7 Appendix

7.1 Proof of Theorem 1

(1) An FA(&) recognizes the language defined by a regular expression with shuffle, where each alphabet symbol occurs at most once.

Proof. Let $res^{\leq 1}$ denote a regular expression with shuffle, where each alphabet symbol occurs at most once. For a regular expression r, if an FA(&) recognizes the language $\mathcal{L}(r)$, then, for the ith subexpression of the form $r_i = r_{i_1} \& r_{i_2} \& \cdots \& r_{i_k}$ $(i, k \in \mathbb{N}, k \geq 2)$ in r, there are start marker $\&_i$ and end marker $\&_i^+$ in an FA(&) for recognizing the strings derived by r_i. For each subexpression r_{i_j} $(1 \leq j \leq k)$ in r_i, there is a concurrent marker $||_{ij}$ in an FA(&) for recognizing the symbols or strings derived by r_{i_j}. Let the regular expression r be denoted by a $res^{\leq 1}$.

For strings recognition, an FA(&) recognizes a string by treating symbols in a string individually. A symbol y in a string $s \in \mathcal{L}(r)$ is recognized if and only if the current state (a set of nodes) p is reached such that $y \in p$. The end symbol $\dashv$ is recognized if and only if the final state is reached. If y (resp. $\dashv$) is not consumed, then y (resp. $\dashv$) will be still read as the current symbol to be recognized.

Since the node in the node transition graph of an FA(&) is labelled by distinct symbols (including alphabet symbols), and the $res^{\leq 1}$ r where each alphabet symbol occurs at most once is a deterministic expression, every symbol in s can be uniquely matched in r, and for every symbol l in r, there must exist a state (a set of nodes) in an FA(&) including l. According to the transition function of an FA(&), for the FA(&) $\mathcal{A}$, every symbol in s can be recognized in a state in $\mathcal{A}$. When the last symbol of s was recognized, the end symbol $\dashv$ is read as the current symbol, suppose the current state is q, q will finally transit to the state q_f such that $\dashv$ is consumed. Therefore, $s \in \mathcal{L}(\mathcal{A})$. Then, $\mathcal{L}(r) \subseteq \mathcal{L}(\mathcal{A})$. The FA(&) recognizes the language defined by a $res^{\leq 1}$.

(2) For a regular expression r, an FA(&) recognizing the language $\mathcal{L}(r)$ has at most $\lceil \frac{|\Sigma|-1}{2} \rceil$ start markers (resp. end markers) and at most $|\Sigma|$ concurrent markers.

Proof. For a regular expression r, if an FA(&) recognizes the language $\mathcal{L}(r)$, r is a $res^{\leq 1}$ for the FA(&) recognizing the language defined by a $res^{\leq 1}$ (see the above proof). For the ith subexpression of the form $r_i = r_{i_1} \& r_{i_2} \& \cdots \& r_{i_k}$ $(i, k \in \mathbb{N}, k \geq 2)$ in r, there are start marker $\&_i$ and end marker $\&_i^+$ in the FA(&) for recognizing the strings derived by r_i. For each subexpression r_{i_j} $(1 \leq j \leq k)$ in r_i, there is a concurrent marker $||_{ij}$ in the FA(&) for recognizing the symbols or strings derived by r_{i_j}.

For the subexpression of the form $r_i = r_{i_1} \& r_{i_2} \& \cdots \& r_{i_k}$, there is a start marker $\&_i$ and an end marker $\&_i^+$ in the FA(&). For the regular expression r, the number of binary operators is $|\Sigma| - 1$, suppose that, the number of the operators & is n_2, the number of the other binary operators is n_2'. Then, there is $n_2 + n_2' = |\Sigma| - 1$. In worst case, for the syntax tree T of r, and each node v

in T labelled by &, either the child node or the parent node is labelled by other binary operator. Then, there is $n'_2 = n_2$, the maximum number of the operator & is $\lceil \frac{|\Sigma|-1}{2} \rceil$.

Since each node v in T labelled by &, either the child node or the parent node is labelled by other binary operator, the FA(&) has a corresponding start mark $\&_i$ and a corresponding end mark $\&_i^+$ for the ith operator & in r. Thus, the FA(&) has at most $\lceil \frac{|\Sigma|-1}{2} \rceil$ start markers (resp. end markers).

