State Complexity of Pattern Matching in Regular Languages ⋆,⋆⋆

Janusz A. Brzozowski[a], Sylvie Davies[b,*], Abhishek Madan[a]

[a] *David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada N2L 3G1*
[b] *Department of Pure Mathematics, University of Waterloo, Waterloo, ON, Canada N2L 3G1*

Abstract

In a simple pattern matching problem one has a pattern w and a text t, which are words over a finite alphabet Σ. One may ask whether w occurs in t, and if so, where? More generally, we may have a set P of patterns and a set T of texts, where P and T are regular languages. We are interested whether any word of T begins with a word of P, ends with a word of P, has a word of P as a factor, or has a word of P as a subsequence. Thus we are interested in the languages $(P\Sigma^*) \cap T$, $(\Sigma^* P) \cap T$, $(\Sigma^* P \Sigma^*) \cap T$, and $(\Sigma^* \sqcup P) \cap T$, where $\sqcup$ is the shuffle operation. The state complexity $\kappa(L)$ of a regular language L is the number of states in the minimal deterministic finite automaton recognizing L. We derive the following upper bounds on the state complexities of our pattern-matching languages, where $\kappa(P) \leqslant m$, and $\kappa(T) \leqslant n$: $\kappa((P\Sigma^*) \cap T) \leqslant mn$; $\kappa((\Sigma^* P) \cap T) \leqslant 2^{m-1}n$; $\kappa((\Sigma^* P \Sigma^*) \cap T) \leqslant (2^{m-2}+1)n$; and $\kappa((\Sigma^* \sqcup P) \cap T) \leqslant (2^{m-2}+1)n$. We prove that these bounds are tight, and that to meet them, the alphabet must have at least two letters in the first three cases, and at least $m-1$ letters in the last case.

Keywords: all-sided ideal, combined operation, factor, finite automaton, left ideal, pattern matching, prefix, regular language, right ideal, state complexity, subsequence, suffix, two-sided ideal

⋆This work was supported by the Natural Sciences and Engineering Research Council of Canada grant No. OGP0000871.

 Publication available at https://doi.org/10.1016/j.tcs.2018.12.014

⋆Corresponding author

Email addresses: brzozo@uwaterloo.ca (Janusz A. Brzozowski), sldavies@uwaterloo.ca (Sylvie Davies), a7madan@edu.uwaterloo.ca (Abhishek Madan)

1. Introduction

Given a regularity-preserving operation on regular languages, we may ask the following natural question: in the worst case, how many states are necessary and sufficient for a deterministic finite automaton (DFA) to accept the language resulting from the operation, in terms of the number of states of the input DFAs? For example, consider the intersection of two languages: if the input DFAs have m and n states respectively, then an mn-state DFA is sufficient to accept the intersection; this follows by the usual direct product construction. It was proved by Yu, Zhuang, and Salomaa [11] that an mn-state DFA is also necessary in the worst case; for all $m, n \geqslant 1$, there exist a language accepted by an m-state DFA and a language accepted by an n-state DFA whose intersection is accepted by a minimal DFA with mn states.

This worst-case value is called the *state complexity* [7, 8, 11] of the operation. The *state complexity* of a regular language L, denoted by $\kappa(L)$, is the number of states in the minimal DFA accepting L. Thus $\kappa(L) = n$ means the minimal DFA for L has exactly n states, and $\kappa(L) \leqslant n$ means L can be recognized by an n-state DFA. If a language has state complexity n, we indicate this by the subscript n and use L_n instead of L. Then the state complexity of an operation is the worst-case state complexity of the result of the operation, expressed in terms of the maximal allowed state complexity of the inputs. For example, the state complexity of intersection is mn because if $\kappa(K) \leqslant m$ and $\kappa(L) \leqslant n$, then $\kappa(K \cap L) \leqslant mn$ and this bound is tight for all $m, n \geqslant 1$.

Aside from "basic" operations like union, intersection, concatenation and star, the state complexity of *combined operations* [10] such as "star of intersection" and "star of union" has also been studied. We investigate the state complexity of new combined operations inspired by pattern matching problems.

For a comprehensive treatment of pattern matching, see [4]. In a pattern matching problem we have a *text* and a *pattern*. In its simplest form, the pattern w and the text t are both words over an alphabet Σ. Some natural questions about patterns in texts include the following: Does w occur in t, and if so, where?

Pattern matching has many applications. Aho and Corasick [1] developed an algorithm to determine all occurrences of words from a finite pattern in a given text; this algorithm leads to significant improvements in the speed of bibliographic searches. Pattern matching is used in bioinformatics [6]; in this context the text t is often a DNA sequence, and the pattern w is a sequence of nucleotides searched for in the text.

More generally, we can have a *set* P of patterns and a *set* T of texts. These could be finite sets, or they could be arbitrary regular languages, specified by a finite automaton or a regular expression. For example, many text editors and text processing utilities have a *regular expression search* feature, which finds all lines in a text file that match a certain regular expression. In this context, the pattern set P is often a regular language (but not always, as software implementations of "regular expressions" typically have extra features allowing irregular languages to be specified). We can view a text file as either an ordered

sequence of single-word texts t (each representing a line of the file), or if the order of lines is not important, as a finite set T. There could also be cases where it is useful to allow T to be an arbitrary regular language rather than a finite set; for example, T could be the set of all possible interleaved execution traces from the processes in a distributed system, as described in [5].

In this paper, we ask whether a pattern from the set P occurs as a *prefix*, *suffix*, *factor* or *subsequence* of a text from the set T. If $u, v, w \in \Sigma^*$ and $w = uv$, then u is a *prefix* of w and v is a *suffix* of w. If $w = xvy$ for some $v, x, y \in \Sigma^*$, then v is a *factor* of w. If $w = w_0 a_1 w_1 \cdots a_n w_n$, where $a_1, \ldots, a_n \in \Sigma$, and $w_0, \ldots, w_n \in \Sigma^*$, then $v = a_1 \cdots a_n$ is a *subsequence* of w.

If L is any language, then $L\Sigma^*$ is the *right ideal* generated by L, $\Sigma^* L$ is the *left ideal* generated by L, and $\Sigma^* L \Sigma^*$ is the *two-sided ideal* generated by L.

The *shuffle* $u \sqcup v$ of words $u, v \in \Sigma^*$ is defined as follows:

$$u \sqcup v = \{u_1 v_1 \cdots u_k v_k \mid u = u_1 \cdots u_k, v = v_1 \cdots v_k, u_1, \ldots, u_k, v_1, \ldots, v_k \in \Sigma^*\}.$$

The shuffle of two languages K and L over Σ is defined by

$$K \sqcup L = \bigcup_{u \in K, v \in L} u \sqcup v.$$

The language $\Sigma^* \sqcup L$ is an *all-sided ideal*. The language $\Sigma^* \sqcup w$ consists of all words that contain w as a subsequence. Such a language could be used, for example, to determine whether a report has all the required sections and that they are in the correct order.

The combined operations we consider are of the form "the intersection of T with the right (left, two-sided, all-sided) ideal generated by P". We study four problems with pattern sets $P \subseteq \Sigma^*$ and text sets $T \subseteq \Sigma^*$.

1. Find $(P\Sigma^*) \cap T$, the set of all the words in T each of which begins with a word in P.
2. Find $(\Sigma^* P) \cap T$, the set of all the words in T each of which ends with a word in P.
3. Find $(\Sigma^* P \Sigma^*) \cap T$ the set of all the words in T each of which has a word in P as a factor.
4. Find $(\Sigma^* \sqcup P) \cap T$, the set of all the words in T each of which has a word of P as a subsequence.

We find tight upper bounds on the state complexity of each of these four operations. For languages P and T such that $\kappa(P) \leqslant m$ and $\kappa(T) \leqslant n$, we derive the following bounds:

1. Prefix: $\kappa((P\Sigma^*) \cap T) \leqslant mn$.
2. Suffix: $\kappa((\Sigma^* P) \cap T) \leqslant 2^{m-1} n$.
3. Factor: $\kappa((\Sigma^* P \Sigma^*) \cap T) \leqslant (2^{m-2} + 1)n$.
4. Subsequence: $\kappa((\Sigma^* \sqcup P) \cap T) \leqslant (2^{m-2} + 1)n$.

