Information Processing Letters 140 (2018) 8–12

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

# Long directed $(s,t)$-path: FPT algorithm

Fedor V. Fomin [a], Daniel Lokshtanov [a], Fahad Panolan [a], Saket Saurabh [a,b], Meirav Zehavi [c]

[a] University of Bergen, Bergen, Norway
[b] The Institute of Mathematical Sciences, HBNI, Chennai, India
[c] Ben-Gurion University, Beersheba, Israel

ARTICLE INFO

*Article history:*
Received 13 January 2018
Received in revised form 8 April 2018
Accepted 10 April 2018
Available online 28 May 2018
Communicated by Marcin Pilipczuk

*Keywords:*
Parameterized algorithm
Long directed $(s,t)$-path
Long directed cycle

ABSTRACT

Given a digraph $G$, two vertices $s,t \in V(G)$ and a non-negative integer $k$, the Long Directed $(s,t)$-Path problem asks whether $G$ has a path of length at least $k$ from $s$ to $t$. We present a simple algorithm that solves Long Directed $(s,t)$-Path in time $\mathcal{O}^\star(4.884^k)$. This results also in an improvement upon the previous fastest algorithm for Long Directed Cycle.



## 1. Introduction

Given a digraph (directed graph) $G$, two vertices $s,t \in V(G)$ and a non-negative integer $k$, the Long Directed $(s,t)$-Path problem asks whether $G$ has an $(s,t)$-path (i.e. a path from $s$ to $t$) of length *at least* $k$.[1] Here, the term *length* refers to the number of vertices on the path, and paths are assumed to be directed simple paths. Observe that Long Directed $(s,t)$-Path and Directed $k$-$(s,t)$-Path are not equivalent problems, where that latter problem asks whether $G$ has an $(s,t)$-path of length *exactly* $k$. Indeed, $G$ may not have any $(s,t)$-path of length exactly $k$, or more generally, it may not even have any $(s,t)$-path of "short" length (say, length $10k$), but it may have an $(s,t)$-path of "long" length (say, $100k$). For example, the only $(s,t)$-path of length at least $k$ in $G$ might be a Hamiltonian path.

In the classic Directed $k$-Path problem, the objective is to determine whether $G$ has a path of length at least $k$. Both Long Directed $(s,t)$-Path and Directed $k$-$(s,t)$-Path are generalizations of this problem. For the Directed $k$-Path problem, a large number of algorithms have been developed over the years (for some recent developments, see [2,8,1,3,7,4,9]). Currently, the fastest deterministic algorithm for this problem runs in time $\mathcal{O}^\star(2.597^k)$ [9]. Notably, algorithms for Directed $k$-Path implicitly solve Directed $k$-$(s,t)$-Path. However, in the case of Long Directed $(s,t)$-Path, these algorithms do not solve the problem. To substantiate the difficulty posed by Long Directed $(s,t)$-Path, let us consider the related Long Directed Cycle and Directed $k$-Cycle problems. The first problem asks whether $G$ has a cycle of length at least $k$, while the second problem asks whether $G$ has a cycle of length exactly $k$. It has been known how to solve Directed $k$-Cycle in time $\mathcal{O}^\star(2^{\mathcal{O}(k)})$ already in 1994.[2] In contrast, only in 2014 was it first known how to solve Long Directed Cycle in time $\mathcal{O}^\star(2^{\mathcal{O}(k)})$ [5] (previously, this problem was only

*E-mail addresses:* fomin@ii.uib.no (F.V. Fomin), daniello@ii.uib.no (D. Lokshtanov), fahad.panolan@ii.uib.no (F. Panolan), saket@imsc.res.in (S. Saurabh), meiravze@bgu.ac.il (M. Zehavi).

[1] In this paper, we consider only digraphs (our algorithm also works for undirected graphs).

[2] The standard notation $\mathcal{O}^\star$ is used to hide factors polynomial in the input size.

https://doi.org/10.1016/j.ipl.2018.04.018
0020-0190/

known to be solvable in time $\mathcal{O}^\star(k^{\mathcal{O}(k)})$ [6]). Currently, the fastest deterministic and randomized algorithms for Long Directed Cycle run in times $\mathcal{O}^\star(6.74^k)$ and $\mathcal{O}^\star(4^k)$, respectively [10].

