# The Parameterized Complexity of Intersection and Composition Operations on Sets of Finite-State Automata

H. Todd Wareham

Department of Computer Science, Memorial University of Newfoundland,
St. John's, NF, Canada A1B 3X5
harold@cs.mun.ca

**Abstract.** This paper uses parameterized complexity analysis to delimit possible non-polynomial time algorithmic behaviors for the finite-state acceptor intersection and finite-state transducer intersection and composition problems. One important result derived as part of these analyses is the first proof of the $NP$-hardness of the finite-state transducer composition problem for both general and $p$-subsequential transducers.

## 1 Introduction

Certain applications of finite-state automata are most naturally stated in terms of the intersection or composition of a set of automata [7,10]. One approach to solving these problems is to use state Cartesian product constructions [6, pp. 59-60] to build the automaton associated with the intersection or composition and then answer the query relative to a determinized and/or minimized version of that automaton. Though such queries as emptiness or membership can typically be answered in time and space linear in the size of the derived automaton, the automaton may have $O(|Q|^{|A|})$ states, where $|A|$ is the number of automata in the set and $|Q|$ is the maximum number of states in any automaton in the set. This is to be expected, as many problems on sets of automata are $NP$-hard and hence do not have polynomial-time algorithms unless $P = NP$. However, are there other non-polynomial time algorithmic options for solving such problems, *e.g.*, an algorithm whose non-polynomial time complexity term is purely a function of $|Q|$ and $|\Sigma|$, where $|\Sigma|$ is the size of the language-alphabet? Knowledge of such options would be useful in practice for choosing the most efficient algorithm in situations in which one or more of the characteristics of the problem are of bounded value, *e.g.*, $|Q| \leq 4$ and $|\Sigma| \leq 26$.

In this paper, techniques from the theory of parameterized computational complexity [4] are used to determine part of the range of possible non-polynomial time algorithmic behaviors for the finite-state acceptor intersection and finite-state transducer intersection and composition problems. These analyses generalize and simplify results given in [13]. One important result derived as part of these analyses is the first proof of the $NP$-hardness of the finite-state transducer composition problem for both general and $p$-subsequential transducers.

### 1.1 Terminology

A **finite state acceptor (FSA)** is a 5-tuple $\langle Q, \Sigma, \delta, s, F\rangle$ where $Q$ is a set of states, $\Sigma$ is an alphabet, $\delta : Q \times \{\Sigma \cup \{\epsilon\}\} \times Q$ is a transition relation, $s \in Q$ is the start state, and $F \subseteq Q$ is a set of final states. If $\delta$ has no entries of the form $(q, \epsilon, q')$ and is also a function, *i.e.*, for each $q \in Q$ and $s \in \Sigma$ there is at most one state $q' \in Q$ such that $(q, s, q') \in \delta$, the FSA is a **deterministic finite-state acceptor (DFA)**.

A **finite state transducer (FST)** is a 6-tuple $\langle Q, \Sigma_i, \Sigma_o, \delta, s, F\rangle$ where $Q$ is a set of states, $\Sigma_i$ and $\Sigma_o$ are the input and output alphabets, respectively, $\delta : Q \times \Sigma_i^* \times \Sigma_o^* \times Q$ is a transition relation, $s \in Q$ is the start state, and $F \subseteq Q$ is a set of final states. There are several possible definitions of determinism for FST; types of interest here are:

- *i*-**Deterministic FST (sequential FST [11])**: For each $q \in Q$ and $x \in \Sigma_i^*$, there is at most one $y \in \Sigma_o^*$ and $q' \in Q$ such that $(q, x, y, q') \in \delta$.
- *i/o*-**Deterministic FST**: For each $q \in Q$, $x \in \Sigma_i^*$ and $y \in \Sigma_o^*$, there is at most one $q' \in Q$ such that $(q, x, y, q') \in \delta$.

All FST in this paper are restricted to singleton labels that are $\epsilon$-free, *i.e.*, $\delta : Q \times \Sigma_i \times \Sigma_o \times Q$. Note that such FST will always produce output strings of the same length as the input string, [7, Lemma 3.3].