In the prefix, suffix and factor cases, there exist binary witnesses meeting the bounds. For the subsequence case, an alphabet of at least $m-1$ letters is needed to reach the bound.

2. Terminology and Notation

A *deterministic finite automaton (DFA)* is a 5-tuple $\mathcal{D} = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite non-empty set of *states*, Σ is a finite non-empty *alphabet*, $\delta\colon Q \times \Sigma \to Q$ is the *transition function*, $q_0 \in Q$ is the *initial* state, and $F \subseteq Q$ is the set of *final* states. We extend δ to functions $\delta\colon Q \times \Sigma^* \to Q$ and $\delta\colon 2^Q \times \Sigma^* \to 2^Q$ as usual.

A DFA $\mathcal{D}$ *accepts* a word $w \in \Sigma^*$ if $\delta(q_0, w) \in F$. The language accepted by $\mathcal{D}$ is the set of all words that $\mathcal{D}$ accepts, and is denoted by $L(\mathcal{D})$. If q is a state of $\mathcal{D}$, then the language $L_q(\mathcal{D})$ of q is the language accepted by the DFA $(Q, \Sigma, \delta, q, F)$. A state is *empty* (or *dead* or a *sink state*) if its language is empty. Two states p and q of $\mathcal{D}$ are *indistinguishable* if $L_p(\mathcal{D}) = L_q(\mathcal{D})$. A state q is *reachable* if there exists $w \in \Sigma^*$ such that $\delta(q_0, w) = q$. A DFA $\mathcal{D}$ is *minimal* if it has the smallest number of states and the smallest alphabet among all DFAs accepting $L(\mathcal{D})$. It is well known that a DFA is minimal if it uses the smallest alphabet, all of its states are reachable, and no two states are indistinguishable.

A *nondeterministic finite automaton (NFA)* is a 5-tuple $\mathcal{N} = (Q, \Sigma, \delta, q_0, F)$, where δ is now a function $\delta\colon Q \times \Sigma \to 2^Q$, and all other components are as in a DFA. Extending δ to a function $\delta\colon 2^Q \times \Sigma^* \to 2^Q$, the NFA $\mathcal{N}$ accepts a word $w \in \Sigma^*$ if $\delta(\{q_0\}, w) \cap F \neq \emptyset$. As with DFAs, the language accepted by the NFA $\mathcal{N}$ is the set of all accepted words.

A *transformation* of a set Q is a function $t\colon Q \to Q$. The *image* of $q \in Q$ under the transformation t is denoted by qt. If s, t are transformations of Q, their composition is denoted by st and defined by $q(st) = (qs)t$; that is, composition is performed from *left to right*. The *preimage* of $q \in Q$ under the transformation t is denoted by qt^{-1}, and is defined to be the set $qt^{-1} = \{p \in Q \mid pt = q\}$. This notation extends to sets: for $S \subseteq Q$, we have $St = \{qt \mid q \in S\}$ and $St^{-1} = \{p \in Q \mid pt \in S\}$.

For $k \geqslant 2$, a transformation t of a set $P = \{q_0, q_1, \ldots, q_{k-1}\} \subseteq Q$ is a *k-cycle* if $q_0 t = q_1$, $q_1 t = q_2$, $\ldots$, $q_{k-2} t = q_{k-1}$, $q_{k-1} t = q_0$, and $qt = q$ for all $q \in Q \setminus P$. This k-cycle is denoted by $(q_0, q_1, \ldots, q_{k-1})$. A 2-cycle (q_0, q_1) is a *transposition*. The identity transformation is denoted by $\mathbb{1}$. If $Q = \{0, 1, \ldots, n-1\}$ for some n, we write $\binom{j}{i}\ q \to q+1)$ for a transformation that sends q to $q+1$ for $i \leqslant q \leqslant j$ and fixes all other elements of Q, and $\binom{j}{i}\ q \to q-1)$ is defined similarly.

In a DFA $\mathcal{D} = (Q, \Sigma, \delta, q_0, F)$, each letter $a \in \Sigma$ induces a transformation of the set of states Q, defined by $q \mapsto \delta(q, a)$ for $q \in Q$. We denote this transformation simply by the corresponding letter a. Thus instead of writing $\delta(p, a) = q$, we can simply write $pa = q$. Specifying the transformation $a\colon Q \to Q$ induced by each letter $a \in \Sigma$ completely specifies the transition function δ, so we often define δ in this way. This notation extends to words: $w\colon Q \to Q$ denotes the transformation $q \mapsto \delta(q, w)$. It should always be clear from context whether we are talking about the word $w \in \Sigma^*$ or the function $w\colon Q \to Q$.

Henceforth we often refer to state complexity as simply *complexity*, since we do not discuss other measures of complexity in this paper.

3. Prefix Matching

Let P and T be regular languages over an alphabet Σ. We compute the set L of all the words of T that are prefixed by words in P; that is, the language $P\Sigma^* \cap T$. We want to find the worst-case complexity of this language.

Theorem 1. *For $m, n \geqslant 1$, if $\kappa(P) \leqslant m$ and $\kappa(T) \leqslant n$, then $\kappa(P\Sigma^* \cap T) \leqslant mn$, and this bound is tight if the cardinality of Σ is at least 2.*

Proof. The language $P\Sigma^*$ is the right ideal generated by P. It is known that the complexity of $P\Sigma^*$ is at most m [3]. Furthermore, it was shown in [11] that the complexity of intersection of an m-state DFA language with an n-state DFA language is at most mn. Hence mn is an upper bound on the complexity of $(P\Sigma^*) \cap T$.

Now, for all $m, n \geqslant 1$, we find witnesses P_m and T_n with $\kappa(P_m\Sigma^* \cap T_n) = mn$. Define T_n to be the language of $\mathcal{D}_n = (Q_n, \Sigma, \delta_T, 0, \{n-1\})$, where $Q_n = \{0, 1, \ldots, n-1\}$, $\Sigma = \{a, b\}$ and δ_T is defined by the transformations $a = (0, 1, \ldots, n-1)$ and $b = \mathbb{1}$. See Figure 1. This DFA is minimal because the word a^{n-1-q} distinguishes q from every other state.

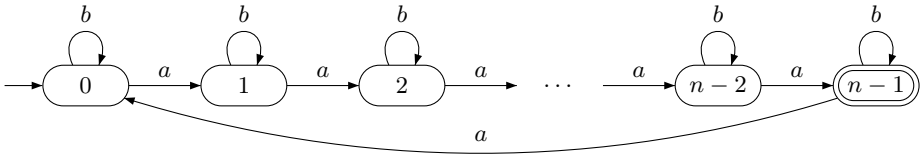

Figure 1: Minimal DFA $\mathcal{D}_n$ of T_n.

Define P_m to be the language of $\mathcal{D}_m^{\leftrightarrow} = (Q_m, \Sigma, \delta_P, 0, \{m-1\})$, where $Q_m = \{0, 1, \ldots, m-1\}$ and δ_P is defined by $a = \mathbb{1}$ and $b = (0, 1, \ldots, m-1)$. Notice that $\mathcal{D}_m^{\leftrightarrow}$ is simply the DFA $\mathcal{D}_m$ with the roles of a and b switched (hence the $\leftrightarrow$ symbol). It follows that this DFA is also minimal.

To find $P_m\Sigma^*$, we concatenate the language P_m with the language Σ^*. Let $(\mathcal{D}_m^{\leftrightarrow})'$ denote the DFA for the concatenation $P_m\Sigma^*$. Note that once state $m-1$ is reached in $(\mathcal{D}_m^{\leftrightarrow})'$, every word is accepted. Thus the transition from $m-1$ to 0 is not needed, because it is replaced by a self-loop on state $m-1$ under b. Thus we obtain the DFA $(\mathcal{D}_m^{\leftrightarrow})' = (Q_m, \Sigma, \delta'_P, 0, \{m-1\})$ of Figure 2, where δ'_P is defined by $a = \mathbb{1}$ and $b = \binom{m-2}{0} q \to q+1)$.

