TheoretıCS

# The NFA Acceptance Hypothesis: Non-Combinatorial and Dynamic Lower Bounds

**Karl Bringmann** [a] ✉

**Allan Grønlund** [b] ✉

**Marvin Künnemann** [c] ✉

**Kasper Green Larsen** [d] ✉

**a** Saarland University and Max-Planck-Institute for Informatics, Saarland Informatics Campus

**b** Aarhus University and Kvantify

**c** Karlsruhe Institute of Technology

**d** Aarhus University

**ABSTRACT.** We pose the fine-grained hardness hypothesis that the textbook algorithm for the NFA Acceptance problem is optimal up to subpolynomial factors, even for dense NFAs and fixed alphabets.

We show that this barrier appears in many variations throughout the algorithmic literature by introducing a framework of Colored Walk problems. These yield fine-grained equivalent formulations of the NFA Acceptance problem as problems concerning detection of an $s$-$t$-walk with a prescribed color sequence in a given edge- or node-colored graph. For NFA Acceptance on *sparse* NFAs (or equivalently, Colored Walk in sparse graphs), a tight lower bound under the Strong Exponential Time Hypothesis has been rediscovered several times in recent years. We show that our hardness hypothesis, which concerns *dense* NFAs, has several interesting implications:

— It gives a tight lower bound for Context-Free Language Reachability. This proves conditional optimality for the class of 2NPDA-complete problems, explaining the *cubic bottleneck* of interprocedural program analysis.

— It gives a tight $(nm^{1/3} + m)^{1-o(1)}$ lower bound for the Word Break problem on strings of length $n$ and dictionaries of total size $m$.

— It implies the popular OMv hypothesis. Since the NFA acceptance problem is a static (i.e., non-dynamic) problem, this provides a static reason for the hardness of many dynamic problems.

Thus, a proof of the NFA Acceptance hypothesis would resolve several interesting barriers. Conversely, a refutation of the NFA Acceptance hypothesis may lead the way to attacking the current barriers observed for Context-Free Language Reachability, the Word Break problem and the growing list of dynamic problems proven hard under the OMv hypothesis.

# 1.     Introduction

Consider a classic problem that lies at the heart of introductory undergraduate courses on the theory of computation: Given a nondeterministic finite automaton (NFA) $M$ and a string $x$ over some alphabet $\Sigma$, determine whether $x$ is accepted by $M$. The textbook algorithm for solving this problem uses dynamic programming and solves the problem in time $O(|M| \cdot |x|)$. Optimality of this algorithm is known for *sparse* NFAs, assuming the Strong Exponential Time Hypothesis and up to subpolynomial factors, as we will discuss in detail in Section 1.2. We put forth the following fine-grained hardness hypothesis, in which we conjecture optimality of this decades-old algorithm also for *dense* NFAs.

**HYPOTHESIS 1.1** (NFA Acceptance hypothesis, informal version). The textbook $O(|M| \cdot |x|)$ time algorithm for NFA Acceptance over any fixed alphabet $\Sigma$ is optimal up to subpolynomial factors, even if $M$ is a *dense* NFA.

In this paper, we shed light on the many guises of the NFA Acceptance problem and highlight the ramifications of the above hardness hypothesis on fine-grained complexity theory in P.

## 1.1   The Many Guises of the NFA Acceptance Problem

To express equivalent formulations of the NFA Acceptance problem, consider the following problem: Given a directed, simple graph $G = (V, E)$ with edge colors $c : E \to \Sigma$, distinguished nodes $s, t \in V$ and a color sequence $c_1, \ldots, c_\ell \in \Sigma$, determine whether there is a walk of length $\ell$ from $s$ to $t$ such that the color sequence of the traversed edges is equal to $c_1, \ldots, c_\ell$. In a graph with $n$ nodes and $m \geq n$ edges, this problem is easily solved in time $O(m\ell)$, by maintaining a set $S \subseteq V$ of states reachable from $s$ via a walk with color sequence $c_1, \ldots, c_i$, over all $1 \leq i \leq \ell$. There are natural variants of this problem, by considering *node* colors rather than *edge* colors, *undirected* rather than *directed* graphs, as well as various restrictions on the *size of the*

*alphabet*: We call these problems DIRECTED/UNDIRECTED Σ-EDGE-COLORED/Σ-NODE-COLORED WALK, see Section 2 for a formal definition. All of these problems can be solved in time $O(m\ell)$. These Colored Walk problems arise in different communities, typically with different generalizations:

— (Formal language theory:) NFA Acceptance is a fundamental problem in formal language theory, as it is the membership problem for regular languages. DIRECTED Σ-EDGE-COLORED WALK is precisely the special case in which the NFA has no loops (transitions from some state $q$ to itself) or multiple transitions between any two states (with different labels from the terminal set).[1] In Section 6, we will show that this special case is fine-grained equivalent to the general case of the NFA Acceptance problem.

— (Hidden Markov models:) A Hidden Markov Model (HMM) is a Markov chain in which every state has a distribution over possible observations Σ it emits. In the VITERBI PATH problem, the task is to determine, given a sequence $c_1, \ldots, c_\ell$ of observations over Σ and an HMM $M$, the most likely walk through $M$ to emit $c_1, \ldots, c_\ell$. An extension of the NFA Acceptance algorithm, well-known as Viterbi's algorithm [68], solves this problem in time $O(m\ell)$, where $m$ denotes the size of $M$. Exploiting the weighted nature of this problem, Viterbi's algorithm has been proven optimal up to subpolynomial factors under the weighted $k$-Clique hypothesis [11]. Σ-NODE-COLORED WALK is equivalent to the following interesting *unweighted* special case of the problem: in an HMM for which every node $v$ emits a single observation $\sigma_v$ with probability 1, determine whether there exists *any* walk with positive probability for emitting the observations $c_1, \ldots, c_\ell$.[2]

— (Combinatorial Pattern matching, Bioinformatics:) In some applications in combinatorial pattern matching, e.g., in bioinformatics, it is natural to represent a *set* of strings using a node-labeled graph $G$: Here, every node $u$ is equipped with a string $L_u \in \Sigma^*$ and the represented set of strings is simply the list of all concatenations $L_{u_1} \ldots L_{u_k}$ over all walks $u_1 \ldots u_k$ in $G$. A natural question is to perform pattern matching on these graphs, i.e., find an occurrence, or more generally, all occurrences of a given pattern $P$ in some string $T$ occurring in $G$. A line of work, see, e.g., [50, 7, 53, 38] derives algorithms for exact and approximate matches of a string, see also [30] for references. Σ-Node-Colored Walk is equivalent to detecting an exact occurrence in a graph in which every node is labeled by a single character in Σ.[3]

— (Database Theory:) An important topic in the area of graph databases is *regular path queries*, see, e.g. [52, 12, 18] and references therein. Here, a database is given as a graph $G = (V, E)$ with edge labels over Σ and the aim is to support queries, which on input a regular expression $q$, return information about walks in the graph whose labels match the regular

---

1　　Formulating the NFA Acceptance problem as a colored walk problem is not uncommon, see, e.g., [8]. Here we use the Colored Walk problem as an umbrella to express various related problems in a common language.

2　　The equivalence of detecting an $s$-$t$-walk with a given color sequence and detecting *any* walk with a given color sequence is discussed in Section 6.

3　　See previous footnote.

expression $q$ (variants of these queries include emptiness, counting, or enumeration of such walks or their endpoints). The emptiness query can be solved in time $O((|V|+|E|)|q|)$ using a straightforward approach of constructing a product graph of $G$ and an NFA representing $q$; conditional optimality of this approach has been investigated in [18]. The special case in which $q$ describes a single string of length $\ell$ is precisely the Σ-EDGE-COLORED WALK problem.[4]

Interestingly, it turns out that all the special cases discussed above are equivalent in a fine-grained sense. In particular, the NFA Acceptance problem, as well as all $n^{o(1)}$-Edge-Colored Walk and $n$-Node-Colored Walk problems are equivalent to the following seemingly simple problem:

---

DIRECTED 2-EDGE-COLORED WALK

*Input:*　　　Directed graph $G = (V, E)$, $s, t \in V$, $c : E \to \{1, 2\}$, $\ell \in \mathbb{N}$, colors $c_1, \ldots, c_\ell \in \{1, 2\}$

*Question:* Exist $s = v_0, v_1, \ldots, v_\ell = t$ with $(v_{i-1}, v_i) \in E$ and $c(v_{i-1}, v_i) = c_i$ for all $1 \leq i \leq \ell$?

---

For a formal statement of this equivalence and its proof, see Sections 2 and 6.

Thus, the same barrier has been observed across different communities, with extensions of the $O(m\ell)$-time algorithm in various distinct directions. The only generalization with a tight conditional lower bound for all graph densities is known in the weighted setting of Viterbi's algorithm [11].

A refutation of the NFA Acceptance hypothesis would thus immediately be interesting for a host of communities – as a case in point, several works derive essentially the same conditional lower bounds for sparse graphs and for combinatorial algorithms, see Section 1.2. Furthermore, refuting the hypothesis is a prerequisite to obtaining faster algorithms also for the discussed generalizations, e.g., for *approximating* the VITERBI PATH problem.

## 1.2　Support for the Hypothesis

The current support for the hypothesis is threefold: (I) There are tight conditional lower bounds for *sparse* NFAs based on the Orthogonal Vectors Hypothesis (OVH) and thus the Strong Exponential Time Hypothesis (SETH), (II) there is a tight combinatorial lower bound for all densities of NFAs based on the combinatorial $k$-Clique hypothesis, and (III) simply the lack of a faster algorithms despite the centrality and age of this problem. We discuss these reasons in detail:

**Support I: Hardness for Sparse NFAs**　It is known that an $O((n\ell)^{1-\epsilon})$-time algorithm for acceptance of a length-$\ell$ string by an $n$-state NFA would refute OVH and SETH. This can be viewed as a tight conditional lower bound for *sparse* NFAs with $m = O(n)$ transitions, but it

---

4　　See previous footnotes.

fails to provide a tight lower bound for NFAs with $m = \Theta(n^\alpha)$ transitions with $1 < \alpha \leq 2$. Several versions of this result have been proven and published in recent years. A subset of the authors have learned the result phrased for NFAs from Russell Impagliazzo in 2015. Since regular expressions of size $n$ can be easily converted to an NFA with $O(n)$ states and transitions, the quadratic conditional lower bounds for regular expression membership/matching in [10, 15] translate to NFA Acceptance lower bounds. Subsequently, the lower bound for sparse NFAs has been given in the language of string matching in labeled graphs in [30], in the language of NFA Acceptance in [57], and in the language of regular path queries in graph databases in [18].[5]

Interestingly, the same lower bound already holds under a weaker hypothesis, specifically Formula-SETH [3, 59, 32].

**Support II: Combinatorial Hardness**  Beyond sparse NFAs, the literature provides, perhaps a bit implicitly, a (combinatorial) reduction from $k$-Clique to NFA Acceptance with a dense NFA with $n$ states with the following implications: (1) An $O((\ell n^2)^{1-\epsilon})$-time *combinatorial*[6] NFA Acceptance algorithm for any $\ell = n^\beta, \beta > 0$ would give a *combinatorial* $O(n^{(1-\epsilon')k})$-time $k$-Clique algorithm for all sufficiently large $k$, and (2) an $O((\ell n^2)^{\omega/3-\epsilon})$-time NFA Acceptance algorithm would give an $O(n^{(\omega/3)k-\epsilon'})$-time $k$-Clique algorithm for sufficiently large $k$, breaking the state of the art for current $k$-Clique algorithms, see [55, 2].

These lower bounds can be obtained either by adapting the reduction from weighted $k$-Clique in [11] to the unweighted case, or as a special case of [1, Theorem I.5][7], or as an appropriate generalization of the reduction from Triangle Detection to NFA Acceptance with $\ell \approx n$ in [57], or as an appropriate generalization of the reduction from Triangle Detection to regular path queries with queries of length $\ell \approx n$ in [18]. For a self-contained reduction see Section 7.

These reductions suggest that to break the NFA Acceptance hypothesis, a non-trivial application of fast matrix multiplication techniques is needed. Put differently, our hypothesis boils down to postulating that *fast matrix multiplication is not applicable for the NFA Acceptance problem*.

**Support III: Long-standing State of the Art**  Finally, the NFA Acceptance hypothesis is plausible simply due to the lack of any improved algorithms. To the best of our knowledge, the best algorithm for NFA Acceptance for dense NFAs over a constant-sized alphabet $\Sigma$ runs in time $n^2\ell/2^{\Omega(\sqrt{\log n})}$ (see Section 5).

Let us contrast the state of the art for NFA Acceptance, i.e., the membership problem for regular languages, to the membership problem for context-free languages, specifically

---

5     By the the fine-grained equivalence of the Colored Walk problems of Section 2, these results turn out to be essentially equivalent in their main statement. However, some aspects of the reductions can differ.

6     Here, a *combinatorial* algorithm refers to an algorithm that avoids the algebraic techniques underlying fast matrix multiplication algorithms, see Section 1.4 for a discussion.

7     In the theorem, simply choose $\alpha_N = \alpha_n = \beta$.

context-free grammar parsing. The fine-grained complexity of this problem has parallels to NFA Acceptance: The best combinatorial algorithms decide membership of a length-$n$ string in a fixed grammar $\Gamma$ in mildly subcubic time $O(n^3/\text{polylog } n)$, see [2] for an overview. Any truly subcubic combinatorial algorithm would refute the combinatorial $k$-Clique algorithm [2]. However, by a highly non-trivial application of fast matrix multiplication, Valiant [61] gave an algorithm for context-free grammar parsing in time $O(n^\omega)$ for constant-sized grammars, but this is known already since 1975! It appears unlikely that a similar algorithm using fast matrix multiplication for NFA Acceptance has been overlooked for decades.

In Section 8, we also discuss why natural algorithmic approaches towards refuting the NFA Acceptance hypothesis appear to fail.

It is apparent that the NFA Acceptance hypothesis poses a significant barrier. Finally, we remark that [57] recently posed our NFA Acceptance hypothesis as an open problem.[8]

### 1.3    Evidence Against the Hypothesis

The only evidence against the NFA Acceptance hypothesis that we are aware of is the existence of (co-)nondeterministic verifiers with running time $O((n^2\ell)^{1-\epsilon})$. This is even known for the weighted generalization to the Viterbi Path problem [10], but can be proven directly (see Section 7) or follows as a consequence of our reduction in Section 3 combined with [22]. This rules out tight *deterministic* reductions from SAT and OV to NFA Acceptance with dense NFAs, assuming a nondeterministic variant of SETH [17]. However, there are several hypotheses with a similar status, in particular, the APSP hypothesis, 3SUM hypothesis and the Hitting Set hypothesis.

### 1.4    Applications I: Non-combinatorial Lower Bounds

As our first major consequence of the NFA Acceptance hypothesis, we give novel tight conditional lower bounds against general algorithms where previously only combinatorial lower bounds were known.

**Combinatorial Lower Bounds**  Some of the oldest conditional lower bounds in the polynomial-time regime are based on Boolean matrix multiplication (BMM), see, e.g. [47] (for a survey of fine-grained complexity theory in P, we refer to [64]). The best known algorithm for BMM uses fast matrix multiplication over the integers, yielding an $n^{\omega+o(1)}$-time algorithm, where $\omega$ is the matrix multiplication exponent, with a current state-of-the-art bound of $\omega < 2.371552$ due to works of [60, 25, 63, 46, 6, 29, 67], among others.

---

8    Potechin and Shallit note, following the SETH-based lower bound for sparse NFAs: "However, this does not rule out an improvement when the NFA is dense, and we leave it as an open problem to either find a significant improvement to this algorithm, or show why such an improvement is unlikely."