In this work, we present a very simple (deterministic) algorithm for Long Directed $(s,t)$-Path that runs in time $\mathcal{O}^\star(4.884^k)$. (Our algorithm is the first algorithm that solves Long Directed $(s,t)$-Path.) We remark that as our algorithm invokes an algorithm for Directed $k$-$(s,t)$-Path as a black box, a faster deterministic algorithm for Directed $k$-$(s,t)$-Path than the current state-of-art (that is, the previously mentioned $\mathcal{O}^\star(2.597^k)$-time algorithm [9]) would also directly speed-up our algorithm. As a consequence of our result, we also obtain a deterministic algorithm that solves Long Directed Cycle in time $\mathcal{O}^\star(4.884^k)$, improving upon the previous best $\mathcal{O}^\star(6.74^k)$-time deterministic algorithm for this problem. We remark that our algorithm revisits ideas introduced in the papers [5] and [10], and employs them in a manner that is (a) clean and simple, and (b) results in a faster running time for Long Directed Cycle.

*Overview* Let us give a short, informal overview of our algorithm and its proof. First, if there exists an $(s,t)$-path in $G$ of length at least $k$ and at most $\alpha k$ (for some $1.5 \le \alpha \le 2$), then we detect it using a known algorithm for Directed $k$-$(s,t)$-Path (in Section 4). Else, we aim to reduce our problem to a simpler variant of it that is solvable in polynomial time. Let us first explain this simpler variant (handled in Section 2). In this "balanced" variant, it is "guaranteed" that if the input graph has a solution (of length at least $k$), then all its solutions have length at least $2k$. Additionally, if it has a solution, then there is a solution where the first $k$ vertices are "annotated by $L$", and the last $k$ vertices are annotated by $R$. The simplicity of this variant is observed by two lemmas with respect to a shortest "well-annotated" $k$-path $P$ (the statements are given with $\alpha$ rather than 2 in order to reuse them later): the first states that the shortest path (in the input graph) between the first and the $(\alpha-1)k$-th vertex on $P$ is $(\alpha-1)k$, and the second states that given a shortest path between the first and the $(\alpha-1)k$-th vertex on $P$, this path does not contain any vertex on the suffix of $P$ that starts at its $(\alpha-1)k+1$-th vertex. Together, we show that these claims imply that our variant can be solved by a simple usage of BFS (to find the beginning of a solution) and a reachability test (to find its end).

The crux of the algorithm lies in the reduction of the general problem (once we know that no solution on at most $\alpha k$ vertices exists). Here, we use two levels of annotations. First, we annotate the first $(\alpha-1)k$ vertices of a solution by $L'$ and the last $k$ vertices of that solution by $R'$ (in Section 4). Specifically, we use the tool of universal sets (see Preliminaries) to ensure that if there is a solution, then we will annotate it well (at least once and using an exponential number of tries). We call the problem that results (informally, the original problem where the sought solution should respect our annotations) an "unbalanced" variant of our problem. Then, to solve this unbalanced variant (in Section 3), we first "guess" the $(\alpha-1)k$-th vertex on the sought solution and compute a shortest path from $s$ to it. Then, we remove the vertices of this path from the graph (this removal is justified by relying on the two lemmas mentioned earlier), and annotate it using the tool of universal sets. In this second level of annotations, we aim to annotate the first $(2-\alpha)k$ vertices of the sought solution by $L$, and the last $(2-\alpha)k$ vertices of it by $R$. At this point, we actually have the simpler variant at hand, since we only need to find a solution on at least $(2-\alpha)k$ vertices, and we know that there is no solution on at most $2(2-\alpha)k$ vertices (since $\alpha \ge 1.5$).

*Preliminaries* Given a graph $G$, let $V(G)$ and $E(G)$ denote the vertex and edge sets of $G$, respectively, and denote $n=|V(G)|$. For a set $A \subseteq V(G)$, let $G[A]$ denote the subgraph of $G$ induced by $A$, and define $G-A$ as $G[V(G)\setminus A]$. Given two vertices $s,t \in V(G)$ and an integer $k$, let $\Lambda^k_G(s,t)$ denote the minimum length of an $(s,t)$-path in $G$ whose length is at least $k$, where $\Lambda^k_G(s,t)=-\infty$ if no such path exists.