To complete the proof, we find a minimal DFA $\mathcal{D}_L$ for $L = P_m\Sigma^* \cap T_n$. To compute the intersection we take a direct product: $\mathcal{D}_L = (\mathcal{D}_m^{\leftrightarrow})' \times \mathcal{D}_n$. An example of this product for $m = n = 4$ is given in Figure 3. Let $\mathcal{D}_L = (Q_m \times Q_n, \Sigma, \delta_L, (0,0), \{(m-1, n-1)\})$, where the transformation induced by $a \in \Sigma$ is defined by $(p, q)a = (pa, qa)$. (Less concisely, we have $\delta_L((p, q), a) = (\delta'_P(p, a), \delta_T(q, a))$.) Since DFA $\mathcal{D}_L$ has mn states, it remains to prove that it is minimal.

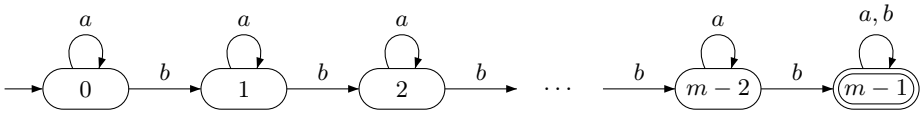

Figure 2: Minimal DFA $(\mathcal{D}_m^{\leftrightarrow})'$ of $P_m\Sigma^*$.

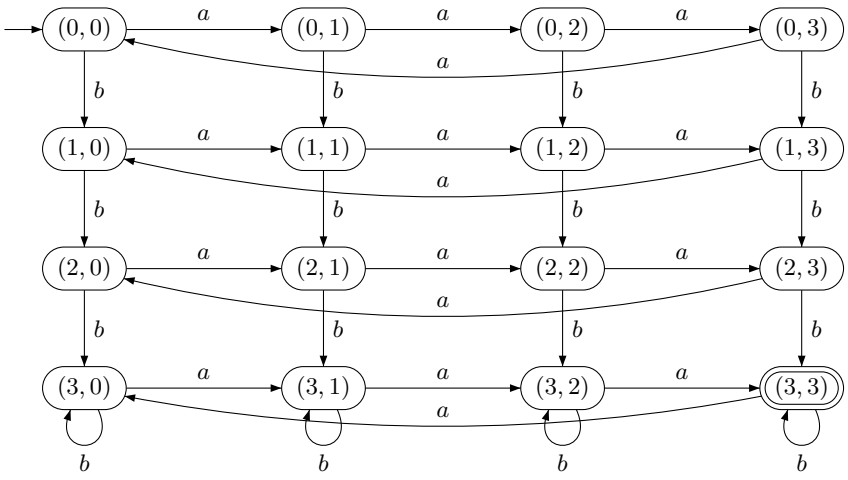

Figure 3: The direct product of $(\mathcal{D}_4^{\leftrightarrow})'$ and $\mathcal{D}_4$ for intersection.

We observe that $(0,0)a^qb^p = (p,q)$, for all $0 \leqslant p \leqslant m-1$ and $0 \leqslant q \leqslant n-1$. Therefore, every state is reachable. We also observe that the minimal word in a^*b^* accepted by a state (p,q) is $a^{n-1-q}b^{m-1-p}$, where $0 \leqslant p \leqslant m-1$ and $0 \leqslant q \leqslant n-1$. Therefore, each state in $Q_m \times Q_n$ has a unique minimal word in a^*b^*; this makes all states pairwise distinguishable. Hence, $\mathcal{D}_L$ is minimal, and $L = P_m\Sigma^* \cap T_n$ has state complexity mn. □

4. Suffix Matching

We are now interested in the worst-case state complexity of the set of all the words of T that end with words in P, that is, the set $\Sigma^*P \cap T$.

Proposition 2. *For $m, n \geqslant 2$, if $\kappa(P) \leqslant m$ and $\kappa(T) \leqslant n$, then $\kappa(\Sigma^*P \cap T) \leqslant 2^{m-1}n$.*

Proof. The language Σ^*P is the left ideal generated by P. It is known that the complexity of this ideal is at most 2^{m-1} [3]. Taking the intersection with T, we see that $2^{m-1}n$ is an upper bound on the complexity of $\Sigma^*P \cap T$. □

Next, for $m, n \geqslant 2$, we describe witnesses P_m and T_n that meet the upper bound. Let T_n be accepted by the DFA $\mathcal{D}_n = (Q_n, \Sigma, \delta_T, 0, \{n-1\})$, where $Q_n = \{0, 1, \ldots, n-1\}$, $\Sigma = \{a, b\}$, and the transformations are $a = (0, 1, \ldots, n-1)$ and $b = (1, 2, \ldots, n-1)$. See Figure 4. This DFA is minimal because a^{n-1-q} distinguishes q from any other state.

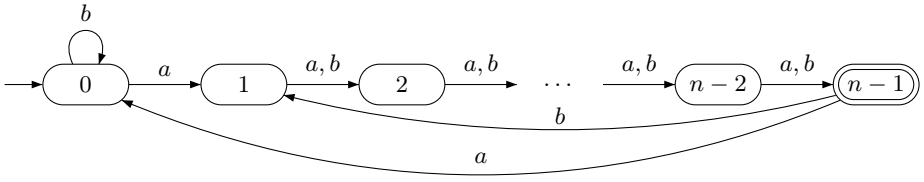

Figure 4: Minimal DFA $\mathcal{D}_n$ of T_n.

As in the previous section, let P_m be the language of the DFA $\mathcal{D}_m^{\leftrightarrow}$, which is just $\mathcal{D}_m$ with the roles of a and b switched: in $\mathcal{D}_m^{\leftrightarrow}$ we have $a = (1, 2, \ldots, m-1)$ and $b = (0, 1, \ldots, m-1)$. See Figure 5.

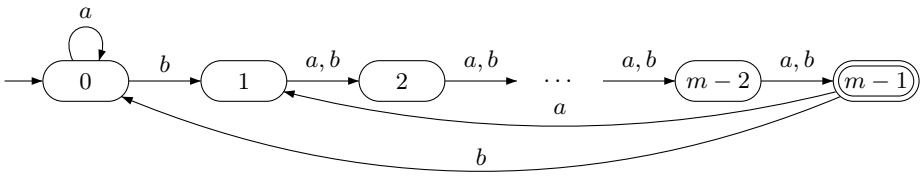

Figure 5: Minimal DFA $\mathcal{D}_m^{\leftrightarrow}$ of P_m.

Now we find a description of the left ideal $\Sigma^* P_m$. Let $E = \Sigma^{m-2}(a\Sigma^{m-2})^*$. Then P_m can be described by the regular expression $(a \cup bEb)^* bE$. Now we have

$$\Sigma^* P_m = \Sigma^*(a \cup bEb)^* bE = \Sigma^* bE = \Sigma^* b \Sigma^{m-2}(a\Sigma^{m-2})^*.$$

Thus $\Sigma^* P_m = \Sigma^* G_m$, where $G_m = b\Sigma^{m-2}(a\Sigma^{m-2})^*$. The words in G_m consist of a word of length $m-1$ beginning with b, possibly followed by some number of words of length $m-1$ beginning with a. An NFA accepting $\Sigma^* G_m$ is shown in Figure 6.

Next, we describe a DFA for the language $\Sigma^* P_m$. The states of this DFA are binary $(m-1)$-tuples, which we denote by $x = (x_1, \ldots, x_{m-1})$.

Definition 3. Define the following DFA:

$$\mathcal{B}_m = (\{0,1\}^{m-1}, \{a, b\}, (0, \ldots, 0), \beta, \{x \in \{0,1\}^{m-1} \mid x_1 = 1\}),$$

where $\beta((x_1, x_2, \ldots, x_{m-2}, x_{m-1}), \sigma) = \begin{cases} (x_2, x_3, \ldots, x_{m-1}, x_1), & \text{if } \sigma = a; \\ (x_2, x_3, \ldots, x_{m-1}, 1), & \text{if } \sigma = b. \end{cases}$

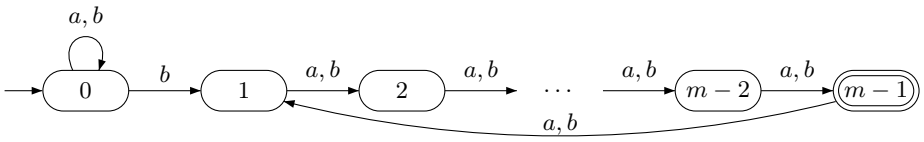

Figure 6: NFA for $\Sigma^* G_m = \Sigma^* P_m$.

