge $(u, v) \notin E$ that was introduced by the application of the rule **R1**. By replacing the edge (u, v) in P' by two edges $(u, x), (x, v)$, we obtain a path P in G and clearly P avoids all forbidden pairs F; it follows from the description of the rule **R1** that $\rho(P) = \rho(P')$ and $w(P) = w(P')$.

Finally, let P be an F-path in G containing the vertex x and let (u, x) and (x, v) be two consecutive edges of P, adjacent to x. If $(u, v) \in E$ and $w(u, v) \geq w(u, x) + w(x, v)$, then a path derived from P by replacing $(u, x), (x, v)$ by (u, v) is the desired F-path P' in G'. Otherwise, an edge (u, v) with a weight $w(u, x) + w(x, v)$ and a label $\rho(u, x)\rho(x, v)$ appears in G' and the desired F-path P' is derived from P by replacing the two edges $(u, x), (x, v)$ by (u, v); clearly $w(P') = w(P)$ (and even $\rho(P) = \rho(P')$) which completes the proof.

Concerning the other two reduction rules, it is easy to see that they do not effect the solution of the problem.

□

Proposition 4.2 *Let $G = (V, E)$ be a directed acyclic graph and F a set of forbidden pairs with the hierarchical structure. If none of the reduction rules* **R1**, **R2**, **R3** *can be applied then $V = \{s, t\}$.*

Proof: The proof is by contradiction. Assume that there exists a vertex $x \in V$ distinct from s, t. Since the rule **R1** is not applicable, the vertex x is contained in an F-pair, and therefore $F \neq \emptyset$. Note that for each F-pair (u, v) there is a path from u to v since otherwise we can apply the rule **R3**. Consider a pair $(u, v) \in F$ such that no other pair $(x, y) \in F$ satisfies $u \prec x \prec y \prec v$. Since the rule **R2** is not applicable, there must be an internal vertex, say a vertex x, on the shortest path from u to v. But since the rule **R1** is not applicable, there is another vertex y such that $(x, y) \in F$ or $(y, x) \in F$, and the hierarchical structure of F implies that y is also on the shortest path from u to v, a contradiction.

□

The two lemmas imply the following theorem.

Theorem 4.3 *There exists a polynomial time algorithm for the PAFP problem with the hierarchical structure.*

We remark that a slight modification the algorithm yields an algorithm for variants of the PAFP problem in which we are interested in finding the k shortest or the k longest F-path (for both edge and vertex lengths). It suffice to keep for every edge (vertex) k labels and k weights (initially all k labels equal and all k weights equal for each edge or vertex) and to modify the reduction rule **R1** appropriately.

Fast implementation A naive implementation requires time $O(n^2)$ for a contraction of a single vertex and $O(1)$ for a removal of a single edge; the most time consuming part of a naive implementation is the recognition of redundant forbidden pairs and requires time $O(m)$ per forbidden pairs containing a given vertex. Since the total number of iterations is at most n (in each iteration is contracted at least one vertex) the overall running time is then upperbounded by $O(mn^2)$. By exploiting algorithms for fast matrix multiplication for the recognition of the redundant forbidden pairs, we can reduce the time bound down to about $O(n^{3.3})$. We are going to sketch how to implement the algorithm in time $O(n^3)$.

Italiano [4] described a data structure supporting a sequence of reachability queries and deletion operations in a directed acyclic graph running in time $O(mn + qn)$ where q denotes the number of reachability queries, n the number of nodes in the graph and m the initial number of edges. An inspection of the algorithm reveals that a simple extension yields an algorithm supporting in addition the operation of a vertex contraction in time $O(n^2)$ per contraction (note that a vertex contraction does not affect the reachability within the graph, except for the contracted vertex). With such an algorithm (and a relevant data structure) we implement our algorithm in time $O(n^3)$ as follows. Since there are at most n vertex contractions, the total time for all vertex contractions is $O(n^3)$. A minor drawback of the additional operation of a vertex contraction is that a deleted edge may appear in the graph again, resulting a need to remove the edge several times.

However, the total number of all edge removals can be bounded by $O(n^2)$, and as the amortized complexity of an edge removal is $O(n)$, we need at most $O(n^3)$ time for all edge removals. The recognition (and removal) of all redundant forbidden pairs requires time $O(n^3)$. We conclude that the running time of the algorithm is $O(n^3)$.

Acknowledgement We thank Liam Roditty for pointing us to the paper [4].

References

[1] T. Chen, Ming-Yang Kao, M. Tepel, J. Rush, G. M. Church. A Dynamic Programming Approach to De Nova Peptide Sequencing via Tandem Mass Spectrometry, Journal of Computational Biology 8(3) (2001), 325–337.

[2] V. Dalmau and D. K. Ford. Generalized satisfiability with limited occurrences per variable: A study through delta-matroid parity. In *Mathematical Foundations of Computer Science 2003*, volume 2747 of *Lecture Notes in Computer Science*, pages 358–367, 2003.

[3] H. N. Gabow, S. N. Maheshwari, L. J. Osterweil. On two problems in the generation of program test paths, IEEE Trans. Software Egeneering SE-2 (1976), 227–231.

[4] G. F. Italiano. Finding Paths and Deleting Edges in Directed Acyclic Graphs. Inf. Process. Lett. 28(1), (1988), 5–11.

[5] K. W. Krause, R. W. Smith, M. A. Goodwin, Optimal software test planning through automated network analysis, Proceedings 1973 IEEE Symp. Computer Software Reliability, New York (1973), 18–22.

[6] Pradip K. Srimani, Bhabani P. Sinha, Impossible pair constrained test path generation in a program, Information Sciences 28(2) (1982), 87–103.

[7] H. Yinnone, On path avoiding forbidden pairs of vertices in a graph, Discrete Applied Math. 74 (1997), 85–92.