For a universe $U$, we let $2^U$ denote the family of all subsets of $U$. Our algorithm relies on the notion of universal set:

**Definition 1.1.** Let $U$ be an $n$-element universe, and $p,q \in \mathbb{N}_0$. A family $\mathcal{F} \subseteq 2^U$ is an $(n,p,q)$*-universal set* if for all disjoint $A,B \subseteq U$ such that $|A| \le p$ and $|B| \le q$, there exists $F \in \mathcal{F}$ such that $A \subseteq F$ and $B \cap F = \emptyset$.

It is known that small universal sets can be computed efficiently:

**Proposition 1.1** *([5]). Given an $n$-element universe, and $p,q \in \mathbb{N}_0$, an $(n,p,q)$-universal set $\mathcal{F}$ of size $\mathcal{O}(\binom{p+q}{p}2^{o(p+q)} \cdot \log n)$ can be computed in time $\mathcal{O}(\binom{p+q}{p}2^{o(p+q)} \cdot n\log n)$.*

## 2. Balancedly annotated long $(s,t)$-paths

The purpose of this section is to handle the special case of Long Directed $(s,t)$-Path where it is assumed that no "short" path of length at least $k$ exists (that is, a path of length shorter than $2k$ but at least $k$), and that the prefix and suffix of a solution (if one exists) are "annotated". Here, by annotating a solution we mean that its $k$ first vertices (those closest to $s$) belong to a set $L$, and its last $k$ vertices belong to a set $R$. Specifically, we prove the following lemma.

**Lemma 2.1.** *There is a deterministic polynomial-time algorithm,* Alg1*, that given an instance $(G,s,t,k)$ of* Long Directed $(s,t)$-Path*, and a partition $(L,R)$ of $V(G)$, satisfies the following.*[3]

- *If $\Lambda^k_G(s,t) \ge 2k$ and $G$ has an $(s,t)$-path $s=v_1 \to v_2 \to \cdots \to v_\ell = t$ such that $\ell = \Lambda^k_G(s,t)$, $v_1,v_2,\ldots,v_k \in L$ and $v_{\ell-k+1},v_{\ell-k+2},\ldots,v_\ell \in R$, then* Alg1 *accepts.*
- *If $\Lambda^k_G(s,t)=-\infty$, then* Alg1 *rejects.*

[3] In cases not covered by these conditions, Alg1 can either accept or reject.

Towards the proof of this lemma, we need to establish two results. We prove them in a general form in order to reuse them in the next section.

**Lemma 2.2.** *Fix $1 \leq \alpha$. Let $(G,s,t,k)$ be an instance of* Long Directed $(s,t)$-Path*, and $(L,R)$ be a partition of $V(G)$. Suppose that $\Lambda^k_G(s,t) \geq \lceil \alpha k \rceil$, and $G$ has an $(s,t)$-path $s = v_1 \to v_2 \to \cdots \to v_\ell = t$ such that $\ell = \Lambda^k_G(s,t)$, $v_1, v_2, \ldots, v_{\lceil(\alpha-1)k\rceil} \in L$ and $v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell \in R$. Then, the length of a shortest path from $v_1$ to $v_{\lceil(\alpha-1)k\rceil}$ in $G[L]$ is $\lceil(\alpha-1)k\rceil$.*

**Proof.** Let $P$ be a shortest $(v_1, v_{\lceil(\alpha-1)k\rceil})$-path in $G[L]$. It is clear that $|V(P)| \leq \lceil(\alpha-1)k\rceil$. Thus, to prove that $|V(P)| = \lceil(\alpha-1)k\rceil$, suppose by way of contradiction that $|V(P)| \leq \lceil(\alpha-1)k\rceil - 1$. Denote $P = u_1 \to u_2 \to \cdots \to u_r$, where $s = u_1$ and $v_{\lceil(\alpha-1)k\rceil} = u_r$. Since $v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell \in R$, we have that $V(P) \cap \{v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell\} = \emptyset$. Then, $s = u_1 \to u_2 \to \cdots \to u_r \to v_{k+1} \to v_{k+2} \to v_{\ell-k}$ is a walk in $G$ that avoids the vertices $v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell$. In particular, as $r \leq \lceil(\alpha-1)k\rceil - 1$, this means that $G$ has an $(s, v_{\ell-k})$-path of length at most $\ell - k - 1$ that avoids the vertices $v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell$. By traversing this path and then the path $v_{\ell-k} \to v_{\ell-k+1} \to \cdots \to v_\ell$, we exhibit an $(s,t)$-path in $G$ of length strictly smaller than $\ell$ (but of length at least $k$ and where the last $k$ vertices belong to $R$), which is a contradiction to $\ell = \Lambda^k_G(s,t)$. □