In other words, the input $\sigma = a$ shifts the tuple x one position to the left cyclically, while $\sigma = b$ shifts the tuple to the left, losing the first component and replacing x_{m-1} by 1. DFA $\mathcal{B}_4$ is shown in Figure 7.

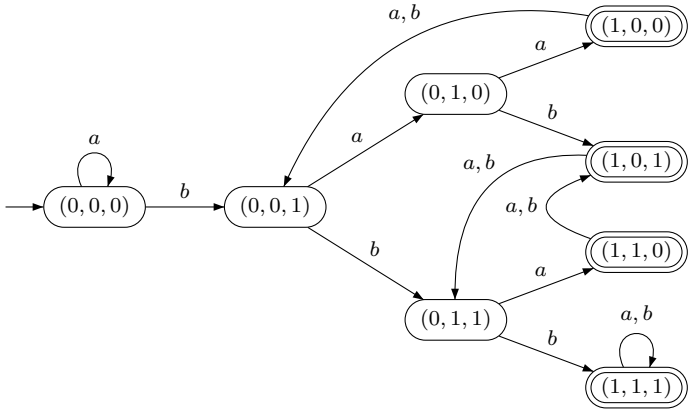

Figure 7: The DFA $\mathcal{B}_4$ for $\Sigma^* P_4$.

Proposition 4. *All the states of $\mathcal{B}_m$ are reachable and pairwise distinguishable.*

Proof. Consider a state $(x_1, \ldots, x_{m-1})$, and view it as the binary representation of a number k. State $k = 0$ is reachable by ε and $k = 1$ by b. If $k > 1$ is even, it is reachable from $k/2$ by a, and $k + 1$ is reachable from $k/2$ by b. Thus all the tuples in $\{0, 1\}^{m-1}$ are reachable.

We note that if a state $q = (x_1, \ldots, x_i, \ldots, x_{m-1})$ has $x_i = 1$, then q accepts the word a^{i-1}. For each state q, define $\mathbf{A}(q) = \{a^{i-1} \mid x_i = 1\}$. Since each state has a unique binary representation, each state has a unique $\mathbf{A}(q)$, which is a subset of all words accepted by q. Therefore, if p and q are distinct states, they are pairwise distinguishable by words in $\mathbf{A}(p) \cup \mathbf{A}(q)$. □

In the example of Figure 7, we have $\mathbf{A}(001) = \{aa\}$, $\mathbf{A}(010) = \{a\}$, $\mathbf{A}(011) = \{a, aa\}$, $\mathbf{A}(100) = \{\varepsilon\}$, $\mathbf{A}(101) = \{\varepsilon, aa\}$, $\mathbf{A}(110) = \{\varepsilon, a\}$, $\mathbf{A}(111) = \{\varepsilon, a, aa\}$.

Lemma 5. *DFA $\mathcal{B}_m$ is a minimal DFA for $\Sigma^* P_m$.*

PROOF. Recall that $\Sigma^* P_m = \Sigma^* G_m$, where $G_m = b\Sigma^{m-2}(a\Sigma^{m-2})^*$. We know from Proposition 4 that $\mathcal{B}_m$ is minimal, so it suffices to prove it accepts $\Sigma^* G_m$.

First we prove that each state $(x_1, \ldots, x_{m-1})$ of $\mathcal{B}_m$ accepts G_m, and thus $\mathcal{B}_m$ accepts a superset of $\Sigma^* G_m = \Sigma^* P_m$. Let w be an arbitary word from G_m. Since w begins with b, this letter "loads" a 1 into position x_{m-1}. Then this b is followed by $m-2$ arbitrary letters, which shift the 1 into position x_1. If there is no more input, the word w is accepted. Otherwise, the next letter is an a. This shifts the positions left cyclically, moving the 1 from position x_1 back into position x_{m-1}. Following the a, we have $m-2$ arbitrary letters, which shift the 1 to position x_1. If there is no more input, the word w is accepted; otherwise the next letter must be an a, and the behaviour just described repeats until there is no more input. Thus G_m is accepted from every state and so $\Sigma^* G_m \subseteq L(\mathcal{B}_m)$.

Next we prove that $L(\mathcal{B}_m) \subseteq \Sigma^* G_m$. If $w \in L(\mathcal{B}_m)$, then w has length at least $m-1$. Let $w = \sigma_1\sigma_2\cdots\sigma_k$, where $\sigma_i \in \Sigma$. Consider the prefix $\sigma_1\cdots\sigma_{k-(m-2)}$ of w. If $\sigma_{k-(m-2)} = b$, then w is in $\Sigma^* b\Sigma^{m-2} \subseteq \Sigma^* G_m$ and we are done. On the other hand, if $\sigma_{k-(m-2)} = a$, then w is in $\Sigma^* a\Sigma^{m-2}$. Now our proof strategy is as follows: jump back $m-1$ letters and look at $\sigma_{k-(m-2)-(m-1)}$. If this letter is a b, then w is in $\Sigma^* b\Sigma^{m-2}a\Sigma^{m-2} \subseteq \Sigma^* G_m$ and we are done. If it's an a, then w is in $\Sigma^*(a\Sigma^{m-2})^2$, and we can keep jumping back $m-1$ letters at a time until we find a b.

More formally, we claim there exists $\ell \geqslant 0$ such that $\sigma_{k-(m-2)-\ell(m-1)} = b$, and for $0 \leqslant i < \ell$ we have $\sigma_{k-(m-2)-i(m-1)} = a$; thus w is in $\Sigma^* b\Sigma^{m-2}(a\Sigma^{m-2})^\ell$. To see this, suppose the claim is false. We can write $k-(m-2) = \ell(m-1)+j$, where ℓ is the quotient upon dividing $k-(m-2)$ by $m-1$, and j is the remainder with $0 \leqslant j < m-1$. Since the claim is false, we have $\sigma_j = \sigma_{k-(m-2)-\ell(m-1)} = a$. In fact, we have $\sigma_{k-(m-2)-i(m-1)} = a$ for $0 \leqslant i \leqslant \ell$. It follows that w is in $\sigma_1\cdots\sigma_{j-1}(a\Sigma^{m-2})^{\ell+1}$. Since $j-1 < m-1$, the prefix $\sigma_1\cdots\sigma_{j-1}$ cannot lead to an accepting state. Now, if we are in a non-accepting state, and we apply a word from the language $(a\Sigma^{m-2})^*$, we will remain in a non-accepting state. Thus w is not accepted, which is a contradiction. So the claim must be true, and this completes the proof. □

To prove that the intersection $\Sigma^* P_m \cap T_n$ has complexity $2^{m-1}n$, we construct the direct product of the DFAs $\mathcal{B}_m$ and $\mathcal{D}_n$ and prove that it is minimal. We will use the following lemma in the proof of reachability:

Lemma 6. *If (a) the DFAs $\mathcal{B}$ and $\mathcal{D}$ are minimal, (b) in $\mathcal{D}$, the transformation $w\colon Q \to Q$ is bijective for all $w \in \Sigma^*$, and (c) every state in $\{p_0\}\times Q$ is reachable in $\mathcal{B}\times\mathcal{D}$, then every state in $P\times Q$ is reachable in $\mathcal{B}\times\mathcal{D}$.*

PROOF. We prove that each state $(p,q) \in P\times Q$ is reachable. Choose $w \in \Sigma^*$ such that $p_0 w = p$; such a word exists since $\mathcal{B}$ is minimal by (a). By (b) the transformation induced by w in $\mathcal{D}$ is bijective and thus has an inverse. By (c), the state (p_0, qw^{-1}) is reachable in $\mathcal{B}\times\mathcal{D}$. Then (p,q) is reachable from this state via w. □

We can now prove the following theorem:

Theorem 7. *For $m, n \geqslant 2$, if $\kappa(P) \leqslant m$ and $\kappa(T) \leqslant n$, then $\kappa(\Sigma^* P \cap T) \leqslant 2^{m-1}n$, and this bound is tight if the cardinality of Σ is at least 2.*

Proof. The upper bound follows from Proposition 2. To prove that the upper bound is tight, we show that all states in the direct product $\mathcal{B}_m \times \mathcal{D}_n$ are reachable and pairwise distinguishable.