For each subexpression r_{i_j} in r_i, there is a concurrent marker $||_{ij}$ in the FA(&) for recognizing the symbols or strings derived by r_{i_j}. If the subexpression $r_{i_j} = a \in \Sigma$, then the number of the concurrent markers is k ($k \leq |\Sigma|$). Thus, the maximum number of the concurrent markers in the FA(&) is $|\Sigma|$, The FA(&) has at most $|\Sigma|$ concurrent markers.

7.2 Proof of Theorem 2

We have proven that an FA(&) recognizes the language defined by a regular expression with shuffle, where each alphabet symbol occurs at most once (See the proofs presented above).

(1) The uniform membership problem for FA(&)s is decidable in polynomial time. I.e., for any string s, and an FA(&) $\mathcal{A}$, we can decide whether $s \in \mathcal{L}(\mathcal{A})$ in $\mathcal{O}(|s||\Sigma|^2)$ time.

Proof. An FAS recognizes a string by treating symbols in a string individually. A symbol y in a string s is recognized if and only if the current state p is reached such that $y \in p$. Let p_y denote the state (a set of nodes) p including symbol y. The next symbol of y is read if and only if y has been recognized at the state p_y. H is the node transition graph of an FA(&) $\mathcal{A}$. The number of nodes in H is $\lceil \log_2 |\Sigma| \rceil + 2|\Sigma| + 2$ (including q_0 and q_f) at most. Assume that the current read symbol is y and the current state is q:

1. q is a set: $|q| \geq 1$ and $\exists v \in \{||_{ij}\}_{i \in \mathbb{D}_\Sigma, j \in \mathbb{P}_\Sigma} \cup \Sigma : v \in q \wedge y \in H.\succ(v)$.
   A state (set) q includes $\lceil \log_2 |\Sigma| \rceil + 2|\Sigma|$ nodes at most. For deterministic FAS, it takes $\mathcal{O}(|\Sigma|)$ time to search the node v. Then, the state $p_y = q \backslash \{v\} \cup \{y\}$ can be reached, y is recognized. Thus, for the current state q, it takes $\mathcal{O}(|\Sigma|^2)$ time to recognize y.
2. q is a set: $|q| \geq 1$ and $\exists \&_i \in q : y \in H.R(\&_i)$.
   For FA(&), it takes $\mathcal{O}(|\Sigma|)$ time to search the node $\&_i$ in state (set) q, and it also takes $\mathcal{O}(|\Sigma|)$ time to decide whether $y \in H.R(\&_i)$. Then, the state q transits to the state $q' = q \backslash \{\&_i\} \cup H.\succ(\&_i)$. Then, there is a node $||_{ij}$ ($||_{ij} \in H.\succ(\&_i), j \in \mathbb{P}_\Sigma\}$) in q' that is checked whether $y \in H.\succ(||_{ij})$. Case (1) will be considered. Then, for the current state q, it takes $\mathcal{O}(|\Sigma|^2)$ time to recognize y.
3. q is a set: $|q| \geq 1$ and $\exists \&_i^+ \in q : y \in H.R(\&_i)$.
   The state including the node $\&_i^+$ will transit to the state including the node $\&_i$, case (2) is satisfied. Then, for the current state q, it takes $\mathcal{O}(|\Sigma|^2)$ time to recognize y.

4. $q = q_0$.
   If $y \in H.\succ(q_0)$, then, for FA(&), it takes $\mathcal{O}(|\Sigma|)$ time to search the state including the node y. Otherwise, a node $\&_i$ ($i \in \mathbb{D}_\Sigma$) is searched and then is decided whether $y \in H.R(\&_i)$. Then, it takes $\mathcal{O}(|\Sigma|^2)$ time for q to transit to the state $\{\&_i\}$. Case (2) is satisfied. Then, for the current state q, it takes $\mathcal{O}(|\Sigma|^2)$ time at most to recognize y.

Thus, for FA(&), a symbol $y \in \Sigma_s$ and a current state q, it takes $\mathcal{O}(|\Sigma|^2)$ time at most to recognize y. When the last symbol of s was recognized, the end symbol $\dashv$ requires to be consumed, it takes $\mathcal{O}(|H.V|) = \mathcal{O}(|\Sigma|)$ time to transit to the final state q_f. Let $|s|$ denote the length of the string s, then for an FA(&), it takes $\mathcal{O}(|s||\Sigma|^2)$ time to recognize s. Therefore, the uniform membership problem for FA(&)s is decidable in polynomial time.