**Lemma 2.3.** *Fix $1 \leq \alpha$. Let $(G,s,t,k)$ be an instance of* Long Directed $(s,t)$-Path*, and $(L,R)$ be a partition of $V(G)$. Suppose that $\Lambda^k_G(s,t) \geq \lceil \alpha k \rceil$, and $G$ has an $(s,t)$-path $s = v_1 \to v_2 \to \cdots \to v_\ell = t$ such that $\ell = \Lambda^k_G(s,t)$, $v_1, v_2, \ldots, v_{\lceil(\alpha-1)k\rceil} \in L$ and $v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell \in R$. Then, for any path $P$ of length $\lceil(\alpha-1)k\rceil$ from $v_1$ to $v_{\lceil(\alpha-1)k\rceil}$ in $G[L]$, there exists a path from $v_{\lceil(\alpha-1)k\rceil}$ to $v_\ell$ in $G - (V(P) \setminus \{v_{\lceil(\alpha-1)k\rceil}\})$ of length at least $k+1$.*

**Proof.** Let $P$ be a $(v_1, v_{\lceil(\alpha-1)k\rceil})$-path of length $\lceil(\alpha-1)k\rceil$ in $G[L]$. Since $v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell \in R$, to prove the lemma it is sufficient to show that $P$ does not contain any vertex from $\{v_{\lceil(\alpha-1)k\rceil+1}, v_{\lceil(\alpha-1)k\rceil+2}, \ldots, v_{\ell-k}\}$. Suppose, by way of contradiction, that this claim is false, and let $i$ be the largest index of a vertex in $\{v_{\lceil(\alpha-1)k\rceil+1}, v_{\lceil(\alpha-1)k\rceil+2}, \ldots, v_{\ell-k}\}$ such that $v_i \in V(P)$. Now, consider the path obtained by traversing $P$ from $v_1$ until $v_i$, then traversing $v_i \to v_{i+1} \to \cdots \to v_{\ell-k}$, and finally traversing $v_{\ell-k} \to v_{\ell-k+1} \to \cdots \to v_\ell$. Notice that this is an $(s,t)$-path in $G$ of length strictly smaller than $\ell$ (but of length at least $k$ and where the last $k$ vertices belong to $R$), which is a contradiction to $\ell = \Lambda^k_G(s,t)$. □

We are now ready to prove Lemma 2.1.

**Proof of Lemma 2.1.** Let $(G,s,t,k)$ be an instance of Long Directed $(s,t)$-Path, and a partition $(L,R)$ of $V(G)$. For every vertex $v \in L$, Alg1 executes the following procedure. First, it uses BFS to find a shortest path $P$ from $s$ to $v$ in $G[L]$. If such a path $P$ exists and its length is $k$, then Alg1 proceeds as follows. It uses BFS to determine whether $t$ is reachable from $v$ in $G - (V(P) \setminus \{v\})$. If the answer is positive, Alg1 accepts. Eventually, if Alg1 did not accept for any $v \in L$, then it rejects. Clearly, the algorithm runs in polynomial time.

In one direction, it is clear that if the algorithm accepts, then $G$ has an $(s,t)$-path of length at least $k$. For the other direction, suppose that $\Lambda^k_G(s,t) \geq 2k$, and $G$ has an $(s,t)$-path $s = v_1 \to v_2 \to \ldots \to v_\ell = t$ such that $\ell = \Lambda^k_G(s,t)$, $v_1, v_2, \ldots, v_k \in L$ and $v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell \in R$. Then, there exists a path of length $k$ from $v_1$ to $v_k$ in $G[L]$. By Lemma 2.2 (with $\alpha = 2$), we also know that no shorter path exists. Moreover, Lemma 2.3 (with $\alpha = 2$) states that for any path $P$ of length $k$ from $s$ to $v_k$ in $G[L]$, $t$ is reachable from $v_k$ in $G - (V(P) \setminus \{v_k\})$. Thus, at the latest, Alg1 accepts in the iteration where it examines $v = v_k$. □

## 3. Unbalancedly annotated long $(s,t)$-paths

In this section we handle another special case of Long Directed $(s,t)$-Path where the prefix and suffix of a solution (if one exists) are "annotated". However, the current annotation may not be balanced as in Lemma 2.1 (specifically, the number of annotated vertices closer to $s$ is smaller), and the paths whose absence is assumed are not as long as those in Lemma 2.1. This special case lies at the heart of our algorithm, and it invokes the algorithm developed in the previous section as a black box. Specifically, we prove the following lemma.