Reachability. Let $B = \{0,1\}^{m-1}$ denote the state set of $\mathcal{B}_m$ and let v_0 denote the initial state of $\mathcal{B}_m$. The initial state of the direct product is $(v_0, 0)$. Every state of the form (v_0, q), where $0 \leqslant q \leqslant n-1$, is reachable by a^q. In $\mathcal{D}_n$, the transformations $a = (0, 1, \ldots, n-1)$ and $b = (1, 2, \ldots, n-1)$ are bijective on $\{0, 1, \ldots, n-1\}$; thus for every $w \in \{a, b\}^*$, the function $w\colon Q \to Q$ is also bijective. Hence we may apply Lemma 6. It follows that every state in $B \times \{0, \ldots, n-1\}$ is reachable.

Distinguishability. First note the following facts about $\mathcal{B}_m$:

- The word b^{m-1} sends all states to the final state $(1, 1, \ldots, 1)$.
- The final state $(1, 1, \ldots, 1)$ is fixed by all words in $\{a, b\}^*$.
- The letter a permutes the states. Thus if u and v are distinct, then ua and va are distinct.
- Suppose $u = (u_1, u_2, \ldots, u_{m-1})$ and $v = (v_1, v_2, \ldots, v_{m-1})$ are states, and define $d(u, v)$ to be the largest integer i such that $u_i \neq v_i$, or 0 if the states are equal. If $d(u, v) = 1$, then u and v are distinguishable by ε (that is, one is final and one is non-final).
- If $d(u, v) \neq 1$, then $b^{d(u,v)-1}$ sends u to a state u' and v to a state v' such that $d(u', v') = 1$.

Now, let (u, p) and (v, q) be distinct states, where $u, v \in \{0,1\}^{m-1}$.

Case 1. $p \neq q$. Without loss of generality, we can assume $u = v = (1, 1, \ldots, 1)$; otherwise apply b^{m-1}. Choose a word w that distinguishes p and q in $\mathcal{D}_n$; then w distinguishes (u, p) and (v, q).

Case 2. $p = q$ (and thus $u \neq v$). We may assume without loss of generality that u and v differ in exactly one component, and that $p = n - 1$. Otherwise, first apply $b^{d(u,v)-1}$ to reach (u', p') and (v', p') such that $d(u', v') = 1$, and note that this implies u' and v' differ in exactly one component. Then apply $a^{n-1-p'}$ to send p' to $n - 1$.

Suppose now that u and v differ in exactly one component and $p = n - 1$. Then $d(u, v)$ is the index of the component where u and v differ. Furthermore, if we apply a word $w \in \{a, b\}^*$, then either $uw = vw$, or uw and vw differ in exactly one component and $d(uw, vw)$ is the index of this component. So as long as w does not erase u and v's differing component, it can be used to shift the differing component's index.

If $d(u, v) = 1$, then $(u, n-1)$ and $(v, n-1)$ are distinguishable by ε. So suppose $d(u, v) > 1$, and set $i = d(u, v)$. Observe that:

- For all $k \geqslant 0$, we have $d(ua^k, va^k) \equiv i - k \pmod{m-1}$.
- For all $k \geqslant 0$, since $d(u, v) = i > 1$, we have $d(uba^k, vba^k) \equiv i - k - 1 \pmod{m-1}$.

Since a^n and ba^{n-2} both fix $p = n-1$, it follows that:

- If we are in states $(u, n-1)$ and $(v, n-1)$ and apply a^n, we reach $(ua^n, n-1)$ and $(va^n, n-1)$ where $d(ua^n, va^n)$ is the unique element of $\{1, \ldots, m-1\}$ equivalent to $i - n$ modulo $m-1$.
- If we are in states $(u, n-1)$ and $(v, n-1)$ and apply ba^{n-2}, we reach $(uba^{n-2}, n-1)$ and $(vba^{n-2}, n-1)$, where $d(uba^{n-2}, vba^{n-2})$ is the unique element of $\{1, \ldots, m-1\}$ equivalent to $i - (n-1)$ modulo $m-1$.

Let $x = a^n$ and $y = ba^{n-2}$. Apply x^{i-1} to the states to reach $(ux^{i-1}, n-1)$ and $(vx^{i-1}, n-1)$, where $d(ux^{i-1}, vx^{i-1})$ is the unique element of $\{1, \ldots, m-1\}$ equivalent to $i - (i-1)n$ modulo $m-1$. We claim that we can now distinguish $(ux^{i-1}, n-1)$ and $(vx^{i-1}, n-1)$ by applying y^k for some value $k \geqslant 0$.

We choose k to be the least integer such that $d(ux^{i-1}y^k, vx^{i-1}y^k) = 1$. Clearly if such a k exists, then y^k distinguishes the states, so we just have to show that k exists. Suppose for a contradiction that k does not exist. Observe then that $d(ux^{i-1}y^\ell, vx^{i-1}y^\ell) > 1$ for all $\ell \geqslant 0$. Otherwise, we can choose a minimal ℓ so that $d(ux^{i-1}y^\ell, vx^{i-1}y^\ell) = 0$; then we necessarily have $d(ux^{i-1}y^{\ell-1}, vx^{i-1}y^{\ell-1}) = 1$, since the only way we can have $u'y = v'y$ is if $d(u', v') \leqslant 1$. It follows then that we can take $k = \ell - 1$. Now, set $\ell = (i-1)(m-2)$. Since $d(ux^{i-1}y^j, vx^{i-1}y^j) > 1$ for all $j \leqslant \ell$, it follows that $d(ux^{i-1}y^\ell, vx^{i-1}y^\ell)$ is the unique element of $\{1, \ldots, m-1\}$ equivalent to $i - (i-1)n - \ell(n-1)$ modulo $m-1$. Indeed, each application of y subtracts $n-1$ (modulo $m-1$) from the component where the bit tuples differ, and since we always have $d(ux^{i-1}y^j, vx^{i-1}y^j) > 1$, the states are never mapped to the same state by the b at the start of y. But now, we have

$$i-(i-1)n-\ell(n-1) = i-(i-1)n-(i-1)(m-2)(n-1) = i-(i-1)(n+(m-2)(n-1)).$$

Since $m - 2 \equiv -1 \pmod{m-1}$, we have

$$i - (i-1)n - \ell(n-1) \equiv i - (i-1)(n-n+1) \equiv i - (i-1) \equiv 1 \pmod{m-1}.$$

So in fact $d(ux^{i-1}y^\ell, vx^{i-1}y^\ell) = 1$. This is a contradiction, and so the integer k exists. Thus if we set $w = x^{i-1}y^k$, the states $(u, n-1)$ and $(v, n-1)$ are distinguished by w (note that both x and y fix the second component $n-1$). □

5. Factor Matching

We want to find the worst-case state complexity of the set of all the words of T that have words of P as factors, that is, $\Sigma^* P \Sigma^* \cap T$.

Proposition 8. *For $m, n \geqslant 3$, if $\kappa(P) \leqslant m$ and $\kappa(T) \leqslant n$, then $\kappa(\Sigma^* P \Sigma^* \cap T) \leqslant (2^{m-2} + 1)n$.*

PROOF. The language $\Sigma^* P \Sigma^*$ is the two-sided ideal generated by P. It is known that the complexity of $\Sigma^* P \Sigma^*$ is at most $2^{m-2}+1$ [3]. Thus the complexity of the intersection with T is at most $(2^{m-2}+1)n$. □

The proof of tightness is quite similar to the proof of tightness for suffix matching. For witnesses, let T_n be accepted by the DFA $\mathcal{D}_n = (Q_n, \Sigma, \delta_T, 0, \{n-1\})$, where $Q_n = \{0, 1, \ldots, n-1\}$, $\Sigma = \{a, b\}$, and $a = (0, 1, \ldots, n-1)$, $b = (1, 2, \ldots, n-2)$. See Figure 8. This DFA is minimal because a^{n-1-q} distinguishes q from the other states. As before, the other witness P_m is the language of the DFA $\mathcal{D}_m^{\leftrightarrow}$, which is $\mathcal{D}_m$ with the roles of a and b swapped. See Figure 9.