## 2 Parameterized Complexity Analysis

The theory of $NP$-completeness [5] proposes a class $NP$ of decision problems that is conjectured to properly include the class $P$ of decision problems that have polynomial-time algorithms. For a given decision problem $\Pi$, if every problem in $NP$ reduces[1] to $\Pi$, *i.e.*, $\Pi$ is $NP$-**hard**, then $\Pi$ does not have a polynomial-time algorithm unless $P = NP$.

It may still be possible to solve $NP$-hard problems by invoking non-polynomial time algorithms that are effectively polynomial time because their non-polynomial terms are purely functions of sets of aspects of the problems that are of bounded size or value in instances of those problems encountered in practice, where an **aspect** of a problem is some (usually numerical) characteristic that can be derived from instances of that problem, *i.e.*, $|Q|$, $|\Sigma|$, and $|A|$ in the case of finite-state automaton intersection and composition problems. The theory of parameterized computational complexity [4] provides explicit mechanisms for analyzing the effects of both individual aspects and sets of aspects on problem complexity.

[1] Given decision problems $\Pi$ and $\Pi'$, $\Pi$ **reduces to** $\Pi'$, *i.e.*, $\Pi \leq_m \Pi'$, if there is an algorithm $A$ that transforms an instance $x$ of $\Pi$ into an instance $y$ of $\Pi'$ such that $A$ runs in time polynomial in the size of $x$ and $x$ has a solution if and only if $y$ has a solution, *i.e.*, $x \in \Pi$ if and only if $y \in \Pi'$.

**Definition 1.** *A* **parameterized problem** $\Pi \subseteq \Sigma^* \times \Sigma^*$ *has instances of the form* $(x, y)$*, where* $x$ *is called the* **main part** *and* $y$ *is called the* **parameter**.

**Definition 2.** *A parameterized problem* $\Pi$ *is* **fixed-parameter tractable** *if there exists an algorithm* $A$ *to determine if instance* $(x, y)$ *is in* $\Pi$ *in time* $f(y) \cdot |x|^\alpha$*, where* $f : \Sigma^+ \mapsto \mathcal{N}$ *is an arbitrary function and* $\alpha$ *is a constant independent of* $x$ *and* $y$*.*

Given a decision problem $\Pi$ with a parameter $p$, let $\langle p \rangle$-$\Pi$ denote the parameterized problem associated with $\Pi$ that is based on parameter $p$ and let $\langle p_c \rangle$-$\Pi$ denote the subproblem of $\langle p \rangle$-$\Pi$ in which $p$ has value $c$ for some constant $c \geq 0$. One can establish that a parameterized problem $\Pi$ is not fixed-parameter tractable by using a parametric reduction[2] to show that $\Pi$ is hard for any of the classes of the **W-hierarchy** $= \{FPT, W[1], W[2], \ldots, W[P], XP\}$ except $FPT$, where $FPT$ is the class of fixed-parameter tractable parameterized problems (see [4] for details). These classes are related as follows:

$$FPT \subseteq W[1] \subseteq W[2] \subseteq \cdots \subseteq W[P] \subseteq \cdots \subseteq XP$$

If a parameterized problem is $\mathcal{C}$-hard for any class $\mathcal{C}$ in the $W$-hierarchy above $FPT$ then that problem is not fixed-parameter tractable unless $FPT = \mathcal{C}$.

The following lemmas will be used in the analyses given in the next section.

**Lemma 3. [16, Lemma 2.1.25]** *Given decision problems* $\Pi$ *and* $\Pi'$ *with parameters* $p$ *and* $p'$*, respectively, if* $\Pi \leq_m \Pi'$ *such that* $p' = g(p)$ *for an arbitrary function* $g$*, then* $\langle p \rangle$-$\Pi$ *parametrically reduces to* $\langle p' \rangle$-$\Pi'$*.*

**Lemma 4. [16, Lemma 2.1.35]** *Given a set* $S$ *of aspects of a decision problem* $\Pi$*, if* $\Pi$ *is* $NP$*-hard when the value of every aspect* $s \in S$ *is fixed, then the parameterized problem* $\langle S \rangle$-$\Pi$ *is not in* $XP$ *unless* $P = NP$*.*

## 3 Results

The analyses in this section will focus on the following three problems:

BOUNDED DFA INTERSECTION (BDFAI)
*Instance:* A set $A$ of DFA over an alphabet $\Sigma$ and a positive integer $k$.
*Question:* Is there a string $x \in \Sigma^k$ that is accepted by each DFA in $A$?