Unfortunately, theoretically fast matrix multiplication algorithms generally turn out to be impractical, as the constants hidden in the $O$-notation are usually too large for the problem sizes observed in applications. Therefore, it has become increasingly popular to use the notion of *combinatorial* algorithms to exclude the use of algebraic techniques underlying current fast matrix multiplications, in an attempt to study algorithms whose asymptotic complexity translates into practical running time. In this direction, it has been hypothesized that BMM does not admit combinatorial $O(n^{3-\epsilon})$-time algorithms with $\epsilon > 0$. This hypothesis is in fact equivalent to the nonexistence of truly subcubic combinatorial algorithms for triangle detection, and gives tight combinatorial lower bounds for problems such as the Sliding Window Hamming Distance (see [31]), CFL Reachability [20] and many more. A generalization of this hypothesis is that for no $k \geq 3$ and $\epsilon > 0$, $k$-Clique admits a combinatorial $O(n^{k-\epsilon})$-time algorithm, which implies a combinatorial version of our NFA Acceptance hypothesis, see Section 7.

However, there are several downsides to the notion of *combinatorial* lower bounds: First and foremost, the notion of *combinatorial algorithm* is not formally defined, which is hardly acceptable (see e.g. the discussion in [35]). Second, in theoretical computer science we are usually interested in the optimal asymptotic worst-case complexity, so why would we exclude algorithmic techniques leading to an improved complexity? Third, Strassen's original algorithm is sometimes found to be practical [37]. Hence, there are several reasons to avoid "combinatorial lower bounds", as they leave open the very real possibility that faster algorithms exploiting fast matrix multiplication techniques exist.

How can we distinguish between problems for which faster algorithms via fast matrix multiplications exist, and those for which such an improvement is unlikely? Generally speaking, the current state of the art for problems with tight combinatorial lower bounds can be categorized as follows:

1. (easy:) Problems whose asymptotic complexity is upper bounded by the complexity of fast matrix multiplication, e.g., context-free grammar recognition for constant-sized grammars [61], maximum node-weighted triangle [26], and others.

2. (intermediate:) Problems for which improvements via fast matrix multiplication are possible, but they do not necessarily lead to quasilinear time in the input if $\omega = 2$: e.g., Sparse Triangle Listing or intermediate $O(n^{\frac{3+\omega}{2}})$-time problems such as $(\min, \max)$ product, Dominance product, Equality product, and All-Edges Monochromatic Triangle. Only for a few of these problems, conditional lower bounds give evidence why they do not appear to belong to the first category, e.g. [56, 48, 66, 19].

3. (difficult:) Problems for which even using fast matrix multiplication, no improved algorithms are known, e.g., APSP, Sliding Window Hamming Distance, Word Break, Context-free Language Reachability, Klee's Measure Problem. To establish that a problem lies in this category, currently only two hardness assumptions appear applicable: If the problem can express *weights*, a reduction from APSP or more generally the weighted $k$-Clique

hypothesis is plausible, e.g. [65, 9, 11]. A second possibility is a reduction based on the 3-uniform Hyperclique hypothesis [49], which was used, e.g., for Klee's measure problem in 3D [43].

The NFA Acceptance hypothesis yields a new route to establish that a problem belongs to the third category. We describe two such applications: (1) Context-free language reachability and, more generally, all 2NPDA-complete problems (which suffer from the *cubic bottleneck* in interprocedural program analysis), and (2) the Word Break problem.

**CFL Reachability and 2NPDA-hard problems**　In Context-free Language (CFL) Reachability for a fixed context-free grammar $\Gamma$, we are given a $\Sigma$-edge-colored directed graph $G$ and distinguished nodes $s, t$, and the task is to determine whether there is a walk from $s$ to $t$ such that the sequence of traversed edge colors spells a word in the language given by $\Gamma$. The CFL Reachability problem has many applications in program analysis, verification and database theory (see, e.g., the references in [22, 42]). A classic algorithm solves CFL Reachability in time $O(n^3)$ [70], which has been slightly improved to time $O(n^3/\log n)$ [58, 21]. In fact, these algorithms extend to the generalization of computing all pairs $(s, t)$ for which such a walk exists.

The inability to obtain truly subcubic time algorithms for CFL Reachability and related problems has led researchers to investigate its relationship to the recognition problem for two-way nondeterministic pushdown automata (2NPDA), which admits a classic cubic time algorithm due to [5]. Several problems, including CFL Reachability, pushdown automata emptiness, and related problems in data flow analysis, have been proven 2NPDA-complete in the sense that they are subcubic equivalent to 2NPDA recognition [54, 51, 34], see also [22].

Since then, there have been attempts to obtain conditional lower bounds for 2NPDA-complete problems: For *combinatorial* algorithms a conditional lower bound ruling out time $O(n^{3-\varepsilon})$ for any $\varepsilon > 0$ was shown in [20]. However, it was proven that obtaining a tight lower bound based on SETH/OVH using deterministic reductions would violate NSETH [22]. Recently, an interesting attempt was made to show that the All-Pairs version of CFL Reachability is strictly harder than matrix multiplication (i.e., does not belong to the first category in the above list) via a reduction from All-Edges Monochromatic Triangle [42].[9]

In Section 3, we show that proving the NFA Acceptance hypothesis settles the cubic bottleneck of 2NPDA-complete problems by showing that the NFA Acceptance hypothesis implies that CFL Reachability, and thus, all 2NPDA-hard problems, have no truly subcubic algorithms. In particular, this gives conditional optimality of the algorithms due to Yannakakis [70] and Aho et al. [5].

---

9　Unfortunately, the reduction from Monochromatic Triangle to All-Pairs CFL Reachability given in [42] appears to be flawed. In this reduction, a graph is created where every edge in the original graph is replaced by a line graph, which in general leads to a quadratic blow-up in the number of nodes. In personal communication, the authors confirm this issue.

**Word Break**  In the Word Break problem we are given a string $S$ and a dictionary $D$ (i.e., $D$ is a set of strings) and we ask whether $S$ can be split into dictionary words, i.e., whether we can partition $S$ into substrings such that each substring is in the set $D$. Denoting the length of $S$ by $n$ and the total length of all dictionary words by $m$, this problem has a relatively simple randomized algorithm running in time $\widetilde{O}(nm^{1/2} + m)$.[10] This running time was first improved to $\widetilde{O}(nm^{1/2-1/18} + m)$ [10] and then to the current state of the art $O(n(m \log m)^{1/3} + m)$ [15]. In [15] it was also shown that Word Break has no combinatorial algorithm running in time $O(nm^{1/3-\varepsilon}+m)$ for any $\varepsilon > 0$, assuming a hypothesis on combinatorial $k$-Clique. Under standard, non-combinatorial hypotheses, no superlinear running time lower bound is known.

In Section 4, we settle this issue by proving a tight conditional lower bound for Word Break under the NFA Acceptance hypothesis.

## 1.5   Application II: A Static Reason for Dynamic Hardness

In 2015, Henzinger, Forster, Nanongkai and Saranurak [35] formulated the OMv hypothesis which unifies and strengthens many conditional lower bounds for data structure problems and dynamic problems and has since been used in various areas such as graph algorithms [35, 27, 4, 36], string algorithms [24, 41, 23], computational geometry [28, 45], linear algebra [62, 39], formal languages [33], and database theory [13, 18, 40]. In fact, almost all conditional lower bounds for dynamic problems known under BMM or other hypotheses such as 3SUM or APSP can also be shown under the OMv hypothesis.[11] Thus, there is one reason that allows to rule out faster algorithms for a wealth of problems, without restricting the class of algorithms that may be used, and without any undefined notions.

Interestingly, it turns out that *the NFA Acceptance hypothesis implies the OMv hypothesis*. The reduction is very direct and has nice implications: (1) It establishes NFA Acceptance as a static reason for the hardness of a wealth of dynamic problems. (2) It gives additional reason to believe in the OMv hypothesis, since NFA Acceptance is an additional problem that has been studied for decades and lacks polynomially improved algorithms. (3) We obtain an improved running time for NFA Acceptance by a factor $2^{\Theta(\sqrt{\log n})}$, by the improved algorithm for OMv [44]. (4) Since NFA Acceptance is a static problem, we can try to obtain similar lower bounds as from OMv now for static versions of dynamic problems. We give the reduction in Section 5.

## 2.    Equivalent formulations: Colored Walk Framework

An NFA $M$ is a tuple $(Q, \Sigma, \delta, q_0, F)$ with $\delta \subseteq Q \times \Sigma \times Q$, $q_0 \in Q$ and $F \subseteq Q$. We say that $M$ accepts a string $x \in \Sigma^*$ if and only if there exists some sequence $q_1, \ldots, q_{|x|} \in Q$ with $(q_i, x_{i+1}, q_{i+1}) \in \delta$

---

10    Throughout the paper we write $\widetilde{O}(T) := \bigcup_{c \geq 0} O(T \log^c T)$, where we denote $\log^c T = (\log(T))^c$.

11    Lower bounds from SETH seem to be an exception from this rule of thumb.

for all $0 \leq i < |x|$ and $q_{|x|} \in F$. Note that we do not allow $\epsilon$-transitions.[12] The NFA Acceptance problem asks to determine, given an NFA $M$ and string $x$ over $\Sigma$, whether $M$ accepts $x$. We usually use $n = |Q|$ to denote the number of states, $m = |\delta|$ to denote the number of transitions and $\ell = |x|$ to denote the length of the string $x$.

We are ready to state our hypothesis formally:

**HYPOTHESIS 2.1** (NFA Acceptance hypothesis, formal version). Let $1 \leq \alpha \leq 2$ and $\beta, \epsilon > 0$ be arbitrary. There is no (randomized) algorithm that solves the NFA Acceptance problem for NFAs with $n$ states and $m = \Theta(n^\alpha)$ transitions, alphabet $\Sigma$ of size $n^{o(1)}$ and strings of length $\ell = \Theta(n^\beta)$ in time $O((m\ell)^{1-\epsilon}) = O(n^{(\alpha+\beta)(1-\epsilon)})$.

It is equivalent to state the hypothesis only for dense NFAs, i.e., $\alpha = 2$, as we show next.

**LEMMA 2.2.** *The NFA Acceptance hypothesis is equivalent to the following statement: Let $\beta, \epsilon > 0$ be arbitrary. There is no (randomized) algorithm that solves the NFA Acceptance problem for NFAs with n states and $m = \Theta(n^2)$ transitions, alphabet $\Sigma$ of size $n^{o(1)}$ and strings of length $\ell = \Theta(n^\beta)$ in time $O((n^2\ell)^{1-\epsilon}) = O(n^{(2+\beta)(1-\epsilon)})$.*

**PROOF.** Assume that there exists $1 \leq \alpha' < 2$ and $\beta' > 0$ such that we can solve the problem with $m = \Theta(n^{\alpha'})$ transitions and sequence length $\ell = \Theta(n^{\beta'})$ in time $O(n^{(\alpha'+\beta')(1-\epsilon)})$. We define $\alpha = 2$, $\beta = (2/\alpha')\beta'$ and consider any instance with $m = \Theta(n^\alpha) = \Theta(n^2)$ and $\ell = \Theta(n^\beta)$. Create an equivalent instance by simply introducing $n^{2/\alpha'} \geq n$ additional isolated states. This instance has $n' = \Theta(n^{2/\alpha'})$ states, the same number of transitions $m' = m = \Theta(n^2) = \Theta((n')^{\alpha'})$ and the same sequence length $\ell' = \ell = \Theta(n^\beta) = \Theta((n')^{\beta'})$. By our assumption, we can solve this instance in time $O((n')^{(\alpha'+\beta')(1-\epsilon)}) = O(n^{(\alpha+\beta)(1-\epsilon)})$. This contradicts the assumption for $\alpha = 2$ and an appropriate $\beta$, as desired.     ∎

Furthermore, our conditional lower bounds for CFL Reachability and OMv follow already from the setting $\alpha = 2$ and $\beta = 1$, which can be viewed as the core setting of the NFA Acceptance hypothesis: Determining whether a given $n$-state NFA $M$ over an $n^{o(1)}$-sized alphabet $\Sigma$ accepts a given length-$n$ string $x$ requires time $n^{3-o(1)}$ in the worst case.

In the remainder of the section, we discuss several equivalent formulations of the NFA Acceptance hypothesis, using a framework of graph problems that we refer to as Colored Walk problems.

## 2.1   Colored Walk Framework

We study the following variants of the Colored Walk problem.

---

12     An $\epsilon$-transition is a transition labeled with the empty word $\epsilon$. Such a transition can be taken by the NFA without reading a character of $x$. While for dense NFAs, allowing $\epsilon$-transitions yields a linear-time equivalent version of the NFA acceptance problem, we leave it as an open problem whether $\epsilon$-transitions make the problem more difficult for sparser NFAs.

---

DIRECTED/UNDIRECTED NODE-$C(n)$-COLORED WALK

*Input:*     directed/undirected graph $G = (V, E)$ with $n = |V|$ and $m = |E|$, vertices $s, t \in V$,

coloring $c \colon V \to \{1, \ldots, C(n)\}$, integer $\ell$, color sequence $c_1, \ldots, c_\ell$

*Question:* Exist $s = v_0, v_1, \ldots, v_\ell = t$ with $(v_{i-1}, v_i) \in E$ and $c(v_i) = c_i$ for all $1 \le i \le \ell$?

---

DIRECTED/UNDIRECTED EDGE-$C(n)$-COLORED WALK

*Input:*     directed/undirected graph $G = (V, E)$ with $n = |V|$ and $m = |E|$, vertices $s, t \in V$,

coloring $c \colon E \to \{1, \ldots, C(n)\}$, integer $\ell$, color sequence $c_1, \ldots, c_\ell$

*Question:* Exist $s = v_0, v_1, \ldots, v_\ell = t$ with $(v_{i-1}, v_i) \in E$ and $c(v_{i-1}, v_i) = c_i$ for all $i$?

---

Note that all of these problems can be solved in time $O(m\ell)$. We show that most of these problem variants are equivalent, in the following sense. Note that statement A1 below is exactly the NFA Acceptance hypothesis.

**LEMMA 2.3 (Colored Walk Framework).** *Let $\alpha \in [1, 2], \beta > 0$. All of the following statements A1, A2, A3, A4, A5, B1, B2, B3, and B4 are equivalent:*

A  *Restricted to instances with $m = \Theta(n^\alpha)$ and $\ell = \Theta(n^\beta)$, there is no $O(n^{\alpha+\beta-\varepsilon})$-time algorithm for any $\varepsilon > 0$ for the problem...*

    1  *... NFA Acceptance with alphabet size $n^{o(1)}$.*

    2  *... Directed Node-2-Colored Walk.*

    3  *... Directed Node-n-Colored Walk.*

    4  *... Directed Edge-2-Colored Walk.*

    5  *... Directed Edge-$n^{o(1)}$-Colored Walk.*

B  *Restricted to instances with $m = \Theta(n^\alpha)$ and $\ell = O(n^\beta)$, there is no $O(n^{\alpha+\beta-\varepsilon})$-time algorithm for any $\varepsilon > 0$ for the problem...*

    1  *... Undirected Node-2-Colored Walk.*

    2  *... Undirected Node-n-Colored Walk.*

    3  *... Undirected Edge-2-Colored Walk.*

    4  *... Undirected Edge-$n^{o(1)}$-Colored Walk.*

The proof is deferred to Section 6.

From now on, we use the term *Colored Walk* for any of the above problems. In particular, in the reductions based on the NFA Acceptance hypothesis, we can always use the variant that is easiest to work with. In particular, all of our reductions in the following sections will start from Directed Edge-2-Colored Walk (i.e., we will use statement A4). By the above lemma, we thus obtain conditional lower bounds under the NFA Acceptance hypothesis.

We leave it as an open problem whether Directed/Undirected Edge-$C(n)$-Colored Walk for $C(n) \gg n^{o(1)}$ is also equivalent to the above problems.