(2) The non-uniform membership problem for FA(&)s is also decidable in polynomial time.

Proof. The non-uniform version of the membership problem for FA(&)s specifies that, the language is fixed, only the string to be tested is considered as input. This indicates that $|\Sigma|$ is a constant. An FA(&) recognizes the language defined by a $red^{\leq 1}$, for a $red^{\leq 1}$ r, r can be in polynomial time transformed to the equivalent FA (&) $\mathcal{A}$ by using Thompson's construction[6]. For a string $s \in \mathcal{L}(r)$, $\mathcal{A}$ accepts or rejects s in $\mathcal{O}(|s||\Sigma|^2)$ time (linear time). Thus, the non-uniform membership problem for FA(&)s is decidable in polynomial time.

Therefore, both the uniform and the non-uniform membership problem for FA(&)s are solvable in polynomial time.

7.3 Proof of Theorem 3

Proof. For any tuple $(a, b) \in U_\&$, the node a connects with the node b in the undigraph $F(V, E)$ ($F.E = U_\&$). The nodes a and b are in a connected component of F. According to the algorithm *ShuffleUnits*, for each connected component f of F, there is a corresponding shuffle unit.

First, the non-adjacent nodes, which are selected from f, compose a set M_f such that the sum of all node degrees is maximum. M_f is one of the sets in a shuffle unit. Then, if one of the nodes a and b occurs in M_f (a and b cannot occur in M_f at the same time), after removing the nodes in f and their associated edges, a new undigraph f' is obtained. If f' is not a connected graph, $[M_f, f'.V]$ forms a shuffle unit, the other node occurs in $f'.V$. Otherwise, M_f is stored in a shuffle unit, algorithm *ShuffleUnits* recursively works on f', the other node must occur in another obtained set.

If neither a nor b occurs in M_f, after removing the nodes in f and their associated edges, a new undigraph f' is obtained, algorithm *ShuffleUnits* recursively

[6]Thompson, K. (1968). Programming techniques: Regular expression search algorithm. Communications of the ACM, 11(6), 419-422.

works on f'. In extreme case, $f'.V=\{a,b\}$ and $f'.E=\{(a,b)\}$, then $M_{f'}=\{a\}$, $[\{a\},\{b\}]$ forms a shuffle unit. The nodes a and b occur in different sets.

All obtained shuffle units are put into $P_{\&}$, thus, for any tuple $(a,b)\in U_{\&}$, there exists a shuffle unit $l\in P_{\&}$ such that a and b are in different sets in l.

7.4 Proof of Theorem 4

(1) We use Lemma 1 to prove the Theorem 4.

Lemma 1. *Let $P_{\&}=\{[e_1,e_2,\cdots,e_k]\}$ $(k\geq 2)$, and let $r(e_i)$ $(1\leq i\leq k)$ denote a regular expression such that $e_i=\Sigma_{r(e_i)}$. Assume that the set $P_{\&}$ of shuffle units is returned by algorithm ShuffleUnits. For a given finite sample S, and a shuffle unit $l'=[e'_1,e'_2,\cdots,e'_t]$ $(t\geq 2)$, if there exists $r(e_i)$ and $r'(e'_j)$ $(1\leq j\leq t)$: $\mathcal{L}(r(e_1)\&\cdots\&r(e_k))\supset\mathcal{L}(r'(e'_1)\&\cdots\ \&r'(e'_t))\supseteq S$, then $t=k$ and $e_i=e'_i$.*

Proof. For $\mathcal{L}(r(e_1)\&\cdots\&r(e_k))\supset\mathcal{L}(r'(e'_1)\&\cdots\&r'(e'_t))\supseteq S$[7], there is $t\leq k$.

For any two distinct symbols $u,v\in\Sigma$, if u is necessarily interleaved with v for S, according to Theorem 3, u and v are in two distinct sets in l'. Otherwise, it will lead to $\mathcal{L}(r'(e'_1)\&\cdots\&r'(e'_t))\not\supseteq S$. Then, according to the algorithm *ShuffleUnits*, each connected component of the undigraph $F(V,E)$, where $F.E=U_{\&}$, forms a shuffle unit, then $k\leq t$. Thus, there is $t=k$.