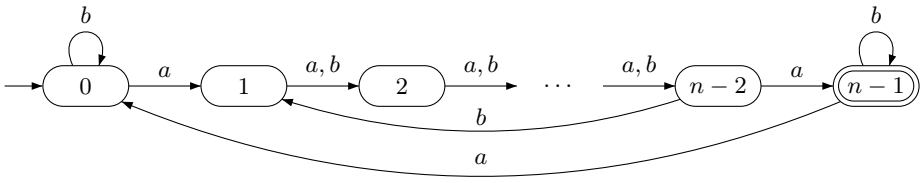

Figure 8: Minimal DFA $\mathcal{D}_n$ of T_n.

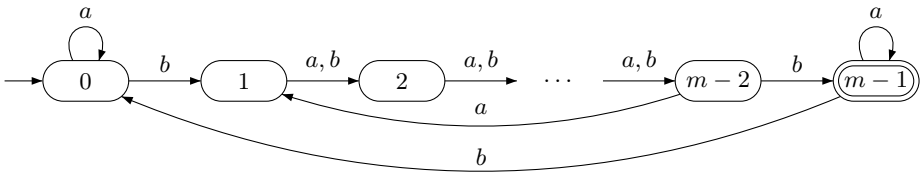

Figure 9: Minimal DFA $\mathcal{D}_m^{\leftrightarrow}$ of P_m.

Let $E = \Sigma^{m-3}(a\Sigma^{m-3})^*$. Then the language P_m can be described by the regular expression $(a \cup bEba^*b)^*bEba^*$. Now we have

$$\Sigma^* P_m(b,a)\Sigma^* = \Sigma^*(a \cup bEba^*b)^*bEba^*\Sigma^* = \Sigma^* bEb\Sigma^*.$$

Thus $\Sigma^* P_m \Sigma^* = \Sigma^* G_m \Sigma^*$, where $G_m = bEb = b\Sigma^{m-3}(a\Sigma^{m-3})^*b$.

Next we describe a DFA for the language $\Sigma^* P_m \Sigma^*$. For states we use binary $(m-2)$-tuples which we denote by $x = (x_1, \ldots, x_{m-2})$.

Definition 9. Define the following DFA:

$$\mathcal{C}_m = (\{0,1\}^{m-2} \cup \{f\}, \{a,b\}, (0,\ldots,0), \gamma, \{f\}),$$

where $\gamma(f, \sigma) = f$ for all $\sigma \in \Sigma$, and

$$\gamma((x_1, x_2, \ldots, x_{m-3}, x_{m-2}), \sigma) = \begin{cases} (x_2, x_3, \ldots, x_{m-2}, x_1), & \text{if } \sigma = a; \\ (x_2, x_3, \ldots, x_{m-2}, 1), & \text{if } \sigma = b,\ x_1 = 0; \\ f, & \text{if } \sigma = b,\ x_1 = 1. \end{cases}$$

In other words, if $x \neq f$, input $\sigma = a$ shifts x one position to the left cyclically; input $\sigma = b$ shifts the tuple to the left, losing the leftmost component and replacing x_{m-2} by 1 if $x_1 = 0$. Finally, γ sends the state to f if $x_1 = 1$ and $\sigma = b$, and all inputs are the identity on f. DFA $\mathcal{C}_5$ is shown in Figure 10.

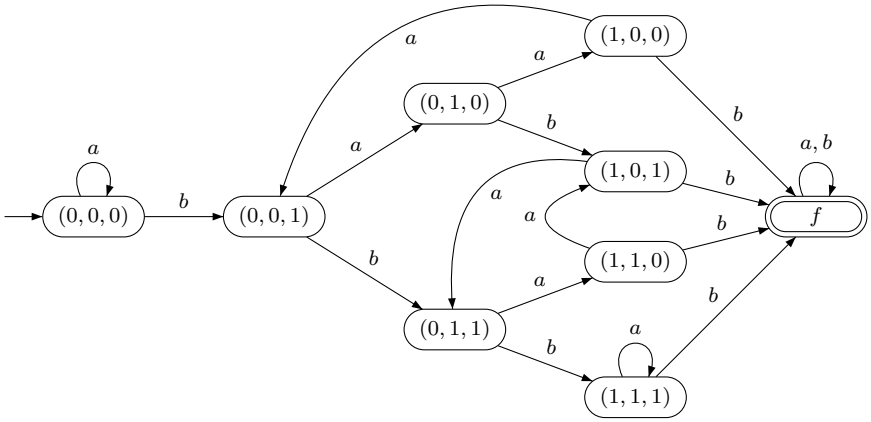

Figure 10: The DFA $\mathcal{C}_5$ for $\Sigma^* P_5 \Sigma^*$.

Proposition 10. *All the states of $\mathcal{C}_m$ are reachable and pairwise distinguishable.*

Proof. Consider a state $(x_1, \ldots, x_{m-2})$, and view it as the binary representation of a number k. Then k is reachable as in the proof of Proposition 4, and f is reached by applying b to any state that has $x_1 = 1$.

We note that if a state $q = (x_1, \ldots, x_i, \ldots, x_{m-2})$ has $x_i = 1$, then q accepts the word $a^{i-1}b$. Define $\mathbf{Ab}(q) = \{a^{i-1}b \mid x_i = 1\}$. As in Proposition 4, each binary (that is, non-f) state has a unique binary representation, and so each of these states has a unique $\mathbf{Ab}(q)$, which is a subset of all words accepted by q. Therefore, if p and q are distinct binary states, they are pairwise distinguishable by words in $\mathbf{Ab}(p) \cup \mathbf{Ab}(q)$. We observe that f is the only final state, and is therefore distinguishable from every other state by ε. □

In the example of Figure 10, we have $\mathbf{Ab}(001) = \{aab\}$, $\mathbf{Ab}(010) = \{ab\}$, $\mathbf{Ab}(011) = \{ab, aab\}$, $\mathbf{Ab}(100) = \{b\}$, $\mathbf{Ab}(101) = \{b, aab\}$, $\mathbf{Ab}(110) = \{b, ab\}$, $\mathbf{Ab}(111) = \{b, ab, aab\}$.

Lemma 11. *$\mathcal{C}_m$ is a minimal DFA for $\Sigma^* P_m \Sigma^*$.*

Proof. First we prove that each state of $\mathcal{C}_m$ accepts $G_m \Sigma^*$, and thus $\mathcal{C}_m$ accepts $\Sigma^* G_m \Sigma^* = \Sigma^* P_m \Sigma^*$. Since f accepts Σ^*, it also accepts $G_m \Sigma^*$. In binary

states of the form $(x_1, x_2, \ldots, x_{m-2})$, applying b "loads" a 1 into x_{m-2}. Then after applying a word from Σ^{m-3}, the resulting state will either be a binary state where $x_1 = 1$, or f. If the current state is f, then no matter what inputs are applied, the word will be accepted, and hence $G_m\Sigma^*$ is accepted. If the current state is a binary state with $x_1 = 1$, then applying a will cycle the 1 at x_1 to x_{m-2}, and applying a word from Σ^{m-3} will either shift the 1 back to x_1 or move to f if another 1 in the state is shifted to x_1 and b is applied. Therefore, applying a word from $(a\Sigma^{m-3})^*$ from a state where $x_1 = 1$ will result in either a binary state where $x_1 = 1$ or f, and applying a word from $b\Sigma^*$ from one of those states will result in f, so $G_m\Sigma^*$ is accepted. Therefore, $\Sigma^*G_m\Sigma^* \subseteq L(\mathcal{C}_m)$.