In Section 6, we also show equivalence of Colored Walk to a version without source and target nodes $s, t$, specifically, the version in which we determine existence of *any* walk in the graph with a given color sequence $c_1, \ldots, c_\ell$. Such versions occur e.g., for string matching in labeled graphs or regular path queries.

## 3.    Hardness of CFL Reachability

In this section, we give a tight hardness result for CFL reachability under the NFA acceptance hypothesis.

A *context-free grammar* $\Gamma$ is a tuple $(N, \Sigma, P, S)$, where $N$ is a set of nonterminals, $\Sigma$ is a set of terminals disjoint from $N$, $S \in N$ is the start symbol and $P$ is a set of production rules of the form $X \to \alpha$ with $X \in N$ and $\alpha \in (N \cup \Sigma)^*$. For a string $x \in (N \cup \Sigma)^*$, replacing an occurrence of a nonterminal $X \in N$ by $\alpha$ (where $X \to \alpha$ is in $P$) is called an application of the production rule $X \to \alpha$. We say that a string $x \in \Sigma^*$ is generated by $\Gamma$ if $x$ can be obtained from $S$ by (repeated) applications of production rules. We let $L(\Gamma)$ denote the set of strings generated by $\Gamma$.

Recall that in the CFL Reachability problem we are given a context-free grammar $\Gamma$ describing a language $L(\Gamma)$ over a terminal set $\Sigma$ as well as a directed graph $G = (V, E)$ with designated vertices $s, t \in V$ where every edge $e \in E$ is labeled by a terminal $\sigma(e) \in \Sigma$. We call any sequence of terminals $w \in \Sigma^*$ a *word*, and we say that a walk $v_0, v_1, \ldots, v_t$ in $G$ *spells the word* $\sigma(v_0, v_1)\sigma(v_1, v_2) \ldots \sigma(v_{\ell-1}, v_\ell)$, i.e., we concatenate all edge labels along the walk. The task is to decide whether there is a walk from $s$ to $t$ in $G$ spelling a word in $L(\Gamma)$, i.e., whether there is a walk $s = v_0, v_1 \ldots, v_\ell = t$ such that $\sigma(v_0, v_1)\sigma(v_1, v_2) \ldots \sigma(v_{\ell-1}, v_\ell) \in L(\Gamma)$. We write $n = |V|, m = |E|$ and we assume that $\Gamma$ is fixed, in particular it has constant size and any running time dependence on the size of $\Gamma$ can be ignored.

The CFL Reachability problem has a classic algorithm running in time $O(n^3)$ [70], which has been slightly improved to time $O(n^3/\log n)$ [21]. For *combinatorial* algorithms a conditional lower bound ruling out time $O(n^{3-\varepsilon})$ for any $\varepsilon > 0$ was shown in [20].

We prove a tight lower bound for CFL Reachability under the NFA Acceptance hypothesis:

**THEOREM 3.1.** *There is a fixed grammar $\Gamma$ such that the CFL Reachability problem on $\Gamma$ has no $O(n^{3-\varepsilon})$-time algorithm for any $\varepsilon > 0$ assuming the NFA Acceptance hypothesis.*

**PROOF.** We reduce from Directed Edge-2-Colored Walk. To this end, we are given a directed graph $G = (V, E)$ with colors $c \colon E \to \{1, 2\}$, designated vertices $s, t \in V$, and a color sequence $c_1, \ldots, c_\ell \in \{1, 2\}$. Let $n = |V|$ and $m = |E|$. We assume the graph to be dense (i.e., $m = \Theta(n^2)$) and we assume $\ell = \Theta(n)$. By the NFA Acceptance hypothesis in the setting $\alpha = 2, \beta = 1$, there is no algorithm solving all such instances of Directed Edge-2-Colored Walk in time $O(n^{3-\varepsilon})$ for any $\varepsilon > 0$.

We construct an instance of CFL Reachability as follows. The context-free grammar $\Gamma$ is the language of well-formed expressions on two types of parenthesis (also known as Dyck-2) given by the nonterminal $S$, the terminals $(_1, )_1, (_2, )_2$, and the production rules $S \rightarrow SS, S \rightarrow (_1S)_1, S \rightarrow (_1)_1, S \rightarrow (_2S)_2, S \rightarrow (_2)_2$. Note that the grammar is independent of the input size, i.e., $\Gamma$ has constant size.[13]

We construct a directed graph $G' = (V', E')$ by starting from the graph $G$, adding a directed path of length $\ell$ on nodes $u_0, u_1, \ldots, u_\ell$, and identifying the nodes $t$ and $u_0$. In other words, we attach a path $u_1, \ldots, u_\ell$ to the node $t$, and we use $u_0$ as another name for node $t$. We set $s' := s$ and $t' := u_\ell$. For each edge $(u, v) \in E$ we set the label to $\sigma(e) := (_{c(u,v)}$. For each $1 \leq i \leq \ell$ we set the label $\sigma(u_{i-1}, u_i) := )_{c_{\ell+1-i}}$. This finishes the construction of the CFL Reachability instance $(\Gamma, G', s', t', \sigma)$.

Note that any walk from $s'$ to $t'$ in $G'$ ends with the path $u_0, u_1, \ldots, u_\ell$ and thus with the labels $)_{c_\ell} )_{c_{\ell-1}} \ldots )_{c_2} )_{c_1}$. Since there are no other edge labels with closing brackets, we must choose a walk from $s' = s$ to $u_0 = t$ spelling the word $(_{c_1} (_{c_2} \ldots (_{c_{\ell-1}} (_{c_\ell}$ in order to match all parentheses. Such a walk corresponds to a walk from $s$ to $t$ in $G$ with color sequence $c_1, c_2, \ldots, c_\ell$. This shows that the constructed CFL Reachability instance is a YES-instance if and only if the given Colored Walk instance is a YES-instance, and thus shows correctness of the reduction.

Note that the constructed graph $G'$ consists of $n + \ell = O(n)$ nodes (using our assumption on $\ell$). Therefore, any algorithm for CFL Reachability running in time $O(n^{3-\varepsilon})$ solves the given Colored Walk instance in time $O(n^{3-\varepsilon})$. This contradicts the NFA Acceptance hypothesis (as discussed in the first paragraph). ∎

## 4. Hardness of Word Break

Recall that in the Word Break problem, we are given a string $S$ and a dictionary $D$ (i.e., $D$ is a set of strings) and we ask whether $S$ can be split into dictionary words, i.e., whether we can partition $S$ into substrings such that each substring is in the set $D$. We denote the length of $S$ by $n$ and the total length of all strings in $D$ by $m$. In this section, we prove that the NFA Acceptance hypothesis implies optimality (up to subpolynomial factors) of the $O(n(m \log m)^{1/3} + m)$ time algorithm given in [15].

**THEOREM 4.1.** *The Word Break problem has no $O(nm^{1/3-\varepsilon} + m)$-time algorithm for any $\varepsilon > 0$ assuming the NFA Acceptance hypothesis. This even holds restricted to $m = \Theta(n^\gamma)$ for any constant $\gamma \in (0, 3/2)$.*

Note that the $O(n(m \log m)^{1/3} + m)$ algorithm from [15] solves Word Break in time $\widetilde{O}(m)$ whenever $m = \Omega(n^{3/2})$. Since this is near-linear, there is no need for proving a fine-grained lower bound in this case. Therefore, it is natural that we assume $\gamma < 3/2$ in the above theorem.

---

13    This special case of the CFL Reachability problem, where $\Gamma$ is Dyck-2, is also known as Dyck-2 Reachability.

**PROOF OF THEOREM 4.1.** In the following proof, we use the upper case letters $N$ and $M$ for the parameters of Word Break in order to differentiate them from the parameters of Colored Walk.

We reduce from Directed Edge-2-Colored Walk. To this end, we are given a directed graph $G = (V, E)$ with colors $c\colon E \to \{1, 2\}$, designated vertices $s, t \in V$, and a color sequence $c_1, \ldots, c_\ell \in \{1, 2\}$. We write $V = \{1, \ldots, n\}$ and let $m = |E|$. We assume the graph to be dense (i.e., $m = \Theta(n^2)$) and we assume $\ell = \Theta(n^\beta)$ for $\beta := 3/\gamma - 1$. By the NFA Acceptance hypothesis in the setting $\alpha = 2, \beta = 3/\gamma - 1$, there is no algorithm solving all such instances of Directed Edge-2-Colored Walk in time $O(n^{2+\beta-\varepsilon})$ for any $\varepsilon > 0$.

We use the notation $0^k$ to denote the string of length $k$ consisting of $k$ times the letter 0. We construct the string $S$ and the dictionary $D$ over alphabet $\{0, 1, 2\}$ as follows:

$$S := 0^s \, c_1 \, 0^n \, c_2 \, 0^n \, c_3 \, \ldots \, 0^n \, c_{\ell-1} \, 0^n \, c_\ell \, 0^{n-t}.$$

$$D := \{\, 0^u \, c(u, v) \, 0^{n-v} \mid (u, v) \in E\}.$$

**Correctness**  We claim that the string $S$ can be split into dictionary words if and only if there is a walk from $s$ to $t$ with color sequence $c_1, \ldots, c_\ell$ in $G$. This is straightforward to show: To match the prefix $0^s c_1$ we must choose a dictionary word corresponding to an edge $(s, v_1)$ of color $c_1$, and the remaining string starts with the prefix $0^{v_1} c_2$. Generally, in the $i$th step the remaining string is of the form $0^{v_i} c_i 0^n c_{i+1} \ldots 0^n c_\ell 0^{n-t}$, so in order to match the prefix $0^{v_i} c_i$ we must choose a dictionary word corresponding to an edge $(v_i, v_{i+1})$ of color $c_i$. The last vertex $v_\ell$ must satisfy $v_\ell = t$ in order to match the suffix $0^{n-t}$. Hence, any valid partitioning of the string $S$ into dictionary words corresponds to a walk $s = v_0, v_1, \ldots, v_\ell = t$ with color sequence $c_1, \ldots, c_\ell$ in $G$, and this is an equivalence.

**Running Time**  The length of the string $S$ is $N = \Theta(\ell n) = \Theta(n^{1+\beta})$, by the assumption on $\ell$. The total length of all strings in $D$ is $M = \Theta(nm) = \Theta(n^3)$, by the assumption that $G$ is dense. Note that $M = \Theta(N^{3/(1+\beta)}) = \Theta(N^\gamma)$ by our setting of $\beta = 3/\gamma - 1$, so we constructed instances with the desired setting of parameters. If Word Break can be solved in time $O(NM^{1/3-\varepsilon} + M)$, then by plugging in the bounds on $N$ and $M$, our setting of Directed Edge-2-Colored Walk can be solved in time $O(n^{1+\beta}(n^3)^{1/3-\varepsilon} + n^3) = O(n^{2+\beta-3\varepsilon} + n^3)$. Since $\gamma < 3/2$ we have $\beta = 3/\gamma - 1 > 1$ and thus by setting $\varepsilon' := \min\{3\varepsilon, 2 + \beta - 3\} > 0$ we can bound the running time by $O(n^{2+\beta-\varepsilon'})$. This contradicts the NFA Acceptance hypothesis in the setting $\alpha = 2, \beta = 3/\gamma - 1$.  ∎

## 5.  Hardness of OMv

In the Online Boolean Matrix-Vector Multiplication (OMv) problem, an algorithm is initially given an $n \times n$ Boolean matrix $M$. Then the following repeats for $n$ rounds: In the $i$th round, the algorithm is given an $n$-dimensional Boolean vector $v_i$ and has to compute $Mv_i$. The algorithm

must compute the output $Mv_i$ before it can proceed to the next round. The running time of an OMv algorithm is the total running time for the initialization and all $n$ rounds together.

By naively multiplying $Mv_i$ in time $O(n^2)$ in each of the $n$ rounds, OMv can be solved in time $O(n^3)$. This running time has been improved to $n^3/2^{\Omega(\sqrt{\log n})}$ [44]. The OMv Hypothesis, due to Henzinger et al. [35], postulates that OMv cannot be solved in strongly subcubic time.

**HYPOTHESIS 5.1 (OMv Hypothesis).** For any constant $\varepsilon > 0$, OMv has no $O(n^{3-\varepsilon})$-time algorithm (with an error probability of at most 1/3).

Many data structure problems and dynamic problems have matching lower bounds under the OMv Hypothesis, as its usefulness has been established in various areas such as graph algorithms [35, 27, 4, 36], string algorithms [24, 41, 23], computational geometry [28, 45], linear algebra [39], formal languages [33], and database theory [13, 18, 40].

We show the following relation to the NFA Acceptance hypothesis:

**THEOREM 5.2.** *The NFA Acceptance hypothesis implies the OMv Hypothesis.*

In particular, all implications of the OMv Hypothesis shown in [35, 27, 4, 36, 24, 41, 23, 28, 45, 39, 33, 13, 18, 40] also hold under the NFA Acceptance hypothesis.

**PROOF OF THEOREM 5.2.** We reduce from Directed Edge-2-Colored Walk. To this end, we are given a directed graph $G = (V, E)$ with colors $c \colon E \to \{1, 2\}$, designated vertices $s, t \in V$, and a color sequence $c_1, \ldots, c_\ell \in \{1, 2\}$. We assume the graph to be dense ($m = \Theta(n^2)$) and we assume $\ell = \Theta(n)$. By the NFA Acceptance hypothesis in the setting $\alpha = 2, \beta = 1$, there is no algorithm solving all such instances of Directed Edge-2-Colored Walk in time $O(n^{3-\varepsilon})$ for any $\varepsilon > 0$.

Let $N := \max\{n, \ell\}$. If $N > n$ we add $N - n$ isolated dummy vertices to $G$.

Consider for each color $c \in \{1, 2\}$ the *transposed* adjacency matrix $M^{(c)}$ corresponding to the edges with color $c$, i.e., $M^{(c)} \in \{0, 1\}^{N \times N}$ where $M^{(c)}_{u,v} = 1$ if and only if $(v, u) \in E$ and $c(v, u) = c$. Let $u_0 \in \{0, 1\}^N$ be the indicator vector for $s$, i.e., $u_0[v] = 1$ if and only if $v = s$.

Suppose there is an algorithm $\mathcal{A}$ solving OMv in time $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$. We use $\mathcal{A}$ to preprocess $M^{(1)}$ and $M^{(2)}$ (as independent OMv instances). For each $i = 1, \ldots, \ell$, we compute $u_i := M^{(c_i)} u_{i-1}$ using algorithm $\mathcal{A}$ on the corresponding OMv instance. Finally, we accept the colored walk instance if and only if $u_\ell[t] = 1$.

Inductively, it is straightforward to show that $u_i[v] = 1$ if and only if there is a walk from $s$ to $v$ with color sequence $c_1, \ldots, c_i$; hence correctness follows. Note that for each of the two OMv instances we execute at most $\ell \le N$ rounds (and we can add dummy rounds to obtain exactly $N$ rounds). Therefore, algorithm $\mathcal{A}$ solves both instances in total time $O(N^{3-\varepsilon})$ over all rounds. Since $N = \max\{n, \ell\} = O(n)$ by our assumption on $\ell$, it follows that we can solve the given Directed Edge-2-Colored Walk instance in time $O(n^{3-\varepsilon})$, contradicting the NFA Acceptance hypothesis (as discussed in the first paragraph). This finishes the proof.