$i/o$-DETERMINISTIC FST INTERSECTION (FST-I)
*Instance:* A set $A$ of $i/o$-deterministic FST, all of whose input and output alphabets are $\Sigma_i$ and $\Sigma_o$, respectively, and a string $u \in \Sigma_i^+$.
*Question:* Is there a string $s \in \Sigma_o^{|u|}$ such that the string-pair $u/s$ is accepted by each FST in $A$?

[2] Given parameterized problems $\Pi$ and $\Pi'$, $\Pi$ **parametrically reduces to** $\Pi'$ if there is an algorithm $A$ that transforms an instance $(x, y)$ of $\Pi$ into an instance $(x', y')$ of $\Pi'$ such that $A$ runs in time $f(y)|x|^\alpha$ time for an arbitrary function $f$ and a constant $\alpha$ independent of both $x$ and $y$, $y' = g(y)$ for some arbitrary function $g$, and $(x, y) \in \Pi$ if and only if $(x', y') \in \Pi'$.

$i/o$-DETERMINISTIC FST COMPOSITION (FST-C)
*Instance:* A set $A$ of $i/o$-deterministic FST, all of whose input and output alphabets are $\Sigma$, a composition-order $O$ on these FST, and a string $u \in \Sigma^+$.
*Question:* Is there a sequence of strings $\{s_0, s_1, \ldots, s_{|A|}\}$ with $s_0 = u$ and $s_i \in \Sigma^{|u|}$ for $1 \le i \le |A|$ such that for the ordering $\{a_1, a_2, \ldots, a_{|A|}\}$ of $A$ under $O$ and $1 \le i \le |A|$, $s_{i-1}/s_i$ is accepted by $a_i$?

The version of problem BDFAI in which $x \in \Sigma^*$ is $PSPACE$-complete [9] and problem FST-I is $NP$-hard by a slight modification of the reduction given in [2, Section 5.5.1]. All parameterized complexity analyses given in this section will be done relative to the following aspects: the number of finite-state automata in $A$ ($|A|$), the required length of the result-string ($k$ in the case of BDFAI, $|u|$ in the case of FST-I and FST-C), the maximum number of states of any finite-state automaton in $A$ ($|Q|$), and the size of the alphabet ($|\Sigma|$ in the case of BDFAI and FST-C, $|\Sigma_i|$ *and* $|\Sigma_o|$ in the case of FST-I).

## 3.1 Bounded DFA Intersection

Hardness results will be derived via reductions from the following problems:

LONGEST COMMON SUBSEQUENCE (LCS) [5, Problem SR10]
*Instance:* A set of strings $X = \{x_1, \ldots, x_k\}$ over an alphabet $\Sigma$, an integer $m$.
*Question:* Is there a string $y \in \Sigma^m$ that is a subsequence of $x_i$ for $i = 1, \ldots, k$?

DOMINATING SET [5, Problem GT2]
*Instance:* A graph $G = (V, E)$, an integer $k$.
*Question:* Is there a set of vertices $V' \subseteq V$, $|V'| \le k$, such that each vertex in $V$ is either in $V'$ or adjacent to a vertex in $V'$?

Note that all reductions below are phrased in terms of BDFAI$_R$, the restricted version of BDFAI in which $k \le |Q|$.

**Lemma 5.** LCS $\le_m$ BDFAI$_R$.

*Proof.* Given an instance $\langle X, k, \Sigma, m\rangle$ of LCS, construct the following instance $\langle A', \Sigma', k'\rangle$ of BDFAI$_R$: Let $\Sigma' = \Sigma$ $k' = m$, and $A'$ be created by applying to each string $x \in X$ the algorithm in [1] which produces a DFA on $|x|+1$ states that recognizes all subsequences of a string $x$. Note that in the constructed instance of BDFAI$_R$, $|A'| = k$, $k' = m$, and $|\Sigma'| = |\Sigma|$; moreover, $k' = m < |Q|$. □