Let $\mathcal{L}(r(e_1)\&\cdots\&r(e_k))\supset\mathcal{L}(r'(e'_1)\&\cdots\&r'(e'_k))\supseteq S$. If there exists $1\leq i\leq k$ such that $e_i\neq e'_i$, then $r(e_i)\neq r'(e'_i)$, there exists a string s' such that $s'\in\mathcal{L}(r'(e'_1)\&\cdots\&r'(e'_k))$ but $s'\notin\mathcal{L}(r(e_1)\&\cdots\&r(e_k))$. Then, $\mathcal{L}(r(e_1)\&\cdots\&r(e_k))\not\supset\mathcal{L}(r'(e'_1)\&\cdots\ \&r'(e'_k))$. Therefore, $e_i=e'_i$ for any $1\leq i\leq k$.

(2) There does not exist an FAS $\mathcal{A}'$, which is learned from S such that $\mathcal{L}(\mathcal{A})\supset\mathcal{L}(\mathcal{A}')\supseteq S$. The FAS $\mathcal{A}$ is a precise representation of S.

Proof. The FAS $\mathcal{A}$ is learned by constructing the corresponding node transition graph H. We convert the SOA G built for S to the digraph H by traversing shuffle units in $P_{\&}$, which is obtained from Algorithm 2. The built SOA G is a precise representation of S [9].

Assume that there exists an FAS $\mathcal{A}'$ learned from S such that $\mathcal{L}(\mathcal{A})\supset\mathcal{L}(\mathcal{A}')\supseteq S$. For the node transition graph H' of the FAS $\mathcal{A}'$, H' should be constructed from the SOA G built for S, otherwise, the above assumption can not hold. Suppose that there is the set $P'_{\&}$ of shuffle units such that the digraph H' can be constructed from the SOA G by traversing shuffle units in $P'_{\&}$.

For each shuffle unit $l\in P_{\&}$, let $l=[e_1,\cdots,e_k]$ $(k\geq 2)$, according to Algorithm 3, there are corresponding start marker $\&_m$ and end marker $\&_m^+$ $(m\in\mathbb{D}_\Sigma)$ are added into G. Let $\mathcal{B}$ denote the constructed FAS. The FAS $\mathcal{B}$ can recognize the shuffled strings which consist of the symbols in $\bigcup_{1\leq i\leq k}e_i$. Let $S_{\&}$ denote the set of the above shuffled strings extracted from S.

Then, for constructing FAS $\mathcal{A}'$, for each shuffle unit $l'\in P'_{\&}$ and $l'=[e'_1,\cdots,e'_t]$ $(t\geq 2)$, we obtain the currently constructed FAS $\mathcal{B}'$ by adding the corresponding

[7] For simplicity of proof, let $r(e_1)\&\cdots\&r(e_k)$ denote that there exists $r(e_i)$ such that $r(e_1)\&\cdots\&r(e_k)$ is a regular expression supporting shuffle.

start marker $\&_n$ and end marker $\&_n^+$ $(n \in \mathbb{D}_\Sigma)$ into G. If $\mathcal{L}(\mathcal{B}) \supset \mathcal{L}(\mathcal{B}') \supseteq S_\&$, then there exists $r(e_i)$ and $r'(e'_j)$ $(1 \leq j \leq t)$ such that $\mathcal{L}(r(e_1)\& \cdots \& r(e_k)) \supset \mathcal{L}(r'(e'_1)\& \cdots \& r'(e'_t)) \supseteq S_\&$ (Let $\mathcal{L}(\mathcal{B}) = \mathcal{L}(r(e_1)\& \cdots \& r(e_k))$ and $\mathcal{L}(\mathcal{B}') = \mathcal{L}(r'(e'_1)\& \cdots \& r'(e'_t))$.). According to Lemma 1, there are $t = k$ and $e_i = e'_i$, then there is $l = l'$.

This implies that, if $\mathcal{L}(\mathcal{A}) \supset \mathcal{L}(\mathcal{A}') \supseteq S$, there is $P_\& = P'_\&$. For digraphs H and H', they are both constructed from the SOA G, then there is $\mathcal{A} = \mathcal{A}'$ and $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}') \supseteq S$. There is a contraction to the initial assumption. Therefore, the initial assumption does not hold, the FAS $\mathcal{A}$ is a precise representation of S.