*Remark:* As described above, the reduction is not many-one, since we create two instances of OMv corresponding to the matrices $M^{(1)}$ and $M^{(2)}$. However, by a slight adaptation we can make the reduction many-one. To this end, we construct the matrix

$$M = \begin{pmatrix} M^{(1)} & 0 \\ 0 & M^{(2)} \end{pmatrix},$$

which stacks the matrices $M^{(1)}, M^{(2)}$ in block matrices along the main diagonal. We use algorithm $\mathcal{A}$ to preprocess the OMv instance $M$. Then given a vector $u \in \{0,1\}^N$ and a color $c \in \{1,2\}$, we can compute the Boolean product $M^{(c)}u$ by one call to the OMv instance $M$: To this end, we set $u' := (u_1, \ldots, u_N, 0, \ldots, 0) \in \{0,1\}^{2N}$ if $c = 1$, and $u' := (0, \ldots, 0, u_1, \ldots, u_N) \in \{0,1\}^{2N}$ if $c = 2$, and we call $\mathcal{A}$ to compute $Mu'$. This yields $M^{(c)}u$, padded with some zeroes. Therefore, each step $u_i := M^{(c_i)}u_{i-1}$ performed in the above reduction can be implemented by one call to the OMv instance $M$. This makes the reduction many-one. ∎

## 6.  Equivalences of Colored Walk

In this section, we prove the equivalences of Colored Walk problems stated in Lemma 2.3. We also prove an equivalence to a version without source and target nodes, see Lemma 6.12 at the end of this section. Throughout this section we abbreviate Colored Walk as CW.

**LEMMA 2.3** (Colored Walk Framework). **(Restated)** *Let $\alpha \in [1,2]$, $\beta > 0$. All of the following statements A1, A2, A3, A4, A5, B1, B2, B3, and B4 are equivalent:*

 A  *Restricted to instances with $m = \Theta(n^\alpha)$ and $\ell = \Theta(n^\beta)$, there is no $O(n^{\alpha+\beta-\varepsilon})$-time algorithm for any $\varepsilon > 0$ for the problem...*

    1  *... NFA Acceptance with alphabet size $n^{o(1)}$.*

    2  *... Directed Node-2-Colored Walk.*

    3  *... Directed Node-n-Colored Walk.*

    4  *... Directed Edge-2-Colored Walk.*

    5  *... Directed Edge-$n^{o(1)}$-Colored Walk.*

 B  *Restricted to instances with $m = \Theta(n^\alpha)$ and $\ell = O(n^\beta)$, there is no $O(n^{\alpha+\beta-\varepsilon})$-time algorithm for any $\varepsilon > 0$ for the problem...*

    1  *... Undirected Node-2-Colored Walk.*

    2  *... Undirected Node-n-Colored Walk.*

    3  *... Undirected Edge-2-Colored Walk.*

    4  *... Undirected Edge-$n^{o(1)}$-Colored Walk.*

For the node version of the problem, we say that the *(node) color sequence* of a walk $v_0, v_1, \ldots, v_k$ is the sequence $c(v_1), c(v_2), \ldots, c(v_k)$. Similarly, for the edge version we say that the *(edge) color sequence* of a walk $v_0, v_1, \ldots, v_k$ is the sequence $c(v_0, v_1), c(v_1, v_2), \ldots, c(v_{k-1}, v_k)$.

The remainder of this section is devoted to the proof of Lemma 2.3 (and to the proof of Lemma 6.12). To this end, we define the following notion of reductions. Here, a *parameter* is a function mapping any instance to a natural number, e.g., $n, m, \ell$ are parameters of the above problems.

**DEFINITION 6.1.** Let $X, Y$ be problems with the same set of parameters $P$. We say that there is a *P-preserving (many-one) reduction from X to Y*, written $X \leq_P Y$ if there is an algorithm $A$ that given an instance $I$ of $X$ computes an equivalent instance $J$ of $Y$ such that for all parameters $p \in P$ we have $p(J) \leq p(I)^{1+o(1)}$, and $A$ runs in almost-linear time $N^{1+o(1)}$ in its input size $N$.

The following lemmas show $\{n, m, \ell\}$-preserving reductions between the problems from Lemma 2.3.

**LEMMA 6.2.** *Directed Node-2-CW $\leq_{\{n,m,\ell\}}$ Directed Edge-2-CW.*

**PROOF.** Given a directed graph $G$ with node coloring $c \colon V \to \{1, 2\}$, we define the edge coloring $c' \colon E \to \{1, 2\}$ by setting $c'(u, v) := c(v)$ for all edges $(u, v) \in E$. This yields an equivalent Directed Edge-2-CW instance, since for any walk $v_0, v_1, \ldots, v_\ell$ in $G$, the sequence of node colors $c(v_1), \ldots, c(v_\ell)$ is the same as the sequence of edge colors $c'(v_0, v_1), \ldots, c'(v_{\ell-1}, v_\ell)$. The reduction preserves the exact values of all parameters. ∎

**LEMMA 6.3.** *Directed Edge-$n^{o(1)}$-CW $\leq_{\{n,m,\ell\}}$ NFA Acceptance with $n^{o(1)}$ terminals.*

**PROOF.** This is a simple statement, as NFA Acceptance with $n^{o(1)}$ terminals is the generalization of Directed Edge-$n^{o(1)}$-CW where we allow loops and we allow multiple edges (with different labels) between two nodes. Since the two problems are formulated in a different language, we make the correspondence explicit: Given a directed graph $G = (V, E)$ with designated vertices $s, t \in V$ and edge coloring $c \colon E \to \{1, \ldots, C\}$ as well as a color sequence $c_1, \ldots, c_\ell \in \{1, \ldots, C\}$, we construct the NFA $M = (Q, \Sigma, \delta, q_0, F)$ by setting $Q := V$, $\Sigma := \{1, \ldots, C\}$, $\delta := \{(u, c(u, v), v) \mid (u, v) \in E\}$, $q_0 := s$, $F := \{t\}$, and we construct the string $S := c_1 c_2 \ldots c_\ell \in \Sigma^\ell$. It is straightforward to show that $M$ accepts $S$ if and only if there is a walk from $s$ to $t$ with color sequence $c_1 c_2 \ldots c_\ell$ in $G$. All parameters are preserved since $|Q| = |V|$, $|\delta| = |E|$ and $\ell$ remains unchanged. ∎

**LEMMA 6.4.** *NFA Acceptance with $n^{o(1)}$ terminals $\leq_{\{n,m,\ell\}}$ Directed Node-$n^{o(1)}$-CW.*

**PROOF.** Given an NFA $M = (Q, \Sigma, \delta, q_0, F)$ with set of states $Q$, input alphabet $\Sigma$, transitions $\delta \subseteq Q \times \Sigma \times Q$, initial state $q_0$, and set of accepting states $F \subseteq Q$, as well as a string $S \in \Sigma^\ell$, we first transform it into an equivalent instance without loops and with exactly one accepting state.

To remove loops, we replace the states $Q$ by two copies: $\hat{Q} := Q \times \{1, 2\}$. Each transition $(q, \sigma, q')$ is replaced by two transitions $((q, 1), \sigma, (q', 2))$ and $((q, 2), \sigma, (q', 1))$; note that this ensures that we have no loops (i.e., no transition starts and ends in the same state). We also

replace $q_0$ by $\hat{q}_0 := (q_0, 1)$ and $F$ by $\hat{F} := F \times \{1, 2\}$. It is easy to check that the new NFA accepts $S$ if and only if the old NFA accepts $S$, and the size of the NFA is only changed by a constant factor.

To ensure exactly one accepting state, we add a new state $f_0$ to $\hat{Q}$. We fix an arbitrary symbol $\sigma \in \Sigma$, and we add transitions $(f, \sigma, f_0)$ for all $f \in \hat{F}$. Finally, we replace $\hat{F}$ by $\{f_0\}$ and $S$ by $S\sigma$. Note that this results in an equivalent instance with exactly one accepting state and all sizes stay the same up to constant factors. Therefore, in the following we assume without loss of generality that the given NFA $M = (Q, \Sigma, \delta, q_0, F)$ satisfies $F = \{f_0\}$ and has no loops. Moreover, we can assume without loss of generality that $\Sigma = \{1, \dots, |\Sigma|\}$.

We define a graph $G = (V, E)$ with vertex set $V = Q \times \Sigma$. For each transition $(q, \sigma', q') \in \delta$, we add the edges $((q, \sigma), (q', \sigma')) \in E$ for all $\sigma \in \Sigma$. We define the vertex coloring $c \colon V \to \{1, \dots, |\Sigma|\}$ by $c((q, \sigma)) = \sigma$. Observe that for any choice of "starting color" $\sigma_0 \in \{1, \dots, |\Sigma|\}$, the mapping from any transition sequence $q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \dots \xrightarrow{\sigma_\ell} q_\ell$ in $M$ to the walk $v_0 := (q_0, \sigma_0)$, $v_1 := (q_1, \sigma_1), \dots, v_\ell := (q_\ell, \sigma_\ell)$ in $G$ is indeed a one-to-one mapping from transition sequences in $M$ to walks in $G$ starting in the node $(q_0, \sigma_0)$ such that the string read by the transition sequence equals the color sequence of the walk. In particular, $M$ accepts the string $S$ if and only if there is a walk from $s := (q_0, \sigma_0)$ to $t := (f_0, S[\ell])$ in $G$ with color sequence $S[1], \dots, S[\ell]$. Thus, for any NFA $M$ we can construct, in linear time in the output, an equivalent Directed Node-$|\Sigma|$-CW instance.

We verify that the parameters are preserved for $|\Sigma| = |Q|^{o(1)}$: The number of vertices in $G$ is bounded by $|V| = |Q| \cdot |\Sigma| = |Q|^{1+o(1)}$, similarly, we have $|E| \le |\delta| \cdot |\Sigma| = |\delta|^{1+o(1)}$. Finally, the length $\ell$ of the color sequence equals the length of the string $S$. ∎

**LEMMA 6.5.** *Directed Node-n-CW* $\le_{\{n,m,\ell\}}$ *Directed Node-2-CW.*

**PROOF.** Given a directed graph $G$ with node coloring $c \colon V \to \{1, \dots, C\}$, we let $B := \lceil \log_2 C \rceil$ and define a graph $G' = (V', E')$ with vertex set $V' = V \times [B]$. For every edge $(u, v) \in E$, we add the edge $((u, B), (v, 1)) \in E'$, and for every $v \in V$ and $1 \le i < B$, we add the "path" edge $((v, i), (v, i+1)) \in E'$. For a color $c \in \{1, \dots, C\}$, let $\text{bin}(c, i)$ be the $i$-th bit in the $B$-bit binary representation of $c$. We define the node coloring $c' \colon V' \to \{1, 2\}$ by $c'((v, i)) = \text{bin}(c(v), i) + 1$ for all $v \in V, i \in [B]$. Thus, for any Directed Node-$n$-CW instance $G, s, t, (c_1, \dots, c_\ell)$, we define a corresponding instance on the graph $G'$ with source $(s, B)$, target $(t, B)$ and color sequence $c'_1, \dots, c'_{\ell B}$ where we set $c'_{(j-1)B+i} := \text{bin}(c_j, i) + 1$ for all $1 \le j \le \ell$ and $1 \le i \le B$. It is straightforward to verify that $v_0, v_1, \dots, v_\ell$ is a walk in $G$ with color sequence $c_1 = c(v_1), \dots, c_\ell = c(v_\ell)$ if and only if $v'_0, v'_1, \dots, v'_{\ell B}$ with $v'_0 = (v_0, B)$ and $v'_{(j-1)B+i} = (v_j, i), 1 \le j \le \ell, 1 \le i \le B$ is a walk in $G'$ with color sequence $c'_1 = c'(v'_1), \dots, c'_{\ell B} = c'(v'_{\ell B})$.

Note that for $C = n$, we have $B = O(\log n)$ and thus $|V'| = B|V| \le |V|^{1+o(1)}$, $|E'| = |E| + |V|B \le |E|^{1+o(1)}$ and $\ell B \le \ell^{1+o(1)}$. ∎

To reduce from Node-2-CW in a *directed* graph $G$ to Edge-2-CW in an *undirected* graph $G'$, a natural attempt is the following: each vertex $v$ is replaced by two vertices, $v_{\text{in}}$ and $v_{\text{out}}$, such

that a directed edge $(u, v)$ can be represented by the undirected edge $\{u_{\text{out}}, v_{\text{in}}\}$. Additionally, we introduce the edges $\{v_{\text{in}}, v_{\text{out}}\}$ of color $c(v)$. If we were allowed to introduce an additional color $\tilde{c} \notin \{1, 2\}$, we could label each edge $\{u_{\text{out}}, v_{\text{in}}\}$ with $\tilde{c}$, and any walks with node color sequence $c_1, c_2, \ldots, c_\ell$ in $G$ would uniquely correspond to walks with edge colors sequence in $\tilde{c}, c_1, \tilde{c}, c_2, \ldots \tilde{c}, c_\ell$ and vice versa. To avoid the blow-up in the number of colors, however, we must reuse a color for $\tilde{c}$, say $\tilde{c} = 1$. Since this would allow a walk to use an edge $\{u_{\text{in}}, v_{\text{out}}\}$ in the "reverse" direction towards $u_{\text{in}}$ whenever we are supposed to "check" that a node has color 1, we might obtain illegal transitions in the resulting walk. An analogous reasoning applies for a reduction to undirected Node-2-CW.

We use slightly more involved gadgetry for both reductions: For a given directed graph $G = (V, E)$ and an integer $P$, we create an *undirected* graph $G' = (V', E')$ with vertex set $V' = \{v_{\text{in}}, v_{\text{out}}, v^{(1)}, \ldots, v^{(P)} \mid v \in V\}$ as follows: we introduce, for each edge $(u, v) \in E$, the edges $\{u_{\text{out}}, v_{\text{in}}\}$ in $E'$, as well as, for all $v \in V$, all "path edges" $\{v_{\text{in}}, v^{(1)}\}$, $\{v^{(P)}, v_{\text{out}}\}$ and $\{v^{(i)}, v^{(i+1)}\}$ for all $1 \le i < P$. By choosing the color sequences for all path vertices/edges appropriately, we will be able to enforce that every edge is used in "forward" direction, i.e., any feasible walk traverses an edge $\{u_{\text{out}}, v_{\text{in}}\}$, then all path edges toward $v_{\text{out}}$, then an edge $\{v_{\text{out}}, w_{\text{in}}\}$, and so on.

**LEMMA 6.6.** *Directed Node-2-CW $\le_{\{n,m,\ell\}}$ Undirected Edge-2-CW.*

**PROOF.** Given a directed graph $G = (V, E)$, we construct the undirected version $G'$ using path length $P = 4$ as described above. We define the coloring $c' : E \to \{1, 2\}$ as follows: for all $(u, v) \in E$, we set $c'(u_{\text{out}}, v_{\text{in}}) = 1$. For each $v \in V$, we set the colors of the path edges to

$$\left(c'(v_{\text{in}}, v^{(1)}), \; c'(v^{(1)}, v^{(2)}), \; c'(v^{(2)}, v^{(3)}), \; c'(v^{(3)}, v^{(4)}), \; c'(v^{(4)}, v_{\text{out}})\right) := (2, \; c(v), \; c(v), \; 1, \; 2).$$

The main property of this construction is captured by the following claim. In the remainder of the proof, for any $c \in \{1, 2\}$ and $v \in V$ we set:

$$\text{col}(c) := (2, c, c, 1, 2) \quad \text{and} \quad \text{path}(v) := (v_{\text{in}}, v^{(1)}, v^{(2)}, v^{(3)}, v^{(4)}, v_{\text{out}}).$$

In what follows, for any walk $z = (z_0, \ldots, z_\ell)$ with color sequence $c' = (c'_1, \ldots, c'_\ell)$, we use the shorthands $z_{\ldots j} = (z_0, \ldots, z_j)$ and $c'_{\ldots j} = (c'_1, \ldots, c'_j)$ to denote the length-$j$ prefix of $z$ and its corresponding color sequence, respectively.