**Lemma 6.** DOMINATING SET $\le_m$ BDFAI$_R$.

*Proof.* Given an instance $\langle G = (V, E), k\rangle$ of DOMINATING SET, construct the following instance $\langle A', \Sigma', k'\rangle$ of BDFAI$_R$: Let $\Sigma' = V$ be an alphabet such that each vertex $v \in V$ has a distinct corresponding symbol in $\Sigma'$, and let $k' = k$. For each $v \in V$, let $adj(v)$ be the set of vertices in $V$ that are adjacent to $v$ in $G$ (including $v$ itself) and $nonadj(v) = V - adj(v)$. For each vertex $v \in V$, construct a two-state DFA $A_v = \langle\{q_1, q_2\}, \Sigma', \delta, q_1, \{q_2\}\rangle$ with transition relation

$\delta = \{(q_1, v', q_1) \mid v' \in \mathit{nonadj}(v)\} \cup \{(q_1, v', q_2) \mid v' \in \mathit{adj}(v)\} \cup \{(q_2, v', q_2) \mid v' \in V\}$. Let $A'$ be the set consisting of all DFA $A_v$ corresponding to vertices $v \in V$ plus the $k+1$ state DFA that recognizes all strings in ${\Sigma'}^k$. Note that in the constructed instance of BDFAI$_R$, $|A'| = |\Sigma'| + 1 = |V| + 1$, $k' = k$, and $|Q| = k + 1$; moreover, $k' = k \leq |V|$ and $k' = k < |Q|$. □

**Lemma 7.** BDFAI$_R$ $\leq_m$ BDFAI$_R$ *such that* $|\Sigma| = 2$.

*Proof.* Given an instance $\langle A, \Sigma, k\rangle$ of BDFAI$_R$, construct the following instance $\langle A', \Sigma', k'\rangle$ of BDFAI$_R$: Let $\Sigma' = \{0, 1\}$ and assign each symbol in $\Sigma$ a binary codeword of fixed length $\ell = \lceil \log |\Sigma| \rceil$. For each DFA $a \in A$, create a DFA $a' \in A'$ by adjusting $Q$ and $\delta$ such that each state $q$ and its outgoing transitions in $a$ is replaced with a "decoding tree" on $2^\ell - 1$ states in $a'$ that uses $\ell$ bits to connect $q$ to the appropriate states. Finally, let $k' = k\ell$. Note that in the constructed instance of BDFAI$_R$, $|A'| = |A|$ and $|\Sigma'| = 2$; moreover, as $(|Q| + 1)(|\Sigma| - 1) \leq |Q'|$ and $k \leq |Q|$, $k' = k\ell \leq |Q| \lceil \log |\Sigma| \rceil \leq (|Q| + 1)(|\Sigma| - 1) \leq |Q'|$. □

**Theorem 8.**

1. BDFAI$_R$ *is NP-hard when* $|\Sigma| = 2$.
2. $\langle k, |\Sigma|\rangle$-BDFAI$_R$ *is in FPT.*
3. $\langle |A|, |Q|\rangle$-BDFAI$_R$ *is in FPT.*
4. $\langle |Q|, |\Sigma|\rangle$-BDFAI$_R$ *is in FPT.*
5. $\langle |A|, k\rangle$-BDFAI$_R$ *is* $W[1]$*-hard.*
6. $\langle k, |Q|\rangle$-BDFAI$_R$ *is* $W[2]$*-hard.*
7. $\langle |A|, |\Sigma|_2\rangle$-BDFAI$_R$ *is* $W[t]$*-hard for all* $t \geq 1$.
8. $\langle |\Sigma|\rangle$-BDFAI$_R$ $\not\in XP$ *unless* $P = NP$.