We now show that $L(\mathcal{C}_m) \subseteq \Sigma^*G_m\Sigma^*$. First, we observe that every word in $L(\mathcal{C}_m)$ has a length of at least $m-1$ and at least two bs: a b to load a 1 into x_{m-2}, $m-3$ letters to shift the 1 to x_1, and a b to move to f. Let $w = \sigma_1 \ldots \sigma_k$ be a word in $L(\mathcal{C}_m)$, where each $\sigma_i \in \Sigma$. Suppose that the j-th letter of w is what first causes a transition to f; in other words, $(0, \ldots, 0)\gamma_{\sigma_1 \ldots \sigma_{j-1}} \neq f$ and $(0, \ldots, 0)\gamma_{\sigma_1 \ldots \sigma_j} = f$. The remaining letters in w, $\sigma_{j+1} \ldots \sigma_k$, do not matter since they cannot cause a transition away from f, so we only need to consider the prefix $w_j = \sigma_1 \ldots \sigma_j$.

Now the rest of the argument is similar to the proof of Lemma 5. Letter σ_j of w_j must be a b. Look at letter $\sigma_{j-1-(m-3)}$. If this letter is a b, then w_j is in $\Sigma^*b\Sigma^{m-3}b$, and so w is in $\Sigma^*b\Sigma^{m-3}b\Sigma^* \subseteq \Sigma^*G_m\Sigma^*$, and we are done. If the letter $\sigma_{j-1-(m-3)}$ is an a, we keep jumping back $m-2$ letters at a time until we find a b. In other words, we choose $\ell \geqslant 0$ as small as possible such that $\sigma_{j-1-(m-3)-\ell(m-2)} = b$. If no such ℓ exists, then as in the proof of Lemma 5, one can show that w_j must be in $\Sigma^i(a\Sigma^{m-3})^*b$ with $i < m-2$ and that w is not accepted. So ℓ must exist, and therefore w_j is in $\Sigma^*b\Sigma^{m-3}(a\Sigma^{m-3})^\ell b$, which implies $w \in \Sigma^*G_m\Sigma^*$. □

We can now prove the following theorem:

Theorem 12. *For $m, n \geqslant 3$, if $\kappa(P) \leqslant m$ and $\kappa(T) \leqslant n$, then $\kappa(\Sigma^*P\Sigma^* \cap T) \leqslant (2^{m-2}+1)n$, and this bound is tight if the cardinality of Σ is at least 2.*

Proof. The upper bound follows from Proposition 8. To prove that the upper bound is tight, we show that all states in the direct product $\mathcal{C}_m \times \mathcal{D}_n$ are reachable and pairwise distinguishable. The reachability arguments are essentially the same as those of Theorem 7. Distinguishability is also similar, but the second case is much simpler. First note that b^{m-1} sends every state of $\mathcal{C}_m$ to f. For states u and v of $\mathcal{C}_m$, define the function $d(u, v)$ as follows:

$$d(u,v) = \begin{cases} -1, & \text{if } u = v; \\ 0, & \text{if } u = f \text{ or } v = f; \\ \min\{i \mid u_i \neq v_i\}, & \text{if } u = (u_1, \ldots, u_{m-2}) \text{ and } v = (v_1, \ldots, v_{m-2}). \end{cases}$$

Suppose we have two distinct states in $\mathcal{C}_m \times \mathcal{D}_n$: (u, p) and (v, q).

Case 1. $p \neq q$. Assume that $u = v$; if not, apply b^{m-1} to send both to f. (f, pb^{m-1}) and (f, qb^{m-1}) can be distinguished by $a^{n-1-pb^{m-1}}$. Note pb^{m-1} and qb^{m-1} cannot be equal since b^{m-1} is bijective on the states of $\mathcal{D}_n$.

Case 2. $p = q$ (so $u \neq v$). Assume that $d(u,v) = 0$; if not, apply $a^{d(u,v)-1}b$ to send either u or v to f. Then we have the states $(ua^{d(u,v)-1}b, pa^{d(u,v)-1}b)$ and $(va^{d(u,v)-1}b, pa^{d(u,v)-1}b)$. Let us define $p' = pa^{d(u,v)-1}b$; the two states can be distinguished by $a^{n-1-p'}$. □

6. Subsequence Matching

We now turn to the worst-case complexity of the set of all the words of T that have words of P as subsequences, that is, the set $(\Sigma^* \sqcup P) \cap T$.

Theorem 13. *For $m, n \geqslant 3$, if $\kappa(P) \leqslant m$ and $\kappa(T) \leqslant n$, then $\kappa((\Sigma^* \sqcup P) \cap T) \leqslant (2^{m-2}+1)n$, and this bound is tight if $|\Sigma| \geqslant m-1$.*

Proof. Okhotin [9] proved that if $\kappa(P) \leqslant m$, then $(\Sigma^* \sqcup P)$ has state complexity at most $2^{m-2}+1$, and this bound is tight. Okhotin's witness is the DFA $(Q_m, \Sigma, \delta, 0, \{m-1\})$, where $\Sigma = \{a_1, \ldots, a_{m-2}\}$ and $a_i \colon (i \to m-1)(0 \to i)$; the alphabet size $m-2$ cannot be reduced. Thus the state complexity of $(\Sigma^* \sqcup P) \cap T$ is at most $(2^{m-2}+1)n$.

We define P_m as a slight modification of Okhotin's witness, with $m-1$ letters instead of $m-2$. Define $\mathcal{D}_m = (Q_m, \Sigma, \delta, 0, \{m-1\})$ where $\Sigma = \{a_1, \ldots, a_{m-2}, b\}$, $a_i \colon (i \to m-1)(0 \to i)$ as before, and $b \colon \mathbb{1}$. See Figure 11. Let P_m be the language of $\mathcal{D}_m$.

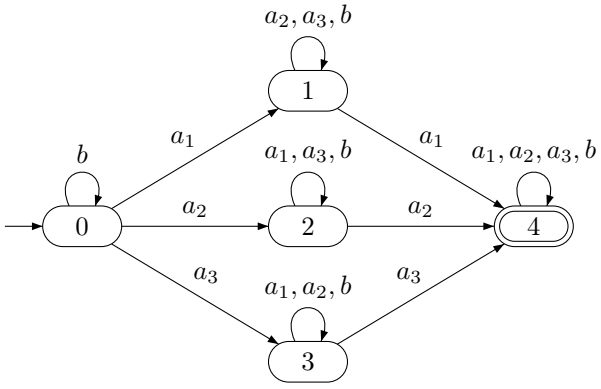

Figure 11: DFA $\mathcal{D}_5$ of P_5 for subsequence matching.

For T_n we use the language of the DFA $\mathcal{A}_n = (Q_n, \Sigma, \delta', 0, \{n-1\})$ where $a_i \colon \mathbb{1}$ for $1 \leqslant i \leqslant m-2$ and $b \colon (0, 1, \ldots, n-1)$. See Figure 12.

Let $\mathcal{S}_m$ be a minimal DFA for the shuffle $(\Sigma^* \sqcup P_m)$ with state set S and initial state s_0. Note that all states of $\mathcal{S}_m$ are reachable from s_0, and pairwise distinguishable from each other, using words over $\{a_1, \ldots, a_{m-2}\}$ (that is, without using b). This follows from the fact that our DFA $\mathcal{D}_m$ for P_m was constructed using Okhotin's witness as a base.

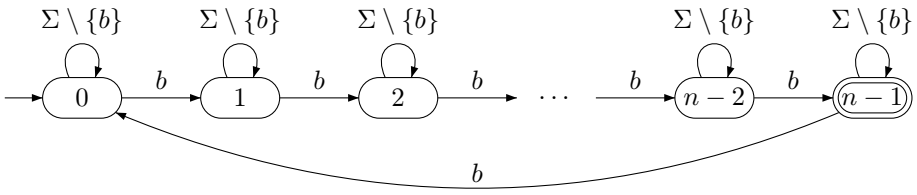

Figure 12: DFA $\mathcal{A}_n$ of T_n for subsequence matching.

Consider the direct product $\mathcal{S}_m \times \mathcal{A}_n$, which recognizes $(\Sigma^* \sqcup P_m) \cap T_n$. The states of the direct product have the form (s, q) where s is a state of $\mathcal{S}_m$ and q is a state of $\mathcal{A}_n$. The initial state of the direct product is $(s_0, 0)$. By words over $\{a_1, \ldots, a_{m-2}\}$ we can reach all states of the form $(s, 0)$ for $s \in S$. Then by words over b^* we reach all states (s, q) for all $s \in S$ and $q \in Q$. So all $(2^{m-2} + 1)n$ states of the direct product are reachable.