**CLAIM 6.7.** *Let $v \in V$ and $c \in \{1, 2\}$. There is a walk $z = (z_0, z_1, \ldots, z_5)$ in $G'$ with $z_0 = v_{\text{in}}$ and edge color sequence $\text{col}(c)$ if and only if $c = c(v)$ and $z = \text{path}(v)$.*

**PROOF.** If $c = c(v)$, then the walk $\text{path}(v)$ clearly has edge color sequence $(2, c, c, 1, 2)$. For the converse, let $z = (z_0, \ldots, z_5)$ be a walk with $z_0 = v_{\text{in}}$ and color sequence $\text{col}(c)$. We analyze the possible values for $z_{\ldots i}$ when observing the color sequence $\text{col}(c)_{\ldots i}$ for $i = 1, \ldots, 5$:

— $z_1 = v^{(1)}$ and thus $z_{\ldots 1} = \text{path}(v)_{\ldots 1}$ always holds, as $\{v_{\text{in}}, v^{(1)}\}$ is the only edge adjacent to $v_{\text{in}}$ that has color 2, because the edges $\{u_{\text{out}}, v_{\text{in}}\}$ have color 1.

— Consider the case $c = 1$:

    — If $c(v) \neq c = 1$, then there is no edge leaving $z_1 = v^{(1)}$ with color $c = 1$. Thus we may assume in the remainder of the case that $c(v) = c = 1$.

    — It follows that $z_2 = v^{(2)}$ and hence $z_{\ldots 2} = \text{path}(v)_{\ldots 2}$.

    — Since both edges adjacent to $z_2 = v^{(2)}$ have color $c(v) = c = 1$, we have either $z_{\ldots 3} = \text{path}(v)_{\ldots 3}$ or $z_3 = v^{(1)}$.

    — Since the only nodes adjacent to $v^{(3)}$ or $v^{(1)}$ via an edge with color 1 are $v^{(4)}$ and $v^{(2)}$, we have that $z_4 \in \{v^{(4)}, v^{(2)}\}$, where $z_4 = v^{(4)}$ only occurs if $z_{\ldots 4} = \text{path}(v)_{\ldots 4}$.

    — Finally, as desired, $z_5 = v_{\text{out}}$ must hold, since among $\{v^{(4)}, v^{(2)}\}$, only $v^{(4)}$ has an adjacent edge of color 2, which leads to $v_{\text{out}}$. This yields $z = \text{path}(v)$ for $c = 1$.

— We analyze the remaining case $c = 2$:

    — We have $z_2 = v_{\text{in}}$, or, only if $c(v) = c = 2$, possibly $z_2 = v^{(2)}$ and hence $z_{\ldots 2} = \text{path}(v)_{\ldots 2}$.

    — Likewise, since $\{v_{\text{in}}, v^{(1)}\}$ is the only edge adjacent to $v_{\text{in}}$ with color $c = 2$, we must have $z_3 = v^{(1)}$, or, only if $c(v) = c = 2$ and $z_{\ldots 2} = \text{path}(v)_{\ldots 2}$, possibly $z_3 = v^{(3)}$ (and hence $z_{\ldots 3} = \text{path}(v)_{\ldots 3}$).

    — Note that $v^{(1)}$ only has an adjacent edge of color 1 if $c(v) = 1 \neq c$. In contrast, if $c(v) = c = 2$, $v^{(3)}$ has its only adjacent edge of color 1 to $v^{(4)}$. Thus, we have either $z_4 = v^{(2)}$, which can only happen if $c(v) \neq c$, or $z_4 = v^{(4)}$, which can only happen if $c(v) = c$ and $z_{\ldots 3} = \text{path}(v)_{\ldots 3}$. In the latter case, we thus must have $z_{\ldots 4} = \text{path}(v)_{\ldots 4}$.

    — Finally, $z_5 = v_{\text{out}}$, $c(v) = c$ and $z_{\ldots 4} = \text{path}(v)_{\ldots 4}$ must hold, since in the case $c \neq c(v)$, $v^{(2)}$ has no adjacent edges of color 2, while $v^{(4)}$ has an edge of color 2 to $v_{\text{out}}$. Thus $z = \text{path}(v)$ and $c(v) = c$.

                                                                  ■

Given a Directed Node-2-CW instance $G$, $s, t \in V$ and $c_1, \ldots, c_\ell$, we construct, in linear time in the output, an Undirected Edge-2-CW instance $G'$ with source $s_{\text{out}}$, target $t_{\text{out}}$ and color sequence $1, \text{col}(c_1), 1, \text{col}(c_2), \ldots, 1, \text{col}(c_\ell)$. For any walk $s = v_0, v_1, \ldots, v_\ell = t$ with color sequence $c_1, \ldots, c_\ell$ in $G$, the walk $v_0, \text{path}(v_1), \text{path}(v_2), \ldots, \text{path}(v_\ell)$ in $G'$ has the desired color sequence $1, \text{col}(c_1), 1, \text{col}(c_2), \ldots, 1, \text{col}(c_\ell)$ by the above claim. Conversely, we see that for each color substring $1, \text{col}(c_i)$ any walk in $G'$ that starts in some node $u_{\text{out}}$ is of the form $(u_{\text{out}}, \text{path}(v))$ for some $v \in V$ with $(u, v) \in E$ and $c(v) = c_i$, since the only 1-colored edges adjacent to $u_{\text{out}}$ lead to some $v_{\text{in}}$ with $(u, v) \in E$ and the above claim proves that the walk $\text{path}(v)$ must follow, which requires $c(v) = c_i$. By repeated application of this fact, any walk in $G'$ with color sequence $1, \text{col}(c_1), 1, \text{col}(c_2), \ldots, 1, \text{col}(c_\ell)$ corresponds to a walk in $G$ with color sequence $c_1, \ldots, c_\ell$, as desired. Note that all parameters have increased by at most a constant factor, which yields the desired reduction.                   ■

     For a reduction to Undirected *Node*-2-CW, we imitate the above reduction by defining a suitable node color sequence for path nodes. The analysis is slightly simpler than for Edge-2-CW.

**LEMMA 6.8.** *Directed Node-2-CW $\leq_{\{n,m,\ell\}}$ Undirected Node-2-CW.*

**PROOF.** Given a directed graph $G = (V, E)$, we construct the undirected version $G'$ using path length $P = 4$ as described above. We define the coloring $c' \colon V' \to \{1, 2\}$, as follows: for all $v \in V$, we set $c'(v_{\text{in}}) = c'(v_{\text{out}}) = 1$ and set

$$(c'(v^{(1)}), \; c'(v^{(2)}), \; c'(v^{(3)}), \; c'(v^{(4)})) := (2, \; c(v), \; 2, \; 2).$$

The main property of this construction is captured by the following claim. In the remainder of the proof, for any $c \in \{1, 2\}$ and $v \in V$ we set:

$$\text{col}(c) := (2, c, 2, 2, 1) \qquad \text{and} \qquad \text{path}(v) := (v_{\text{in}}, v^{(1)}, v^{(2)}, v^{(3)}, v^{(4)}, v_{\text{out}}).$$

Recall that for any walk $z = (z_0, \ldots, z_\ell)$ with color sequence $c' = (c'_1, \ldots, c'_\ell)$, we use the shorthands $z_{\ldots j} = (z_0, \ldots, z_j)$ and $c'_{\ldots j} = (c'_1, \ldots, c'_j)$ to denote the length-$j$ prefix of $z$ and its corresponding color sequence, respectively.

**CLAIM 6.9.** *Let $v \in V$ and $c \in \{1, 2\}$. There is a walk $z = (z_0, z_1, \ldots, z_5)$ with $z_0 = v_{\text{in}}$ and color sequence $\text{col}(c)$ in $G'$ if and only if $c = c(v)$ and $z = \text{path}(v)$.*

**PROOF.** If $c = c(v)$, then the walk $\text{path}(v)$ clearly has node-color sequence $(2, c, 2, 2, 1)$. For the converse, let $z = (z_0, \ldots, z_5)$ be a walk with $z_0 = v_{\text{in}}$ and color sequence $\text{col}(c)$. We analyze the possible values for $z_{\ldots i}$ when observing the color sequence $\text{col}(c)_{\ldots i}$ for $i = 1, \ldots 5$:

— We have $z_1 = v^{(1)}$ and thus $z_{\ldots 1} = \text{path}(v)_{\ldots 1}$, since $v^{(1)}$ is the only neighbor of $v_{\text{in}}$ with color 2.

— We have $z_2 = v_{\text{in}}$ (which might happen if $c = 1$), or $z_2 = v^{(2)}$ (which can happen if and only if $z_{\ldots 1} = \text{path}(v)_{\ldots 1}$ and $c = c(v)$). In the latter case, this yields $z_{\ldots 2} = \text{path}(v)_{\ldots 2}$ and $c = c(v)$.

— Since the only neighbor of $v_{\text{in}}$ with color 2 is $v^{(1)}$, we have either that $z_3 = v^{(1)}$ or, only if $z_{\ldots 2} = \text{path}(v)_{\ldots 2}$ and $c = c(v)$, that $z_3 = v^{(3)}$. In the latter case, it holds that $z_{\ldots 3} = \text{path}(v)_{\ldots 3}$ and $c = c(v)$.

— It follows that $z_4 = v^{(2)}$ (which might happen if $c(v) = 2$), or $z_4 = v^{(4)}$ (which can happen only if $z_{\ldots 3} = \text{path}(v)_{\ldots 3}$ and $c = c(v)$). In the latter case, it holds that $z_{\ldots 4} = \text{path}(v)_{\ldots 4}$ and $c = c(v)$.

— Finally, since $v^{(2)}$ has no neighbor of color 1, we have $z_5 = v^{(5)}$, which can happen if and only if $z_{\ldots 4} = \text{path}(v)_{\ldots 4}$ and $c = c(v)$. This yields $z = \text{path}(v)$ and $c = c(v)$, as desired.

■

The remainder of the reduction is analogous to the proof of Lemma 6.6: Given a Directed Node-2-CW instance $G$, $s, t \in V$ and color sequence $c_1, \ldots, c_\ell$, we construct, in linear time in the output, an Undirected Node-2-CW instance $G'$ with source $s_{\text{out}}$, target $t_{\text{out}}$ and color sequence $1, \text{col}(c_1), 1, \text{col}(c_2), \ldots, 1, \text{col}(c_\ell)$. For any walk $s = v_0, v_1, \ldots, v_\ell = t$ with color sequence

$c_1, \ldots, c_\ell$ in $G$, the walk $v_0, \mathrm{path}(v_1), \mathrm{path}(v_2), \ldots, \mathrm{path}(v_\ell)$ in $G'$ has the desired color sequence $1, \mathrm{col}(c_1), 1, \mathrm{col}(c_2), \ldots, 1, \mathrm{col}(c_\ell)$ by the above claim. Conversely, we see that for each color substring $1, \mathrm{col}(c_i)$ any walk in $G'$ that starts in some $u_{\mathrm{out}}$ is of the form $(u_{\mathrm{out}}, \mathrm{path}(v))$ for some $v \in V$ with $(u,v) \in E$ and $c(v) = c_i$, since the only 1-colored nodes adjacent to $u_{\mathrm{out}}$ lead to some $v_{\mathrm{in}}$ with $(u,v) \in E$ and the above claim proves that the walk $\mathrm{path}(v)$ must follow, which requires $c(v) = c_i$. By repeated application of this fact, any walk in $G'$ with color sequence $1, \mathrm{col}(c_1), 1, \mathrm{col}(c_2), \ldots, 1, \mathrm{col}(c_\ell)$ corresponds to a walk in $G$ with color sequence $c_1, \ldots, c_\ell$, as desired. Note that all parameters have increased by at most a constant factor, which yields the desired reduction.                                                                                                    ■

Now we combine the above relations to a full equivalence.

**LEMMA 6.10.** *The following problems are equivalent under $\{n, m, \ell\}$-preserving reductions:*
— *NFA Acceptance with $n^{o(1)}$ terminals,*
— *Directed Node-2-Colored Walk,*
— *Directed Node-n-Colored Walk,*
— *Directed Edge-2-Colored Walk,*
— *Directed Edge-$n^{o(1)}$-Colored Walk,*
— *Undirected Node-2-Colored Walk,*
— *Undirected Node-n-Colored Walk,*
— *Undirected Edge-2-Colored Walk,*
— *Undirected Edge-$n^{o(1)}$-Colored Walk.*

**PROOF.** The parameter-preserving equivalence of all directed variants for parameters $P = \{n, m, \ell\}$ follows from the following chain of reductions:

$$
\begin{aligned}
\text{Directed Node-2-CW} \ &\leq_P \ \text{Directed Edge-2-CW} && \text{(by Lemma 6.2)} \\
&\leq_P \ \text{Directed Edge-}n^{o(1)}\text{-CW} && \text{(trivially by definition)} \\
&\leq_P \ \text{NFA Acceptance with } n^{o(1)} \text{ terminals} && \text{(by Lemma 6.3),} \\
&\leq_P \ \text{Directed Node-}n^{o(1)}\text{-CW} && \text{(by Lemma 6.4),} \\
&\leq_P \ \text{Directed Node-}n\text{-CW} && \text{(trivially by definition),} \\
&\leq_P \ \text{Directed Node-2-CW} && \text{(by Lemma 6.5).}
\end{aligned}
$$

Note that for any $C$, Undirected Node-$C$-CW $\leq_P$ Directed Node-$C$-CW and Undirected Edge-$C$-CW $\leq_P$ Directed Edge-$C$-CW follows trivially by replacing each undirected edge by directed edges in both directions, while reductions in the other direction (for $C = 2$) are given by Lemmas 6.6

and 6.8. We use these to also conclude equivalence to the undirected variants by

Undirected Node-2-CW $\leq_P$ Undirected Node-$n$-CW (trivially by definition)

$\leq_P$ Directed Node-$n$-CW (trivially, as argued above)

$\leq_P$ Directed Node-2-CW (by already established equivalence),

$\leq_P$ Undirected Node-2-CW (by Lemma 6.8),

and similarly,

Undirected Edge-2-CW $\leq_P$ Undirected Edge-$n^{o(1)}$-CW (trivially by definition)

$\leq_P$ Directed Edge-$n^{o(1)}$-CW (trivially, as argued above)

$\leq_P$ Directed Node-2-CW (by already established equivalence),

$\leq_P$ Undirected Edge-2-CW (by Lemma 6.6).

∎

**LEMMA 6.11.** *Let $\alpha \in [1, 2], \beta > 0$ and $c \geq 1$. Let $X, Y$ be any problems listed in Lemma 6.10. If problem $X$ restricted to instances with $m = O(n^\alpha)$ and $\ell = O(n^\beta)$ has no $O(n^{c-\varepsilon})$-time algorithm for any $\varepsilon > 0$, then the same holds for problem $Y$.*

**PROOF.** Suppose for the sake of contradiction that problem $Y$ restricted to instances with $m = O(n^\alpha)$ and $\ell = O(n^\beta)$ has an algorithm $\mathcal{A}$ running in time $O(n^{c-\varepsilon})$ for some $\varepsilon > 0$. Then we can solve problem $X$ as follows. Given an instance $I$ of problem $X$ with parameters $n, m, \ell$ satisfying $m = O(n^\alpha), \ell = O(n^\beta)$, run the $\{n, m, \ell\}$-preserving reduction to obtain an equivalent instance $I_0$ of $Y$ with parameters $n_0, m_0, \ell_0$. Recall that $n_0 \leq n^{1+o(1)}, m_0 \leq m^{1+o(1)}, \ell_0 \leq \ell^{1+o(1)}$. We set $n_1 := \max\{n_0, m_0^{1/\alpha}, \ell_0^{1/\beta}\}$. We add $n_1 - n_0$ isolated nodes/states to $I_0$, resulting in an instance $I_1$ with parameters $n_1, m_1 = m_0, \ell_1 = \ell_0$. Note that $m_1 \leq n_1^\alpha$ and $\ell_1 \leq n_1^\beta$, so we can run algorithm $\mathcal{A}$ on instance $I_1$. Since $I_1$ is equivalent to $I_0$, and thus equivalent to $I$, this solves the given instance $I$ of problem $X$.