*Proof of (1)*: Follows from the $NP$-hardness of LCS [5, Problem SR10], the reduction in Lemma 5 from LCS to BDFAI$_R$, and the reduction in Lemma 7 from BDFAI$_R$ to BDFAI$_R$ in which $|\Sigma| = 2$.
*Proof of (2)*: Follows from the algorithm that generates all $|\Sigma|^k$ possible $k$-length strings over alphabet $\Sigma$ and checks each string in $O(|A|k)$ time to see whether that string is accepted by each of the DFA in $A$. The algorithm as a whole runs in $O(|\Sigma|^k k|A|)$ time, which is fixed-parameter tractable relative to $k$ and $|\Sigma|$.
*Proof of (3)*: Follows from the algorithm that constructs the intersection DFA of all DFA in $A$ and the $k+1$-state DFA that recognizes all strings in $\Sigma^k$, and then applies depth-first search to the transition diagram for this intersection DFA to determine if any of its final states are reachable from its start state. The intersection DFA can be created in $O(|Q|^{|A|+1}(k+1)|\Sigma|^2) = O(|Q|^{|A|+1} 2k|\Sigma|^2) = O(|Q|^{|A|+1} k|\Sigma|^2)$ time. As the graph $G = (V, E)$ associated with the transition diagram of this DFA has $|V| \leq (k+1)|Q|^{|A|} \leq 2k|Q|^{|A|}$ states and $|A| \leq (k+1)|Q|^{|A|}|\Sigma| \leq 2k|Q|^{|A|}|\Sigma|$ arcs and depth-first search runs in $O(|V| + |A|)$ time, the algorithm as a whole runs in $O(|Q|^{|A|+1}k|\Sigma|^2)$ time, which is fixed-parameter tractable relative to $|A|$ and $|Q|$.
*Proof of (4)*: This result follows from the observation that there are at most $|Q|^{|\Sigma||Q|} \times 2^{|Q|} \leq |Q|^{2|\Sigma||Q|}$ $i/o$-deterministic FST for any choice of $|Q|$ and

$|\Sigma|$. Hence, the number of different FST in any set $A$ is at most $|Q|^{2|\Sigma||Q|+1}$. This suggests the algorithm that removes all redundant FST in $A$ and then performs the algorithm given in part (3) above. The first step involves checking the isomorphism of the transition diagrams of all FST in $A$, where each transition diagram has at most $|Q|$ vertices and at most $|Q|^2|\Sigma|$ edges, which can be done in $O(((|A|(|A|-1)/2)|Q|^{|Q|}|Q|^2|\Sigma|) = O((|A|^2|Q|^{|Q|+2}|\Sigma|)$ time. Hence, the algorithm as a whole runs in $O(|Q|^{(|Q|^{2|\Sigma||Q|})+1}k|\Sigma^2||A|^2)$ time, which is fixed-parameter tractable relative to $|Q|$ and $|\Sigma|$.
*Proof of (5)*: Follows from the $W[1]$-completeness of $\langle k, m\rangle$-LCS [3], the reduction in Lemma 5 from LCS to BDFAI$_R$ in which $|A'| = k$ and $k' = m$, and Lemma 3.
*Proof of (6)*: Follows from the $W[2]$-completeness of $\langle k\rangle$-Dominating set [4], the reduction in Lemma 6 from Dominating set to BDFAI$_R$ in which $k' = k$ and $|Q| = k + 1$, and Lemma 3.
*Proof of (7)*: Follows from the $W[t]$-hardness of $\langle k\rangle$-LCS for $t \geq 1$ [3], the reduction in Lemma 5 from LCS to BDFAI$_R$ in which $|A'| = k$, the reduction in Lemma 7 from BDFAI$_R$ to BDFAI$_R$ in which $|A'| = |A|$ and $|\Sigma'| = 2$, and Lemma 3.
*Proof of (8)*: Follow from (1) and Lemma 4. □

As BDFAI$_R$ is a restriction of BDFAI, all hardness results above also hold for BDFAI. However, as the value of $k$ is not necessarily bounded by a polynomial in the instance size in BDFAI, the algorithm for part (4) only works (and hence results (4) and (5) only hold) for BDFAI if $k$ is also included in the parameter.