**Lemma 3.1.** *Fix $1.5 \leq \alpha \leq 2$. There is a deterministic $\mathcal{O}^\star(4^{(2-\alpha)k} 2^{o(k)})$-time algorithm,* Alg2*, that given an instance $(G,s,t,k)$ of* Long Directed $(s,t)$-Path*, and a partition $(L,R)$ of $V(G)$, satisfies the following.*

- *If $\Lambda^k_G(s,t) \geq \lceil \alpha k \rceil$, and $G$ has an $(s,t)$-path $s = v_1 \to v_2 \to \cdots \to v_\ell = t$ such that $\ell = \Lambda^k_G(s,t)$, $v_1, v_2, \ldots, v_{\lceil(\alpha-1)k\rceil} \in L$ and $v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell \in R$, then* Alg2 *accepts.*
- *If $\Lambda^k_G(s,t) = -\infty$, then* Alg2 *rejects.*

**Proof.** Let $(G,s,t,k)$ be an instance of Long Directed $(s,t)$-Path, and let $(L,R)$ be a partition of $V(G)$. For every vertex $v \in L$, Alg2 executes the following procedure. First, it uses BFS to find a shortest path $P$ from $s$ to $v$ in $G[L]$. If such a path $P$ exists and its length is $\lceil(\alpha-1)k\rceil$, then Alg2 executes the following. It first uses Proposition 1.1 to compute an $(n, k - \lceil(\alpha-1)k\rceil, k - \lceil(\alpha-1)k\rceil)$-universal set $\mathcal{F}$. For every $F \in \mathcal{F}$ and vertex $u \notin V(P)$ that is an outgoing neighbor of $v$, Alg2 calls Alg1 with $(G' := G - V(P), u, t, k - \lceil(\alpha-1)k\rceil)$ and the partition $(F \setminus V(P), V(G) \setminus (F \cup V(P)))$ as input. Eventually, Alg2 accepts if and only if at least one call to Alg1 accepted.

By Proposition 1.1 and Lemma 2.1, Alg2 runs in time $\mathcal{O}^\star(\binom{2(k-\lceil(\alpha-1)k\rceil)}{k-\lceil(\alpha-1)k\rceil} 2^{o(k)})$, which implies the bound $\mathcal{O}^\star(4^{(2-\alpha)k} 2^{o(k)})$.

In one direction, suppose that Alg2 accepted. Then, by Lemma 2.1, there is a vertex $v \in V(G)$ and an out-neighbor $u$ of $v$ for which there exist vertex disjoint paths $P$ and $P'$ in $G$ such that $P$ is a path of length at least $\lceil(\alpha-1)k\rceil$ from $s$ to $v$, and $P'$ is a path of length $k - \lceil(\alpha-1)k\rceil$ from $u$ to $t$. Thus, $G$ has an $(s,t)$-path of length at least $k$.

For the other direction, suppose that $\Lambda_G^k(s,t) \geq \lceil \alpha k \rceil$, and $G$ has an $(s,t)$-path $s = v_1 \to v_2 \to \cdots \to v_\ell = t$ such that $\ell = \Lambda_G^k(s,t)$, $v_1, v_2, \ldots, v_{\lceil(\alpha-1)k\rceil} \in L$ and $v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell \in R$. Then, there exists a path of length $\lceil(\alpha-1)k\rceil$ from $v_1$ to $v_{\lceil(\alpha-1)k\rceil}$ in $G[L]$. By Lemma 2.2, we also know that no shorter path exists. Let us now examine iterations where $v = v_{\lceil(\alpha-1)k\rceil}$. Then, by the former arguments, Alg2 computes a path $P$ of length exactly $\lceil(\alpha-1)k\rceil$ from $s$ to $v_{\lceil(\alpha-1)k\rceil}$. By Lemma 2.3, there exists a path from $v_{\lceil(\alpha-1)k\rceil}$ to $v_\ell$ in $G - (V(P) \setminus \{v_{\lceil(\alpha-1)k\rceil}\})$ of length at least $k+1$. Let us denote a shortest such path by $P'$. Define $P^\star$ as $P'$ from which we remove $v_{\lceil(\alpha-1)k\rceil}$. Next, consider the iteration where $u$ is selected to be the first vertex on $P^\star$.