Tracing the above inequalities, we observe that $n_1 \leq n^{1+o(1)}$, so the running time of algorithm $\mathcal{A}$ on instance $I_1$ is $O(n_1^{c-\varepsilon}) = n^{c-\varepsilon+o(1)}$. The reduction runs in almost-linear time in the input size, i.e., in time $(n + m + \ell)^{1+o(1)} = n^{\max\{\alpha,\beta\}+o(1)}$. Hence, the total time to solve instance $I$ is $O(n^{c-\varepsilon+o(1)} + n^{\max\{\alpha,\beta\}+o(1)})$. If $c \leq \max\{\alpha, \beta\}$ then the conclusion that $Y$ has no $O(n^{c-\varepsilon})$-time algorithm is trivial, so we can assume $c > \max\{\alpha, \beta\}$. After possibly replacing $\varepsilon$ by $\min\{\varepsilon, c - \max\{\alpha, \beta\}\}$, we can further bound the running time by $O(n^{c-\varepsilon+o(1)})$. Bounding $n^{o(1)} \leq O(n^{\varepsilon/2})$, the running time becomes $O(n^{c-\varepsilon/2}) = O(n^{c-\varepsilon'})$ for $\varepsilon' := \varepsilon/2$. This contradicts the assumption that problem $X$ restricted to instances with $m = O(n^\alpha), \ell = O(n^\beta)$ has no $O(n^{c-\varepsilon''})$-time algorithm for any $\varepsilon'' > 0$. ∎

Finally we are ready to prove Lemma 2.3.

**PROOF OF LEMMA 2.3.** Let $X, Y$ be any problems listed in the lemma statement. If the lemma claim corresponding to $X$ holds, then in particular problem $X$ restricted to instances with $m = O(n^\alpha)$ and $\ell = O(n^\beta)$ has no $O(n^{\alpha+\beta-\varepsilon})$-time algorithm for any $\varepsilon > 0$. Using the $\{n, m, \ell\}$-preserving reduction from $X$ to $Y$ guaranteed by Lemma 6.10, by Lemma 6.11 we obtain the same result for $Y$, i.e., problem $Y$ restricted to instances with $m = O(n^\alpha)$ and $\ell = O(n^\beta)$ has no $O(n^{\alpha+\beta-\varepsilon})$-time algorithm for any $\varepsilon > 0$. To arrive at the lemma claim corresponding to $Y$, we need to additionally ensure $m = \Omega(n^\alpha)$ (and for some $Y$ also $\ell = \Omega(n^\beta)$), which we achieve by padding, as follows.

To pad the parameter $m$, we add $n$ dummy nodes (or states) and among them we add $\Theta(n^\alpha)$ dummy edges (or transitions), to ensure $m = \Theta(n^\alpha)$. Since these new nodes and edges are disconnected from the old graph, they do not change the result.

For the directed variants of Colored Walk we can also pad the length $\ell$: We add new nodes $s_0, s_1$ and edges $(s_0, s_1)$ and $(s_1, s_0)$ of color 1 and an edge $(s_1, s)$ of color 2, and we change the color sequence to $1^{2\lceil n^\beta \rceil} 2\, c_1 \ldots c_\ell$. Observe that there is a walk from $s_0$ to $t$ with the new color sequence in the new graph if and only if there is a walk from $s$ to $t$ with the old color sequence $c_1 \ldots c_\ell$ in the old graph. An analogous construction can be used to pad the number of transitions in a given NFA. Therefore, we can assume $\ell = \Theta(n^\beta)$ for the directed variants of Colored Walk as well as for NFA Acceptance.

Since these constructions change $n$ only by a constant factor, we still rule out a running time of $O(n^{\alpha+\beta-\varepsilon})$. This finishes the proof. ∎

Finally, we also show that Colored Walk is equivalent to a formulation without given source and target nodes $s, t \in V$, specifically, the variant in which we are looking for *any* walk in $G$ with the prescribed color sequence $c_1, \ldots, c_\ell$. We call these formulations DIRECTED/UNDIRECTED $\Sigma$-EDGE-COLORED/$\Sigma$-NODE-COLORED ANYWALK.

Below, we prove a $\{n, m, \ell\}$-preserving equivalence of Undirected Edge-Colored Walk and Undirected Edge-Colored AnyWalk (up to an additive constant in the alphabet size); the proof of equivalence for all directed and/or node-colored variants is analogous (and in fact simpler).

**LEMMA 6.12** (Equivalence of $s$-$t$-version and any-walk version). *The following relationships hold:*

1. *Undirected Edge-2-Colored Walk $\leq_{\{n,m,\ell\}}$ Undirected Edge-4-Colored AnyWalk,*
2. *Undirected Edge-$\sigma$-Colored AnyWalk $\leq_{\{n,m,\ell\}}$ Undirected Edge-$(\sigma + 1)$-Colored Walk.*

**PROOF.** For 1, consider any Undirected Edge-2-Colored Walk instance $G = (V, E)$, $s, t \in V$ and $c_1, \ldots, c_\ell \in \{1, 2\}$. We construct a graph $G'$ from $G$ by adding two nodes $s', t'$, connecting $s'$ to $s$ by an edge labeled 3 and connecting $t$ to $t'$ by an edge labeled 4. Note that any walk in $G'$ with color sequence $3, c_1, \ldots, c_\ell, 4$ must start in $s'$, transition to $s$, use a walk from $s$ to $t$ in $G$ with color sequence $c_1, \ldots, c_\ell$, and finally transition to $t'$. Thus, $G'$ with color sequence $3, c_1, \ldots, c_\ell, 4$

yields an equivalent Undirected 4-Edge-Colored AnyWalk instance. The reduction is clearly $\{n, m, \ell\}$-preserving.

For 2, consider any Undirected Edge-2-Colored AnyWalk instance $G = (V, E)$, $c_1, \ldots, c_\ell \in [\sigma]$. We construct $G'$ from $G$ by adding two nodes $s', t' \in V$, connecting $s'$ to each node $v \in V$ by an edge labeled $\sigma+1$, as well as connecting each node $v \in V$ to $t'$ by an edge labeled $\sigma+1$. Note that there is a walk from $s'$ to $t'$ in $G$ with color sequence $(\sigma+1), c_1, \ldots, c_\ell, (\sigma+1)$ if and only if there is any walk in $G$ with color sequence $c_1, \ldots, c_\ell$. Thus, we have obtained an equivalent Undirected Edge-$(\sigma + 1)$-Colored Walk instance. Since we have added only two nodes, $O(n) = O(m)$ edges and two colors to the color sequence, the reduction is $\{n, m, \ell\}$-preserving. ∎

We obtain in particular that the NFA Acceptance hypothesis is equivalent to the $(m\ell)^{1-o(1)}$ barrier for Directed/Undirected Edge-4-Colored/Node-4-Colored AnyWalk.

## 7.    Relation to Other Hypotheses

For the reader's convenience, we restate the known connections between the NFA Acceptance hypothesis and standard fine-grained hypotheses, as described in Section 1.

### 7.1    Tight Lower Bound for Sparse NFAs Under SETH

In the Orthogonal Vectors (OV) problem, we are given vector sets $A, B \subseteq \{0, 1\}^d$, and the task is to determine whether there is an *orthogonal pair* $(a, b) \in A \times B$, i.e., for all $1 \le k \le d$ we have $a[k] = 0$ or $b[k] = 0$.

**HYPOTHESIS 7.1** (OV Hypothesis). For no $\epsilon > 0$ and $\beta > 0$, there is an algorithm solving OV with sets of size $n := |A|$ and $\ell := |B| = \Theta(n^\beta)$ and $d \le n^{o(1)}$ dimensions in time $O((n\ell)^{1-\epsilon}) = O(n^{(1+\beta)(1-\epsilon)})$.

This hypothesis is usually stated with sets of the same size, but is equivalent to the above version, see, e.g., [16, Lemma II.1]. The hypothesis is well known to be implied by the Strong Exponential Time Hypothesis (SETH) [69].

The following conditional lower bound can be found for different versions of the Colored Walk problem throughout the algorithmic literature, including [10, 15, 30, 57, 18] – recall that these turn out to be fine-grained equivalent due to Lemma 2.3.

**PROPOSITION 7.2.** *If there are $\epsilon, \beta > 0$ such that NFA Acceptance on an n-state NFA with $O(n)$ transitions and a string of length $\ell = \Theta(n^\beta)$ over $\Sigma = \{0, 1, 2\}$ can be solved in time $O((\ell n)^{1-\epsilon})$, then the OV Hypothesis fails.*

Note that this establishes a tight conditional lower bound for NFAs that are *sparse*.

**PROOF.** We give a $\{n, \ell\}$-preserving reduction from OV to NFA Acceptance. It is not difficult to see that the claimed lower bound follows, analogously to Lemma 6.11.

Given an OV instance $A, B \subseteq \{0, 1\}^d$ with $n = |A|$, $\ell = |B|$, $d = n^{o(1)}$, we construct an NFA $M$ over $\Sigma = \{0, 1, 2\}$ as follows: $M$ has initial state $s$, accepting state $t$. We write $A = \{a_1, \ldots, a_n\}$ and introduce, for every $1 \le i \le n$, the states $q_0^{(i)}, \ldots, q_d^{(i)}$. For every $1 \le i \le n$ and $1 \le k \le d$, we add the transitions $(q_{k-1}^{(i)}, 0, q_k^{(i)})$ and, if $a_i[k] = 0$, the transition $(q_{k-1}^{(i)}, 1, q_k^{(i)})$. Concluding the construction of $M$, we add transitions $(s, 2, q_0^{(i)})$ and $(q_d^{(i)}, 2, t)$ for all $1 \le i \le n$ and the loop transitions $(q, 0, q), (q, 1, q), (q, 2, q)$ for each $q \in \{s, t\}$.

To construct the string $S$, we write $B = \{b_1, \ldots, b_\ell\}$ and define $S := 2\, b_1\, 2\, \ldots\, 2\, b_\ell\, 2$, where we interpret $b_i$ as a length-$d$ string over $\{0, 1\}$.

We argue that $M$ accepts $S$ if and only if $A, B$ contain an orthogonal pair: By the structure of $M$ and $S$, the only accepting runs of $M$ have the following form. $M$ reads a prefix $2b_1 2 \ldots b_{j-1}$ for some $1 \le j \le \ell$ while staying in $s$, then uses the transition sequence

$$s \xrightarrow{2} q_0^{(i)} \xrightarrow{b_j[1]} q_1^{(i)} \xrightarrow{b_j[2]} \cdots \xrightarrow{b_j[d]} q_d^{(i)} \xrightarrow{2} t, \tag{1}$$

for some $1 \le i \le n$, and then stays in $t$ while reading the suffix $b_{j+1} 2 \ldots b_\ell 2$. By observing that the transition sequence (1) exists in $M$ if and only if $a_i, b_j$ are orthogonal, the claim follows.

Finally, we note that this indeed yields a $\{n, \ell\}$-preserving reduction: $M$ has $O(nd) = n^{1+o(1)}$ states, and $S$ has length $O(\ell d) = \ell^{1+o(1)}$, and they can be computed in linear time in the input. ∎

## 7.2  Tight Combinatorial Lower Bound from Triangle and $k$-Clique

In the $k$-Clique problem, we are given an undirected, simple graph $G = (V, E)$, and the task is to determine whether there is a set $C \subseteq V$ of $k$ nodes that are pairwise adjacent, i.e., $\{u, v\} \in E$ for all $u, v \in C$. The current state of the art for $k$-Clique detection is captured in the following hypothesis, see [55, 2].

**HYPOTHESIS 7.3** ($k$-Clique hypothesis [2]).

1. *Combinatorial version:* For no $k \ge 3$ and $\epsilon > 0$, there is a *combinatorial* algorithm solving $k$-Clique in time $O(n^{k(1-\epsilon)})$.

2. *Non-combinatorial version:* For no $k \ge 3$ and $\epsilon > 0$, there is an algorithm solving $k$-Clique in time $O(n^{k(\omega/3 - \epsilon)})$.

One can show the following $k$-Clique-based lower bounds, which in particular give a tight combinatorial lower bound for *dense* NFAs. By Lemma 2.2, this yields a tight combinatorial lower bound for all graph densities.

**PROPOSITION 7.4.** *NFA Acceptance has the following k-Clique-based lower bounds:*

1.  Combinatorial algorithms: *Unless the combinatorial k-Clique hypothesis fails, there are no $\epsilon, \beta > 0$ such that there is an $O((\ell n^2)^{1-\epsilon})$-time combinatorial algorithm for NFA Acceptance on an n-state NFA with $\Theta(n^2)$ transitions and a string of length $\ell = \Theta(n^\beta)$ over $\Sigma = \{0, 1, 2, 3\}$.*

2.  General algorithms: *Unless the k-Clique hypothesis fails, there are no $\epsilon, \beta > 0$ such that there is an $O((\ell n^2)^{\omega/3-\epsilon})$-time algorithm for NFA Acceptance on an n-state NFA with $\Theta(n^2)$ transitions and a string of length $\ell = \Theta(n^\beta)$ over $\Sigma = \{0, 1, 2, 3\}$.*

The above lower bound is implicit in the literature. Specifically, it can be obtained either by adapting the reduction from weighted $k$-Clique in [11] to the unweighted case, or as a special case of [1, Theorem I.5][14], or as an appropriate generalization of the reduction from Triangle Detection to NFA Acceptance with $\ell \approx n$ in [57], or as an appropriate generalization of the reduction from Triangle Detection to regular path queries with queries of length $\ell \approx n$ in [18].

For the reader's convenience, we present a simplified proof of this fact.

**PROOF OF PROPOSITION 7.4.** We prove the conditional lower bound for combinatorial algorithms. The claim for general algorithms follows from the same reduction.

Assume for the sake of contradiction that there are $\epsilon > 0$ and $\beta > 0$ such that NFA Acceptance on a $q$-state NFA with $\Theta(q^2)$ transitions and a string $S$ of length $\ell = \Theta(q^\beta)$ over $\Sigma = \{0, 1, 2, 3\}$ can be solved in time $O((\ell q^2)^{1-\epsilon}) = O(q^{(2+\beta)(1-\epsilon)})$.

We reduce from $(2k+k')$-clique where $k > (5+2\beta)/\epsilon$ and $k' = \lfloor \beta k \rfloor$. We let $V = \{0, \dots, n-1\}$, so that we can write $v$ in binary as a $\lceil \log n \rceil$-length bit string, which we call the *node ID* of $v$.

For any $t$, let $C(t)$ denote the set of $t$-cliques in $G$. We construct the string $S$ as follows:

$$ S := 2 \circ \bigcirc_{\{v_1,\dots,v_{k'}\} \in C(k')} \left( v_1^k \, v_2^k \, \dots \, v_{k'}^k \, 3 \, v_1^k \, v_2^k \, \dots \, v_{k'}^k \, 2 \right), $$

where $\circ$ denotes concatenation and $v^k$ for $v \in V$ denotes the $k$-fold concatenation of $v$'s node ID.