### 3.2 *i/o*-Deterministic FST Intersection

**Lemma 9.** BDFAI$_R \leq_m$ FST-I.

*Proof.* Given an instance $\langle A, \Sigma, k\rangle$ of BDFAI$_R$, construct the following instance $\langle A', \Sigma'_i, \Sigma'_o, u'\rangle$ of FST-I: Let $\Sigma'_i = \{\Delta\}$ for some symbol $\Delta \not\in \Sigma$, $\Sigma'_o = \Sigma$ and $u' = \Delta^k$. Given a DFA $a = \langle Q, \Sigma, \delta, s, F\rangle$, let $FST_u(a) = \langle Q, \Sigma'_i, \Sigma, \delta_F, s, F\rangle$ be the FST such that $\delta_F = \{(q, \Delta, x, q') \mid (q, x, q') \in \delta\}$ and let $A'$ be the set consisting of all FST $FST_u(a)$ corresponding to DFA $a \in A$. Note that in the constructed instance of FST-I, $|A'| = |A|$, $|u'| = k$, $|Q'| = |Q|$, $|\Sigma'_u| = 1$, and $|\Sigma'_s| = |\Sigma|$. □

**Theorem 10.**
1. FST-I *is NP-hard when* $|\Sigma_i| = 1$ *and* $|\Sigma_o| = 2$ *and when* $|Q| = 4$ *and* $|\Sigma_o| = 3$.
2. $\langle|u|, |\Sigma_o|\rangle$-FST-I, $\langle|A|, |Q|\rangle$-FST-I, *and* $\langle|Q|, |\Sigma|\rangle$-FST-I *are in FPT.*
3. $\langle|A|, |u|, |\Sigma_i|_1\rangle$-FST-I *is* $W[1]$*-hard.*
4. $\langle|u|, |Q|, |\Sigma_i|_1\rangle$-FST-I *is* $W[2]$*-hard.*
5. $\langle|A|, |\Sigma_i|_1, |\Sigma_o|_2\rangle$-FST-I *is* $W[t]$*-hard for all* $t \geq 1$.
6. $\langle|\Sigma_o|, |\Sigma_i|\rangle$-FST-I *and* $\langle|Q|, |\Sigma_o|\rangle$-FST-I *are not in XP unless P = NP.*

*Proof.* (*Sketch*) Almost all results follow by arguments similar to those in Theorem 8 relative to the reduction in Lemma 9 and the appropriate results in Theorem 8. The $NP$-hardness of FST-I when $|Q| = 4$ and $|\Sigma_o| = 3$ follows from the reduction in [2, Section 5.5.3]. □

### 3.3 *i*/*o*-Deterministic FST Composition

The following reduction formalizes the observation (made independently by Karttunen [8]) that FSA intersection can be simulated by the composition of identity-relation FST.

**Lemma 11.** BDFAI$_R \leq_m$ FST-C.

*Proof.* Given an instance $\langle A, \Sigma, k\rangle$ of BDFAI$_R$, construct the following instance $\langle A', O', \Sigma', u', \rangle$ of FST-C: Let $\Sigma' = \Sigma \cup \{\Delta\}$ for some symbol $\Delta \notin \Sigma$, and $u' = \Delta^k$. Given a DFA $a = \langle Q, \Sigma, \delta, s, F\rangle$, let $FST_u(a) = \langle Q, \Sigma, \Sigma, \delta_F, s, F\rangle$ be the FST such that $\delta_F = \{(q, x, x, q') \mid (q, x, q') \in \delta\}$. Let $A'$ be the set consisting of all FST $FST_u(a)$ corresponding to DFA $a \in A$ plus the special FST $FST_{init} = \langle \{q_1\}, \{\Delta\}, \Sigma, \delta, q_1, \{q_1\}\rangle$ for which $\delta = \{(q_1, \Delta, x, q') \mid x \in \Sigma\}$, and let $O'$ be an ordering on $A'$ such that $FST_{init}$ is the first FST in $O$ and the other FST appear in an arbitrary order. Note that in the constructed instance of FST-C, $|A'| = |A|+1$, $|u'| = k$, $|Q'| = \max(|Q|, 1) = |Q|$, and $|\Sigma'| = |\Sigma|+1$. □

**Theorem 12.**
1. FST-C *is NP-hard when* $|\Sigma| = 3$.
2. $\langle|u|, |\Sigma|\rangle$-FST-C *and* $\langle|A|, |Q|\rangle$-FST-C *are in FPT.*
3. $\langle|A|, |u|\rangle$-FST-C *is* $W[1]$*-hard.*
4. $\langle|u|, |Q|\rangle$-FST-C *is* $W[2]$*-hard.*
5. $\langle|A|, |\Sigma|_3\rangle$-FST-C *is* $W[t]$*-hard for all* $t \geq 1$.
6. $\langle|\Sigma|\rangle$-FST-C *is not in XP unless P = NP.*