We claim that the length of $P^\star$ is $\Lambda_{G-V(P)}^{k-\lceil(\alpha-1)k\rceil}(u,t)$ (which means that $\Lambda_{G-V(P)}^{k-\lceil(\alpha-1)k\rceil}(u,t) \geq k$). To prove this claim, we need to show that $G - V(P)$ has no $(u,t)$-path of length at least $k - \lceil(\alpha-1)k\rceil$ that is shorter than $k$. Suppose, by way of contradiction, that such a path $\widehat{P}$ exists. Then, by traversing first $P$, then the edge from $v$ to $u$ and next the path $\widehat{P}$, we exhibit a path $Q$ such that $|V(Q)| = |V(P)| + |V(\widehat{P})| \geq \lceil(\alpha-1)k\rceil + (k - \lceil(\alpha-1)k\rceil) = k$ and $|V(Q)| = |V(P)| + |V(\widehat{P})| \leq \lceil(\alpha-1)k\rceil + k - 1 < \lceil \alpha k \rceil$. However, this is a contradiction to $\Lambda_G^k(s,t) \geq \lceil \alpha k \rceil$.

Let us observe that $k - \lceil(\alpha-1)k\rceil \leq 0.5k$ because $\alpha \geq 1.5$. Thus, by Definition 1.1, there exists $F \in \mathcal{F}$ such that each of the first $k - \lceil(\alpha-1)k\rceil$ vertices on $P^\star$ belong to $F$ and none of the last $k - \lceil(\alpha-1)k\rceil$ vertices on $P^\star$ belongs to $F$. Consider an iteration where such $F$ is examined. In order to complete the proof, it is sufficient to show that Alg1 accepts $(G - V(P), u, t, k - \lceil(\alpha-1)k\rceil)$ with the partition $(F \setminus V(P), V(G) \setminus (F \cup V(P)))$. To this end, by Lemma 2.1, it remains to show that $\Lambda_{G-V(P)}^{k-\lceil(\alpha-1)k\rceil}(u,t) \geq 2(k - \lceil(\alpha-1)k\rceil)$. However, we have shown that $\Lambda_{G-V(P)}^{k-\lceil(\alpha-1)k\rceil}(u,t) \geq k$, and $k \geq 2(k - \lceil(\alpha-1)k\rceil)$ because $\alpha \geq 1.5$. $\square$

## 4. (Normal) long $(s,t)$-paths

Having proved Lemma 3.1, we now proceed to prove a lemma that, together with Proposition 4.1, will lead us to our main theorem.

**Lemma 4.1.** *Fix $1.5 \leq \alpha \leq 2$. There is a deterministic $\mathcal{O}^\star((\frac{4^{2-\alpha}\alpha^\alpha}{(\alpha-1)^{\alpha-1}})^k \cdot 2^{o(k)})$-time algorithm,* LongAlg*, that given an instance $(G,s,t,k)$ of* Long Directed $(s,t)$-Path *where either $\Lambda_G^k(s,t) \geq \lceil \alpha k \rceil$ or $\Lambda_G^k(s,t) = -\infty$, accepts if and only if $G$ has an $(s,t)$-path of length at least $k$.*

**Proof.** Let $(G,s,t,k)$ be an instance of Long Directed $(s,t)$-Path where either $\Lambda_G^k(s,t) \geq \lceil \alpha k \rceil$ or $\Lambda_G^k(s,t) = -\infty$. LongAlg first uses Proposition 1.1 to compute an $(n, \lceil(\alpha-1)k\rceil, k)$-universal set $\mathcal{F}$. For every $F \in \mathcal{F}$, LongAlg calls Alg2 with $(G,s,t,k)$ and the partition $(F, V(G) \setminus F)$ as input. Eventually, LongAlg accepts if and only if at least one call to Alg2 accepted.