We turn to constructing the NFA $M$. First, for every $\{u_1, \dots, u_k\} \in C(k)$, we construct a clique gadget $CG(u_1, \dots, u_k)$. This gadget will accept a string of the form $v_1^k \, v_2^k \, \dots \, v_{k'}^k$ for $\{v_1, \dots, v_{k'}\} \in C(k')$ if and only if $\{u_1, \dots, u_k, v_1, \dots, v_{k'}\}$ form a $(k+k')$-clique. We can construct such an NFA with $\widetilde{O}(n)$ states and transitions as follows: For any node $u \in V$, it is straightforward to construct an NFA $N(u)$ of size $\widetilde{O}(n)$ accepting only the node IDs of vertices $v \in V$ with $\{u, v\} \in E$. We simply introduce a parallel path from starting to accepting state for every neighbor $v$ of $u$. Since every node has at most $n$ neighbors and each ID is of length $O(\log n)$, the size of $N(u)$ is $\widetilde{O}(n)$. We can now construct $CG(u_1, \dots, u_k)$ by connecting in series the following sequence of NFAs:

$$ \underbrace{(N(u_1) \, \dots \, N(u_k)) \, \dots \, (N(u_1) \, \dots \, N(u_k))}_{k' \text{ times}}. $$

---

14    In the theorem, simply choose $\alpha_N = \alpha_n = \beta$.

Here, by connecting in series, we mean identifying the single accepting state of each NFA in the sequence with the starting state of the subsequent NFA. Note that the constructed NFA accepts $v_1^k v_2^k \ldots v_{k'}^k$, if and only if $\{u_i, v_j\} \in E$ for every $1 \le i \le k$ and $1 \le j \le k'$, yielding the desired property. Furthermore, it has $(k + k') \cdot \widetilde{O}(n) = \widetilde{O}(n)$ states and transitions.

For each constructed clique gadget $CG(u_1, \ldots, u_k)$ with $\{u_1, \ldots, u_k\} \in C(k)$, we construct a copy $CG'(u_1, \ldots, u_k)$. We add transitions from the starting state $s$ of $M$ to the starting state of $CG(u_1, \ldots, u_k)$, labeled 2, and similarly transitions from each $CG'(u_1, \ldots, u_k)$ to the accepting state $t$ of $M$, labeled 2. We also add loop transitions $(q, \sigma, q)$ for each $q \in \{s, t\}$ and $\sigma \in \{0, 1, 2, 3\}$. Finally, for every $\{u_1, \ldots, u_k\}, \{u'_1, \ldots, u'_k\} \in C(k)$ such that $\{u_1, \ldots, u_k, u'_1, \ldots, u'_k\}$ form a $2k$-clique in $G$, we add a transition from the accepting state of $CG(u_1, \ldots, u_k)$ to the starting state of $CG'(u'_1, \ldots, u'_k)$, labeled 3.

By the structure of $M$ and $S$, the only accepting runs have the following form: $M$ reads a prefix of $S$, branches on start of some substring $y = 2v_1^k \ldots, v_{k'}^k 3v_1^k \ldots, v_{k'}^k 2$ with $\{v_1, \ldots, v_{k'}\} \in C(k')$ to some $CG(u_1, \ldots, u_k)$ and some $CG'(u'_1, \ldots, u'_k)$, followed by reading the remaining suffix of $S$ in the accepting state $t$. This run exists if and only if

— $\{v_1, \ldots, v_{k'}, u_1, \ldots, u_k\}$ forms a $(k + k')$-clique (so that the prefix of $y$ can be traversed with $CG(u_1, \ldots, u_k)$),

— $\{u_1, \ldots, u_k, u'_1, \ldots, u'_k\}$ forms a $2k$-clique (so that there is a transition from $CG(u_1, \ldots, u_k)$ to $CG'(u'_1, \ldots, u'_k)$), and

— $\{v_1, \ldots, v_{k'}, u'_1, \ldots, u'_k\}$ forms a $(k + k')$-clique (so that the suffix of $y$ can be traversed with $CG'(u'_1, \ldots, u'_k)$).

This is equivalent to $\{v_1, \ldots, v_{k'}, u_1, \ldots, u_k, u'_1, \ldots, u'_k\}$ forming a $(2k + k')$-clique.

Thus, we have created an NFA Acceptance instance that is equivalent to the given $(2k + k')$-Clique instance. Note that the constructed string $S$ has length $\widetilde{O}(n^{k'})$, that $M$ has $O(n^{k+2})$ states, since it consists of $O(n^k)$ clique gadgets of size $\widetilde{O}(n) = O(n^2)$, and that the instance can be constructed in linear time in its size.

Note that by our choice of $k' = \lfloor \beta k \rfloor \le \beta k$, we have $|S| \le \widetilde{O}(n^{k'}) \le O((n^{k+2})^\beta)$. Thus, by appending additional 2's to $S$ and isolated states to $M$, we can ensure that $|S| = \Theta(q^\beta)$ where $q = \Theta(n^{k+2})$ is the number of states of $M$. By adding transitions between the newly added isolated states we can also ensure that the number of transitions is $\Theta(q^2)$.

Thus, our $O(q^{(2+\beta)(1-\epsilon)})$-time combinatorial algorithm for NFA Acceptance on $q$-state NFAs with $\Theta(q^2)$ transitions and strings of length $\Theta(q^\beta)$ solves any given $k$-Clique instance in time

$O(n^{(k+2)(2+\beta)(1-\epsilon)})$. Note that

$$
\begin{aligned}
(k+2)(2+\beta)(1-\epsilon) &= (2k+\beta k) + (4+2\beta) - (k+2)(2+\beta)\epsilon \\
&\leq (2k+\beta k) + (4+2\beta) - k\epsilon \\
&\leq (2k+k') + (5+2\beta) - k\epsilon \\
&< 2k+k',
\end{aligned}
$$

since $k > (5+2\beta)/\epsilon$. Thus, there exists some $\epsilon' > 0$ such that we can solve $(2k+k')$-Clique in time $O(n^{2k+k'-\epsilon'})$ by a combinatorial algorithm, contradicting the combinatorial $k$-Clique hypothesis.                                                                                    ■

### 7.3   Co-nondeterministic Algorithm

We say that a problem $P_A$ has a $t(n)$-time verifier if $P_A$ can be solved in nondeterministic time $t(n)$ and also its complement problem $\overline{P_A}$ can be solved in nondeterministic time $t(n)$, i.e., $P_A$ is in NTIME$[t(n)] \cap$ coNTIME$[t(n)]$. Under NSETH [17], there exists no deterministic fine-grained reduction from Satisfiability to $P_A$ that would establish a SETH-based lower bound of $t(n)^{1+\delta}$ for any $\delta > 0$.

For the reader's convenience, we give a simple verifier for Colored Walk, and thus NFA Acceptance. This verifier is faster than the one for the generalization to the Viterbi Path problem designed in the arXiv version of [10], and simplifies and extends the verifier one could obtain as a consequence of our reduction in Section 3 combined with [22].

**PROPOSITION 7.5.** *Directed 2-Edge Colored Walk has an $O(n^\omega + \ell n^{\omega-1})$-time verifier.*

As a consequence, we obtain verifiers with the same running time (up to subpolynomial factors) for all equivalent Colored Walk formulations and the NFA Acceptance problem over $\Sigma$ of size $n^{o(1)}$.

Thus, assuming NSETH, there cannot be any deterministic reduction from Satisfiability (or Orthogonal Vectors) that establishes a tight conditional lower bound for NFA Acceptance with $\alpha > \omega - 1$ and $\beta > \omega - \alpha$; this includes in particular the important setting of $\alpha = 2$ and $\beta = 1$. This gives a justification why until now, no tight SETH-based lower bound for dense NFAs could be established.

**PROOF OF PROPOSITION 7.5.** Consider a Directed 2-Edge Colored Walk instance $G = (V, E)$ with edge colors $c : E \to \{1, 2\}$, distinguished nodes $s, t \in V$ and color sequence $c_1, \ldots, c_\ell$. Define $A_c$ as the $n \times n$ matrix with

$$
A_c[u, v] = \begin{cases} 1 & \text{if } (u, v) \in E \text{ and } c(u, v) = c \\ 0 & \text{otherwise.} \end{cases}
$$

Furthermore, define $x_0 \in \{0, 1\}^n$ as the indicator vector representing the starting state $s$, i.e., $x_0[v] = 1$ if and only if $v = s$. For all $1 \le i \le \ell$, we define

$$x_i = A_{c_i}^T \cdot x_{i-1}, \tag{2}$$

where $A^T$ denotes the transpose of $A$ and $A^T \cdot x$ denotes the Boolean matrix-vector product of $A^T$ and $x$. Observe that $x_i$ is the indicator vector with $x_i[v] = 1$ if and only if there exists a walk with color sequence $c_1, \dots, c_i$ from $s$ to $v$.

To obtain nondeterministic algorithms for the problem and its complement problem, we simply need to guess $x_1, \dots, x_\ell$, verify that they have been guessed correctly, and check whether $x_\ell[t] = 1$, or $x_\ell[t] = 0$, respectively.

To verify that the $x_i$'s have been guessed correctly, we batch the equalities (2) that we need to check into two Boolean matrix products: Specifically, for each $c \in \{1, 2\}$, let $X_c$ denote the matrix containing all $x_i$'s with $c_i = c$ as columns, and let $X_c'$ denote the matrix containing all $x_{i-1}$'s with $c_i = c$ as columns, in the same order. For each $c \in \{1, 2\}$, we need to check that

$$X_c = A^T X_c'.$$

Note that this can be done via two (integer) matrix multiplications of two matrices with dimensions $n \times n$ and $n \times \ell$. If $\ell \le n$, this can be done with a single square matrix multiplication, otherwise, we can do this with $\lceil \frac{\ell}{n} \rceil$ square matrix multiplications. This results in a total verification time of $O((1 + \frac{\ell}{n})n^\omega) = O(n^\omega + \ell n^{\omega-1})$, as desired. ∎

## 8.   Failed Algorithmic Approaches for NFA Acceptance

In this section we consider some natural algorithmic approaches to NFA Acceptance and discuss why they do not falsify the NFA Acceptance hypothesis. Instead of directly working with the NFA Acceptance problem, we consider Directed Edge-2-Colored Walk (or short: Colored Walk). This is without loss of generality, as shown by the equivalence established in Lemma 2.3.

**Algorithms Using Fast Matrix Multiplication?**  Note that in the special case of only one color, i.e, $c_1 = \dots = c_\ell$, Colored Walk can be solved by computing the $\ell$-th matrix power $A^\ell$ of the adjacency matrix $A$, and then checking whether the $(s, t)$-entry is non-zero. This takes time $O(n^\omega \log \ell)$, which is much faster than $O(m\ell)$ if $\ell$ is large.

If we could achieve the same running time $O(n^\omega \log \ell)$ in the case of two colors, then we would falsify the NFA Acceptance hypothesis (for any setting $\alpha \in [1, 2]$ and $\beta > \omega - \alpha$, by Lemma 2.3). However, as we argue next, the natural generalization of the above algorithm to two colors fails. Consider adjacency matrices $A^{(1)}, A^{(2)}$, where the $(u, v)$-entry of $A^{(c)}$ is 1 if there is an edge of color $c$ from $u$ to $v$, and 0 otherwise. Then we would want to compute the matrix $A^{(c_1)} A^{(c_2)} \cdots A^{(c_\ell)}$ and check whether its $(s, t)$-entry is non-zero, to solve Directed Edge-2-

Colored Walk. However, in the worst case no two subsequences of $c_1, \ldots, c_\ell$ of length $\log(\ell)$ are equal, and thus we cannot save more than log-factors by precomputing products of the form $A^{(c_i)} \cdots A^{(c_j)}$ – in contrast to the special case of only one color. Hence, we are essentially forced to compute the product $A^{(c_1)} A^{(c_2)} \cdots A^{(c_\ell)}$ one-by-one over $\ell$ steps, resulting in time $O(n^\omega \ell)$. This is worse than the simple $O(m\ell)$-time algorithm. In other words, the natural generalization of the matrix-multiplication-based algorithm for one color fails for two colors.

**Parallel Algorithms for Colored Walk** The matrix multiplication approach sketched above is nevertheless useful to design *parallel* algorithms for Colored Walk. Indeed, matrix multiplication can be easily parallelized, e.g., on the PRAM one can multiply two $n \times n$ matrices with depth $O(\log n)$ and work $O(n^3)$. To compute the product $A^{(c_1)} A^{(c_2)} \cdots A^{(c_\ell)}$ we can recursively compute the product $A^{(c_1)} \cdots A^{(c_{\lfloor \ell/2 \rfloor})}$ and recursively compute the product $A^{(c_{\lfloor \ell/2 \rfloor + 1})} \cdots A^{(c_\ell)}$ and then multiply the results. If we perform both recursive calls in parallel, then we obtain a PRAM algorithm with depth $O(\log(n) \log(\ell))$ and work $O(n^3 \ell)$. From the final matrix product $A^{(c_1)} \cdots A^{(c_\ell)}$ we can then read off the answer as the $(s, t)$-entry. This shows that Colored Walk (and in a similar way also NFA Acceptance) has an efficient parallel algorithm. Note that the total work $O(n^3 \ell)$ of this PRAM algorithm is much more than $O(m\ell)$, and hence this parallel algorithm does not violate the NFA Acceptance hypothesis.

**Inherently Sequential?** The above parallel algorithm has much higher work than the sequential time complexity $O(m\ell)$. It thus seems plausible that any parallel algorithm for Colored Walk with work $\widetilde{O}(m\ell)$ has depth at least $\ell^{1-o(1)}$. In other words, it is plausible that *work-optimal algorithms for Colored Walk are inherently sequential.*

   This intuition is consistent with our current knowledge of the problem. However, there are no tools available to prove this intuition, even conditionally. The only tool that we have to show that certain parallel algorithms are unlikely for a problem $X$ is to show that $X$ is P-hard. However, since we have seen a PRAM algorithm with polylogarithmic depth for Colored Walk, it is very unlikely that Colored Walk is P-hard, and thus this tool is not applicable for this problem. Hence, currently there are no tools to provide evidence for the intuition that work-optimal algorithms for Colored Walk are inherently sequential.

## 9. Conclusion

In this work, we have posed the NFA Acceptance hypothesis, discussed implications of a proof or refutation and considered its connections to standard assumptions in fine-grained complexity theory. At the very least, the NFA Acceptance hypothesis should be understood as a technical challenge towards progress for many interesting problems. Specifically, without refuting it, we cannot expect:

— Context-Free Language Reachability in time $O(n^{3-\epsilon})$,

— the Word Break problem in time $O(m + nm^{1/3-\epsilon})$,

— approximating the Viterbi path in time $O((m\ell)^{1-\epsilon})$,

— pattern matching in string-labeled graphs in time $O((m\ell)^{1-\epsilon})$,

— regular path queries in graph databases in time $O((|q| \cdot m)^{1-\epsilon})$, and

— polynomial improvements for any data structure problems and dynamic problems with tight OMv-hardness, including problems from graph algorithms [35, 27, 4, 36], string algorithms [24, 41, 23], computational geometry [28, 45], linear algebra [39], formal languages [33], and database theory [13, 18, 40].

We leave the following open problems:

— Refute the NFA Acceptance hypothesis or prove that it is implied by any standard fine-grained hypothesis.

— Prove an equivalence of NFA Acceptance and Directed $n^\gamma$-Edge-Colored Walk for any $\gamma > 0$, ideally for all $0 < \gamma \leq 2$.

— Prove an equivalence of NFA Acceptance with and without $\varepsilon$-transitions for densities $m = \Theta(n^\alpha)$ with $1 \leq \alpha < 2$.