*Proof.* (*Sketch*) Almost all results follow by arguments similar to those in Theorem 8 relative to the reduction in Lemma 11 and the appropriate results in Theorem 8. The fixed-parameter tractability of $\langle|u|, |\Sigma|\rangle$-FST-C follows by a variant of an algorithm in [13, Theorem 4.3.3, Part (3)] that uses an $|\Sigma|^{|u|}$-length bit-vector to store the intermediate sets of strings produced during the FST composition. □

## 4 Discussion

All parameterized complexity results for problems BDFAI, FST-I, and FST-C that are either stated or implicit in the previous section are shown in Tables 1 and 2. Recall that results for problems FST-I and FST-C are stated relative to restricted FST; hence, only hardness results necessarily hold for these problems relative to general FST, and given *FPT* algorithms are at best outlines for possible *FPT* algorithms for general FST (see [13, Sections 4.3.3 and 4.4.3] for further discussion). Future research should both look for algorithms that exploit the sets of aspects underlying the state Cartesian product ($|A|$, $|Q|$) and exhaustive string generation ($|u|$, $|\Sigma|$) constructions) in new ways and consider other aspects of finite-state automaton intersection and composition problems, such as characterizations of logic formulas describing the automata [12].

**Table 1.** The Parameterized Complexity of the BOUNDED DFA INTERSECTION and $i/o$-DETERMINISTIC FST COMPOSITION Problems. (a) The BOUNDED DFA INTERSECTION Problem. (b) The $i/o$-DETERMINISTIC FST COMPOSITION Problem.

(a)

| Parameter | Alphabet Size $\|\Sigma\|$ | |
|---|---|---|
| | Unbounded | Parameter |
| – | $NP$-hard | $\not\in XP$ unless $P = NP$ |
| $\|A\|$ | $W[t]$-hard | $W[t]$-hard |
| $k$ | $W[2]$-hard | $FPT$ |
| $\|Q\|$ | $W[2]$-hard | ??? |
| $\|A\|$, $k$ | $W[1]$-hard | $FPT$ |
| $\|A\|$, $\|Q\|$ | ??? | ??? |
| $k$, $\|Q\|$ | $W[2]$-hard | $FPT$ |
| $\|A\|$, $k$, $\|Q\|$ | $FPT$ | $FPT$ |

(b)

| Parameter | Alphabet Size $\|\Sigma\|$ | |
|---|---|---|
| | Unbounded | Parameter |
| – | $NP$-hard | $\not\in XP$ unless $P = NP$ |
| $\|A\|$ | $W[t]$-hard | $W[t]$-hard |
| $\|u\|$ | $W[2]$-hard | $FPT$ |
| $\|Q\|$ | $W[2]$-hard | ??? |
| $\|A\|$, $\|u\|$ | $W[1]$-hard | $FPT$ |
| $\|A\|$, $\|Q\|$ | $FPT$ | $FPT$ |
| $\|u\|$, $\|Q\|$ | $W[2]$-hard | $FPT$ |
| $\|A\|$, $\|u\|$ $\|Q\|$ | $FPT$ | $FPT$ |