For distinguishability, consider two distinct states (s, q) and (s', q'). The final state set of the direct product is $\{(s_F, n-1) \mid s_F \text{ is final in } \mathcal{S}_m\}$. Suppose $q \neq q'$. Since $\mathcal{S}_m$ is minimal, it has at most one empty state. Hence one of s or s' can be mapped to a final state by some word w over $\{a_1, \ldots, a_{m-2}\}$. If we have states (s, q) and (s', q') with one of s or s' final, then a word in b^* distinguishes the states.

Now suppose $q = q'$; then $s \neq s'$. Apply b^{n-1-q} to reach states $(s, n-1)$ and $(s', n-1)$. By minimality of $\mathcal{S}_m$, there is a word over $\{a_1, \ldots, a_{m-2}\}$ that distinguishes s and s'; this word also distinguishes $(s, n-1)$ and $(s', n-1)$. □

The following proposition shows that the alphabet size of our witness cannot be reduced: an alphabet of $m-1$ letters is optimal for this operation.

Proposition 14. *Let P and T be regular languages with $\kappa(P) \leqslant m$ and $\kappa(T) \leqslant n$, both over an alphabet Σ of size less than $m-1$. If $n \geqslant 2$, then $\kappa((\Sigma^* \sqcup P) \cap T) < (2^{m-2} + 1)n$.*

Proof. To prove this, we need to understand the structure of the minimal DFA for $\Sigma^* \sqcup P$. Okhotin [9] proved that if $\mathcal{D} = (Q, \Sigma, \delta, q_0, F)$ recognizes P, then the NFA $\mathcal{N} = (Q, \Sigma, \Delta, q_0, F)$, where $\Delta(q, \sigma) = \{q, q\sigma\}$, recognizes $\Sigma^* \sqcup P$. We obtain a minimal DFA $\mathcal{S}$ for $\Sigma^* \sqcup P$ by determinizing and minimizing $\mathcal{N}$; the states of the DFA $\mathcal{S}$ are subsets of Q. Write $S\sigma'$ as shorthand for $\Delta(S, \sigma)$. Then in $\mathcal{S}$, we have $S\sigma' = S \cup S\sigma$.

Consider the direct product of $\mathcal{S}$ with an arbitrary n-state DFA. Assume without loss of generality that the DFA $\mathcal{D}$ for P has state set Q_m and initial state 0, and the arbitary n-state DFA has state set Q_n and initial state 0. Then the initial state of the direct product is $(\{0\}, 0)$. The only way we can reach states of the form $(\{0\}, q)$ with $q \neq 0$ is if $\{0\}\sigma' = \{0\}$ for some letter σ. In other words, at least one letter must induce a self-loop on $\{0\}$, or else the maximal number of states in the direct product is not reachable. Our alphabet has size

strictly less than $m-1$, and one of the letters in our alphabet induces a self-loop on $\{0\}$, so there are at most $m-3$ letters that do not induce a self-loop on $\{0\}$.

Now, we mimic Okhotin's argument from Lemma 4.4 in [9]. Notice that in the NFA $\mathcal{N}$, we have $S \subseteq S\sigma'$ for all $\sigma \in \Sigma$. Thus every reachable subset of states in this NFA contains the initial state 0. Additionally, if two subsets S and T in the NFA $\mathcal{N}$ both contain a final state, then they are indistinguishable in the DFA $\mathcal{S}$, since from these sets we can only reach other sets containing a final state. If $\mathcal{N}$ has k final states, then there are 2^{m-k-1} sets that contain 0 but do not contain a final state, and the remaining sets are indistinguishable. It follows there are at most $2^{m-k-1}+1$ indistinguishability equivalence classes. If $k \geqslant 2$, this is strictly less than the upper bound. Thus we may assume that $k = 1$, that is, there is a unique accepting state. To reach the upper bound, *all* sets which do not contain the accepting state must be reachable.

Consider subsets of states in $\mathcal{N}$ of the form $\{0, p\}$ for $p \neq 0$ and p non-final; there are $m-2$ such sets, since there is only one accepting state. Since $S \subseteq S\sigma'$ for all $\sigma \in \Sigma$, the only way we can reach a set $\{0, p\}$ is by a self-loop on $\{0, p\}$, or by a direct transition from a smaller set. But the only smaller reachable set is the initial set $\{0\}$. So if $\{0, p\}$ is reachable, then it is reachable by a direct transition from $\{0\}$.

Now, we know one letter induces a self-loop on $\{0\}$, so it is not useful for reaching states of the form $\{0, p\}$. We have at most $m-3$ letters that do not induce a self-loop on $\{0\}$, so we can reach at most $m-3$ sets of the form $\{0, p\}$. Since there are $m-2$ such sets, at least one set must be unreachable, and thus the upper bound on the state complexity of $(\Sigma^* \sqcup P) \cap T$ cannot be reached. □

We did not investigate the state complexity of subsequence matching over a fixed-size alphabet. For a three-letter alphabet, Okhotin [9] proved a lower bound of $\frac{1}{5}4^{\sqrt{m/2}}m^{-\frac{3}{4}}$ on the state complexity of $\Sigma^* \sqcup P$; this could perhaps be used to derive a similar bound for $(\Sigma^* \sqcup P) \cap T$, but we did not pursue this.

7. Conclusions

We investigated the state complexity of four new combined operations on regular languages, inspired by pattern matching problems. The operations we considered were of the form "the intersection of T with the right (left, two-sided, all-sided) ideal generated by P", corresponding to searching for prefixes (suffixes, factors, subsequences) from a set of patterns P in a set of texts T. We found that the state complexity of these combined operations is just equal to the composition of the complexities of the individual operations; the complexity is polynomial in the case of prefix matching, and exponential (in the first parameter) in the case of suffix, factor and subsequence matching. For all four operations, the maximal complexity can be achieved only by languages over an alphabet of at least two letters. In the case of prefix, suffix and factor matching, two letters actually suffice, while for subsequence matching a linearly growing alphabet is needed to reach the maximal complexity. The complexity of subsequence matching for a fixed-size alphabet remains open.

We also considered the special cases where the pattern is a single word, and where both the pattern and text are unary. In these cases the complexity is significantly lower. The details of these cases can be found in the arXiv version of the paper [2].

References

[1] A. Aho, M.J. Corasick, Efficient string matching: An aid to bibliographic search, Communications of the ACM 18 (1975) 333–340.

[2] J.A. Brzozowski, S. Davies, A. Madan, State complexity of pattern matching in regular languages, 2018. `http://arxiv.org/abs/1806.04645`.

[3] J.A. Brzozowski, G. Jirásková, B. Li, Quotient complexity of ideal languages, Theoret. Comput. Sci. 470 (2013) 36–52.

[4] M. Crochemore, C. Hancart, Automata for matching patterns, in: G. Rozenberg, A. Salomaa (Eds.), Handbook of Formal Languages, volume 2, Springer, 1997, pp. 399–462.

[5] L.K. Dillon, G.S. Avrunin, J.C. Wileden, Constrained expressions: Toward broad applicability of analysis methods for distributed software systems, ACM Trans. Program. Lang. Syst. 10 (1988) 374–402.

[6] M. Elloumi, C. Iliopoulos, J.T. Wang, A.Y. Zomaya, Pattern Recognition in Computational Molecular Biology: Techniques and Approaches, Wiley, 2015.

[7] Y. Gao, N. Moreira, R. Reis, S. Yu, A survey on operational state complexity, J. Autom. Lang. Comb. 21 (2016) 251–310.

[8] A.N. Maslov, Estimates of the number of states of finite automata, Dokl. Akad. Nauk SSSR 194 (1970) 1266–1268 (Russian). English translation: Soviet Math. Dokl. **11** (1970) 1373–1375.

[9] A. Okhotin, On the state complexity of scattered substrings and superstrings, Fundamenta Informaticae 99 (2010) 325–338.

[10] A. Salomaa, K. Salomaa, S. Yu, State complexity of combined operations, Theoret. Comput. Sci. 383 (2007) 140–152.

[11] S. Yu, Q. Zhuang, K. Salomaa, The state complexities of some basic operations on regular languages, Theoret. Comput. Sci. 125 (1994) 315–328.