# References

[1] Amir Abboud, Arturs Backurs, Karl Bringmann, and Marvin Künnemann. Fine-grained complexity of analyzing compressed data: quantifying improvements over decompress-and-solve. *Proc. 58th IEEE Annual Symposium on Foundations of Computer Science (FOCS 2017)*, pages 192–203. IEEE Computer Society, 2017. DOI (5, 27)

[2] Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. If the current clique algorithms are optimal, so is Valiant's parser. *SIAM J. Comput.* 47(6):2527–2555, 2018. DOI (5, 6, 26)

[3] Amir Abboud and Karl Bringmann. Tighter connections between Formula-SAT and shaving logs. *Proc. 45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107 of *LIPIcs*, 8:1–8:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. DOI (5)

[4] Amir Abboud and Søren Dahlgaard. Popular conjectures as a barrier for dynamic planar graph algorithms. *Proc. 57th Annual Symposium on Foundations of Computer Science (FOCS 2016)*, pages 477–486. IEEE Computer Society, 2016. DOI (9, 15, 32)

[5] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. Time and tape complexity of pushdown automaton languages. *Inf. Control.* 13(3):186–206, 1968. DOI (8)

[6] Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. *TheoretiCS*, 3, 2024. DOI (6)

[7] Amihood Amir, Moshe Lewenstein, and Noa Lewenstein. Pattern matching in hypertext. *J. Algorithms*, 35(1):82–99, 2000. DOI (3)

[8] Marcelo Arenas, Luis Alberto Croquevielle, Rajesh Jayaram, and Cristian Riveros. Counting the answers to a query. *SIGMOD Rec.* 51(3):6–17, 2022. DOI (3)

[9] **Arturs Backurs**, **Nishanth Dikkala**, and **Christos Tzamos**. Tight hardness results for maximum weight rectangles. *Proc. 43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *LIPIcs*, 81:1–81:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. DOI (8)

[10] **Arturs Backurs and Piotr Indyk**. Which regular expression patterns are hard to match? *Proc. 57th Annual Symposium on Foundations of Computer Science (FOCS 2016)*, pages 457–466. IEEE Computer Society, 2016. DOI (5, 6, 9, 25, 29)

[11] **Arturs Backurs and Christos Tzamos**. Improving Viterbi is hard: better runtimes imply faster clique algorithms. *Proc. 34th International Conference on Machine Learning (ICML 2017)*, volume 70 of *Proceedings of Machine Learning Research*, pages 311–321. PMLR, 2017. URL (3–5, 8, 27)

[12] **Pablo Barceló**. Querying graph databases. *Proc. 32nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2013)*, pages 175–188. ACM, 2013. DOI (3)

[13] **Christoph Berkholz**, **Jens Keppeler**, and **Nicole Schweikardt**. Answering conjunctive queries under updates. *Proc. 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS 2017)*, pages 303–318. ACM, 2017. DOI (9, 15, 32)

[14] **Karl Bringmann**, **Allan Grønlund**, **Marvin Künnemann**, and **Kasper Green Larsen**. The NFA acceptance hypothesis: non-combinatorial and dynamic lower bounds. *15th Innovations in Theoretical Computer Science Conference, ITCS 2024, January 30 to February 2, 2024, Berkeley, CA, USA*, volume 287 of *LIPIcs*, 22:1–22:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. DOI (1)

[15] **Karl Bringmann**, **Allan Grønlund**, and **Kasper Green Larsen**. A dichotomy for regular expression membership testing. *Proc. 58th Annual Symposium on Foundations of Computer Science (FOCS 2017)*, pages 307–318. IEEE Computer Society, 2017. DOI (5, 9, 13, 25)

[16] **Karl Bringmann and Marvin Künnemann**. Quadratic conditional lower bounds for string problems and dynamic time warping. *Proc. 56th Annual Symposium on Foundations of Computer Science, (FOCS 2015)*, pages 79–97. IEEE Computer Society, 2015. DOI (25)

[17] **Marco L. Carmosino**, **Jiawei Gao**, **Russell Impagliazzo**, **Ivan Mihajlin**, **Ramamohan Paturi**, and **Stefan Schneider**. Nondeterministic extensions of the Strong Exponential Time Hypothesis and consequences for non-reducibility. *Proc. 2016 ACM Conference on Innovations in Theoretical Computer Science (ITCS 2016)*, pages 261–270. ACM, 2016. DOI (6, 29)

[18] **Katrin Casel and Markus L. Schmid**. Fine-grained complexity of regular path queries. *Log. Methods Comput. Sci.* 19(4), 2023. DOI (3–5, 9, 15, 25, 27, 32)

[19] **Timothy M. Chan**, **Virginia Vassilevska Williams**, and **Yinzhan Xu**. Algorithms, reductions and equivalences for small weight variants of all-pairs shortest paths. *Proc. 48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, volume 198 of *LIPIcs*, 47:1–47:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. DOI (7)

[20] **Krishnendu Chatterjee**, **Bhavya Choudhary**, and **Andreas Pavlogiannis**. Optimal Dyck reachability for data-dependence and alias analysis. *Proc. ACM Program. Lang.* 2(POPL):30:1–30:30, 2018. DOI (7, 8, 12)

[21] **Swarat Chaudhuri**. Subcubic algorithms for recursive state machines. *Proc. 35th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, (POPL 2008)*, pages 159–169. ACM, 2008. DOI (8, 12)

[22] **Dmitry Chistikov**, **Rupak Majumdar**, and **Philipp Schepper**. Subcubic certificates for CFL reachability. *Proc. ACM Program. Lang.* 6(POPL):1–29, 2022. DOI (6, 8, 29)

[23] **Raphaël Clifford**, **Pawel Gawrychowski**, **Tomasz Kociumaka**, **Daniel P. Martin**, and **Przemyslaw Uznanski**. The dynamic k-mismatch problem. *Proc. 33rd Annual Symposium on Combinatorial Pattern Matching (CPM 2022)*, volume 223 of *LIPIcs*, 18:1–18:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. DOI (9, 15, 32)

[24] **Raphaël Clifford**, **Allan Grønlund**, **Kasper Green Larsen**, and **Tatiana Starikovskaya**. Upper and lower bounds for dynamic data structures on strings. *Proc. 35th Symposium on Theoretical Aspects of Computer Science (STACS 2018)*, volume 96 of *LIPIcs*, 22:1–22:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. DOI (9, 15, 32)

[25] **Don Coppersmith and Shmuel Winograd**. Matrix multiplication via arithmetic progressions. *Journal on Symbolic Computation*, 9(3):251–280, 1990. DOI (6)

[26] **Artur Czumaj and Andrzej Lingas**. Finding a heaviest vertex-weighted triangle is not harder than matrix multiplication. *SIAM J. Comput.* 39(2):431–444, 2009. DOI (7)

[27] **Søren Dahlgaard**. On the hardness of partially dynamic graph problems and connections to diameter. *Proc. 43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55 of *LIPIcs*, 48:1–48:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. DOI (9, 15, 32)

[28] **Justin Dallant and John Iacono**. Conditional lower bounds for dynamic geometric measure problems. *Proc. 30th Annual European Symposium on Algorithms (ESA 2022)*, volume 244 of *LIPIcs*, 39:1–39:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. DOI (9, 15, 32)

[29] **Ran Duan**, **Hongxun Wu**, and **Renfei Zhou**. Faster matrix multiplication via asymmetric hashing. *Proc. 64th IEEE Annual Symposium on Foundations of Computer Science (FOCS 2023)*, pages 2129–2138. IEEE, 2023. DOI (6)

[30] **Massimo Equi**, **Veli Mäkinen**, **Alexandru I. Tomescu, and Roberto Grossi**. On the complexity of string matching for graphs. *ACM Trans. Algorithms*, 19(3):21:1–21:25, 2023. DOI (3, 5, 25)

[31] **Pawel Gawrychowski and Przemyslaw Uznanski**. Towards unified approximate pattern matching for Hamming and $L_1$ distance. *Proc. 45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107 of *LIPIcs*, 62:1–62:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. DOI (7)

[32] **Daniel Gibney**, **Gary Hoppenworth, and Sharma V. Thankachan**. Simple reductions from Formula-SAT to pattern matching on labeled graphs and subtree isomorphism. *Proc. 4th Symposium on Simplicity in Algorithms (SOSA 2021)*, pages 232–242. SIAM, 2021. DOI (5)

[33] **Daniel Gibney and Sharma V. Thankachan**. Text indexing for regular expression matching. *Algorithms*, 14(5):133, 2021. DOI (9, 15, 32)

[34] **Nevin Heintze and David A. McAllester**. On the cubic bottleneck in subtyping and flow analysis. *Proc. 12th Annual Symposium on Logic in Computer Science (LICS 1997)*, pages 342–351. IEEE Computer Society, 1997. DOI (8)

[35] **Monika Henzinger**, **Sebastian Krinninger**, **Danupon Nanongkai, and Thatchaphol Saranurak**. Unifying and strengthening hardness for dynamic problems via the online matrix-vector multiplication conjecture. *Proc. 47th Annual Symposium on Theory of Computing (STOC 2015)*, pages 21–30. ACM, 2015. DOI (7, 9, 15, 32)

[36] **Monika Henzinger and Xiaowei Wu**. Upper and lower bounds for fully retroactive graph problems. *Proc. 17th International Symposium on Algorithms and Data Structures (WADS 2021)*, volume 12808 of *Lecture Notes in Computer Science*, pages 471–484. Springer, 2021. DOI (9, 15, 32)

[37] **Jianyu Huang**, **Tyler M. Smith**, **Greg M. Henry, and Robert A. van de Geijn**. Strassen's algorithm reloaded. *Proc. International Conference for High Performance Computing, Networking, Storage and Analysis (SC 2016)*, pages 690–701. IEEE Computer Society, 2016. DOI (7)

[38] **Chirag Jain**, **Haowen Zhang**, **Yu Gao, and Srinivas Aluru**. On the complexity of sequence-to-graph alignment. *J. Comput. Biol.* 27(4):640–654, 2020. DOI (3)

[39] **Shunhua Jiang**, **Binghui Peng, and Omri Weinstein**. The complexity of dynamic least-squares regression. *Proc. 64th Annual Symposium on Foundations of Computer Science (FOCS 2023)*, pages 1605–1627. IEEE, 2023. DOI (9, 15, 32)

[40] **Ahmet Kara**, **Milos Nikolic**, **Dan Olteanu, and Haozhe Zhang**. Conjunctive queries with free access patterns under updates. *Proc. 26th International Conference on Database Theory (ICDT 2023)*, volume 255 of *LIPIcs*, 17:1–17:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. DOI (9, 15, 32)

[41] **Dominik Kempa and Tomasz Kociumaka**. Dynamic suffix array with polylogarithmic queries and updates. *Proc. 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2022)*, pages 1657–1670. ACM, 2022. DOI (9, 15, 32)

[42] **Paraschos Koutris and Shaleen Deep**. The fine-grained complexity of CFL reachability. *Proc. ACM Program. Lang.* 7(POPL):1713–1739, 2023. DOI (8)

[43] **Marvin Künnemann**. A tight (non-combinatorial) conditional lower bound for Klee's measure problem in 3D. *Proc. 63rd Annual Symposium on Foundations of Computer Science (FOCS 2022)*, pages 555–566. IEEE, 2022. DOI (8)

[44] **Kasper Green Larsen and R. Ryan Williams**. Faster online matrix-vector multiplication. *Proc. 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2017)*, pages 2182–2189. SIAM, 2017. DOI (9, 15)

[45] **Joshua Lau and Angus Ritossa**. Algorithms and hardness for multidimensional range updates and queries. *Proc. 12th Innovations in Theoretical Computer Science Conference (ITCS 2021)*, volume 185 of *LIPIcs*, 35:1–35:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. DOI (9, 15, 32)

[46] **François Le Gall**. Powers of tensors and fast matrix multiplication. *Proc. 39th International Symposium on Symbolic and Algebraic Computation (ISSAC'14)*, pages 296–303, 2014. DOI (6)

[47] **Lillian Lee**. Fast context-free grammar parsing requires fast boolean matrix multiplication. *J. ACM*, 49(1):1–15, 2002. DOI (6)

[48] **Andrea Lincoln**, **Adam Polak, and Virginia Vassilevska Williams**. Monochromatic triangles, intermediate matrix products, and convolutions. *Proc. 11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151 of *LIPIcs*, 53:1–53:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. DOI (7)

[49] **Andrea Lincoln**, **Virginia Vassilevska Williams, and R. Ryan Williams**. Tight hardness for shortest cycles and paths in sparse graphs. *Proc. 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2018)*, pages 1236–1252. SIAM, 2018. DOI (8)

[50] **Udi Manber and Sun Wu**. Approximate string matching with arbitrary costs for text and hypertext. *Proc. International Workshop on Advances in Structural and Syntactic Pattern Recognition*, pages 22–33, 1992. DOI (3)

[51] **David Melski and Thomas W. Reps**. Interconvertibility of a class of set constraints and context-free-language reachability. *Theor. Comput. Sci.* 248(1-2):29–98, 2000. DOI (8)

[52] **Alberto O. Mendelzon and Peter T. Wood**. Finding regular simple paths in graph databases. *SIAM J. Comput.* 24(6):1235–1258, 1995. DOI (3)

[53] **Gonzalo Navarro**. Improved approximate pattern matching on hypertext. *Theor. Comput. Sci.* 237(1-2):455–463, 2000. DOI (3)

[54] Radford Neal. The computational complexity of taxonomic inference. Unpublished manuscript, 1989. Unpublished manuscript (8)

[55] Jaroslav Nešetřil and Svatopluk Poljak. On the complexity of the subgraph problem. eng. *Commentationes Mathematicae Universitatis Carolinae*, 026(2):415–419, 1985. URL (5, 26)

[56] Mihai Patrascu. Towards polynomial lower bounds for dynamic problems. *Proc. 42nd ACM Symposium on Theory of Computing (STOC 2010)*, pages 603–610. ACM, 2010. DOI (7)

[57] Aaron Potechin and Jeffrey O. Shallit. Lengths of words accepted by nondeterministic finite automata. *Inf. Process. Lett.* 162:105993, 2020. DOI (5, 6, 25, 27)

[58] Wojciech Rytter. Fast recognition of pushdown automaton and context-free languages. *Inf. Control.* 67(1-3):12–22, 1985. DOI (8)

[59] Philipp Schepper. Fine-grained complexity of regular expression pattern matching and membership. *Proc. 28th Annual European Symposium on Algorithms (ESA 2020)*, volume 173 of *LIPIcs*, 80:1–80:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. DOI (5)

[60] Volker Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13(4):354–356, August 1969. DOI (6)

[61] Leslie G. Valiant. General context-free recognition in less than cubic time. *J. Comput. Syst. Sci.* 10(2):308–315, 1975. DOI (6, 7)

[62] Jan van den Brand, Danupon Nanongkai, and Thatchaphol Saranurak. Dynamic matrix inverse: improved algorithms and matching conditional lower bounds. *Proc. 60th IEEE Annual Symposium on Foundations of Computer Science (FOCS 2019)*, pages 456–480. IEEE Computer Society, 2019. DOI (9)

[63] Virginia Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. *Proc. 44th Annual ACM Symposium on Theory of Computing Conference (STOC'12)*, pages 887–898, 2012. DOI (6)

[64] Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. *Proceedings of the International Congress of Mathematicians*, ICM'18, pages 3447–3487, 2018. (6)

[65] Virginia Vassilevska Williams and R. Ryan Williams. Subcubic equivalences between path, matrix, and triangle problems. *J. ACM*, 65(5):27:1–27:38, 2018. DOI (8)

[66] Virginia Vassilevska Williams and Yinzhan Xu. Monochromatic triangles, triangle listing and APSP. *Proc. 61st IEEE Annual Symposium on Foundations of Computer Science (FOCS 2020)*, pages 786–797. IEEE, 2020. DOI (7)

[67] Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. New bounds for matrix multiplication: from alpha to omega. *Proc. 2024 ACM-SIAM Symposium on Discrete Algorithms (SODA 2024)*, pages 3792–3835. SIAM, 2024. DOI (6)

[68] Andrew J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Inf. Theory*, 13(2):260–269, 1967. DOI (3)

[69] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.* 348(2-3):357–365, 2005. DOI (25)

[70] Mihalis Yannakakis. Graph-theoretic methods in database theory. *Proc. 9th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 1990)*, pages 230–242. ACM Press, 1990. DOI (8, 12)