**Table 2.** The Parameterized Complexity of the $i/o$-DETERMINISTIC FST INTERSECTION Problem.

| Parameter | Alphabet Sizes ($\|\Sigma_i\|$,$\|\Sigma_o\|$) | | | |
|---|---|---|---|---|
| | (Unb,Unb) | (Unb,Prm) | (Prm,Unb) | (Prm,Prm) |
| – | $NP$-hard | $\not\in XP$ unless $P = NP$ | $\not\in XP$ unless $P = NP$ | $\not\in XP$ unless $P = NP$ |
| $\|A\|$ | $W[t]$-hard | $W[t]$-hard | $W[t]$-hard | $W[t]$-hard |
| $\|u\|$ | $W[2]$-hard | $FPT$ | $W[2]$-hard | $FPT$ |
| $\|Q\|$ | $W[2]$-hard | $\not\in XP$ unless $P = NP$ | $W[2]$-hard | $FPT$ |
| $\|A\|$, $\|u\|$ | $W[1]$-hard | $FPT$ | $W[1]$-hard | $FPT$ |
| $\|A\|$, $\|Q\|$ | $FPT$ | $FPT$ | $FPT$ | $FPT$ |
| $\|u\|$, $\|Q\|$ | $W[2]$-hard | $FPT$ | $W[2]$-hard | $FPT$ |
| $\|A\|$, $\|u\|$, $\|Q\|$ | $FPT$ | $FPT$ | $FPT$ | $FPT$ |

One such aspect of great interest is FST ambiguity (essentially, the maximum number of strings associated with any input string by a FST). Problems FST-I and FST-C are solvable in low-order polynomial time when they are restricted to operate on sequential FST, *i.e.*, FST that associate at most one output string with any input string. The reduction in Lemma 11 suggests that the presence of only one $i/o$-deterministic FST can make FST composition $NP$-hard. What about more restricted classes of FST? One candidate is the **$p$-subsequential FST** [11], which are essentially sequential FST which are allowed to append any one of a fixed set of $p$ strings to their output. Such FST seem adequate for efficiently representing the ambiguity present in many applications [11]. However, the following result suggests that this observed efficiency is not universal.

**Theorem 13.** 2-SUBSEQUENTIAL FST COMPOSITION *is $NP$-hard.*

*Proof.* (*Sketch*) Given instances of BDFAI$_R$ created in Lemma 7, the reduction in Lemma 11 can be modified by setting $u' = \Delta$ and replacing the $i/o$-deterministic FST $FST_{init}$ with a set of $|u|$ 2-sequential FST that echo their input and append a 0 or a 1 (that is, a set of 2-subsequential FST whose composition generates all possible strings of length $|u|$ over the alphabet $\{0, 1\}$). □

**Acknowledgments.** The author would like to thank the CIAA referees for various helpful suggestions, and for pointing out several errors in the original manuscript as well as solutions for some of these errors.

## References

1. R.A. Baeza-Yates. 1991. Searching Subsequences. *Theoretical Computer Science*, 78, 363–376.
2. G.E. Barton, R.C. Berwick, and E.S. Ristad. 1987. *Computational Complexity and Natural Language.* MIT Press, Cambridge, MA.
3. H.L. Bodlaender, R.G. Downey, M.R. Fellows, and H.T. Wareham. 1995. The Parameterized Complexity of Sequence Alignment and Consensus. *Theoretical Computer Science*, 147(1–2), 31–54.
4. R.G. Downey and M.R. Fellows. 1999. *Parameterized Complexity.* Springer-Verlag, Berlin.
5. M.R. Garey and D.S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, San Francisco.
6. J.E. Hopcroft and J.D. Ullman. 1979. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, Reading, MA.
7. R.M. Kaplan and M. Kay. 1994. Regular Models of Phonological Rule Systems. *Computational Linguistics*, 20(3), 331–378.
8. L. Karttunen. 1998. The Proper Treatment of Optimality in Computational Phonology. Technical Report ROA-258-0498, Rutgers Optimality Archive.
9. D. Kozen. 1977. Lower Bounds for Natural Proof Systems. In *18th IEEE Symposium on Foundations of Computer Science*, pp. 254-266. IEEE Press.
10. R.P. Kurshan. 1994. *Computer-Aided Verification of Coordinating Processes: The Automata-Theoretic Approach.* Princeton University Press.
11. M. Mohri. 1997. On the Use of Sequential Transducers in Natural Language Processing. In E. Roche and Y. Schabes, eds., *Finite-State Language Processing*, pp. 355–382. MIT Press, Cambridge, MA.
12. H. Straubing. 1994. *Finite Automata, Formal Logic, and Circuit Complexity.* Birkhäuser, Boston, MA.
13. H.T. Wareham. 1999. *Systematic Parameterized Complexity Analysis in Computational Phonology.* Ph.D. thesis, Department of Computer Science, University of Victoria. Technical Report ROA-318-0599, Rutgers Optimality Archive.