By Proposition 1.1 and Lemma 3.1, LongAlg runs in time $\mathcal{O}^\star(\binom{\lceil \alpha k \rceil}{k} 2^{o(k)} \cdot 4^{(2-\alpha)k})$, which implies (by Stirling's approximation) the bound $\mathcal{O}^\star((\frac{4^{2-\alpha}\alpha^\alpha}{(\alpha-1)^{\alpha-1}})^k \cdot 2^{o(k)})$. In one direction, Lemma 3.1 directly implies that if LongAlg accepts, then $G$ has an $(s,t)$-path of length at least $k$. For the other direction, suppose that $G$ has an $(s,t)$-path of length at least $k$. Then, $G$ has a path $s = v_1 \to v_2 \to \cdots \to v_\ell = t$ such that $\ell = \Lambda_G^k(s,t) \geq \lceil \alpha k \rceil$. By Definition 1.1, there exists $F \in \mathcal{F}$ such that $v_1, v_2, \ldots, v_{\lceil(\alpha-1)k\rceil} \in F$ and $v_{\ell-k+1}, v_{\ell-k+2}, \ldots, v_\ell \notin F$. By Lemma 3.1, when this set $F$ is examined, Alg2 accepts. Thus, LongAlg eventually accepts. $\square$

Our algorithm also relies on the following proposition.

**Proposition 4.1** *([9]). There is a deterministic algorithm,* ShortAlg*, that solves* Directed $k$-$(s,t)$-Path *in time $\mathcal{O}^\star(2.59606^k)$.*

Finally, we prove our main theorem.

**Theorem 1.** *There is a deterministic algorithm,* MainAlg*, that solves* Long Directed $(s,t)$-Path *in time $\mathcal{O}^\star(4.884^k)$.*

**Proof.** Fix $1.5 \leq \alpha \leq 2$ (to be determined). Given an instance $(G,s,t,k)$ of Long Directed $(s,t)$-Path, MainAlg executes the following computation. For all $\ell \in \{k, k+1, \ldots, \lfloor \alpha k \rfloor\}$, it calls ShortAlg with $(G,s,t,\ell)$ as input, and accepts if ShortAlg accepts. If it did not accept in any iteration, then it calls LongAlg with $(G,s,t,k)$ as input, and accepts if and only if LongAlg accepts.

The correctness of the algorithm directly follows from Lemma 4.1 and Proposition 4.1. Moreover, by Lemma 4.1 and Proposition 4.1, the running time of MainAlg is

$$\mathcal{O}^\star(\max\{2.59606^{\alpha k}, (\frac{4^{2-\alpha}\alpha^\alpha}{(\alpha-1)^{\alpha-1}})^k \cdot 2^{o(k)}\}).$$

By choosing $\alpha = 1.6624$, we derive that MainAlg runs in time $\mathcal{O}^\star(4.884^k)$. $\square$

As one can solve Long Directed Cycle by running, for every edge $e \in E(G)$, MainAlg with $s$ and $t$ being the target and source of $e$, respectively, we have the following corollary.

**Corollary 4.1.** *There is a deterministic algorithm that solves* Long Directed Cycle *in time $\mathcal{O}^\star(4.884^k)$.*

## References


[1] A. Björklund, T. Husfeldt, P. Kaski, M. Koivisto, Narrow sieves for parameterized paths and packings, J. Comput. Syst. Sci. 87 (2017) 119–139.

[2] J. Chen, J. Kneis, S. Lu, D. Molle, S. Richter, P. Rossmanith, S.H. Sze, F. Zhang, Randomized divide-and-conquer: improved path, matching, and packing algorithms, SIAM J. Comput. 38 (6) (2009) 2526–2547.

[3] F.V. Fomin, D. Lokshtanov, F. Panolan, S. Saurabh, Representative sets of product families, in: ESA, 2014, pp. 443–454.

[4] F.V. Fomin, D. Lokshtanov, F. Panolan, S. Saurabh, Efficient computation of representative families with applications in parameterized and exact algorithms, J. ACM 63 (4) (2016) 1–60.

[5] F.V. Fomin, D. Lokshtanov, S. Saurabh, Efficient computation of representative sets with applications in parameterized and exact algorithms, in: SODA, 2014, pp. 142–151.

[6] H.N. Gabow, S. Nie, Finding a long directed cycle, ACM Trans. Algorithms 4 (1) (2008).

[7] H. Shachnai, M. Zehavi, Representative families: a unified tradeoff-based approach, J. Comput. Syst. Sci. 82 (3) (2016) 488–502.

[8] R. Williams, Finding paths of length $k$ in $O^*(2^k)$ time, Inf. Process. Lett. 109 (6) (2009) 315–318.

[9] M. Zehavi, Mixing color coding-related techniques, in: ESA, 2015, pp. 1037–1049.

[10] M. Zehavi, A randomized algorithm for long directed cycle, Inf. Process. Lett. 116 (6) (2016) 419–422.