

CPSC-406 Report

Nathan Carnnahan
Chapman University

February 22, 2026

Abstract

Contents

1	Introduction	2
2	Introduction	2
3	Week by Week	2
3.1	Week 1	2
3.2	Week 2	3
4	Synthesis	6
5	Evidence of Participation	6
6	Conclusion	6

1 Introduction

2 Introduction

3 Week by Week

3.1 Week 1

HW1 – DFA Exercises

Exercise 1 We are given two DFAs A_1 and A_2 .

Accepted Words Table

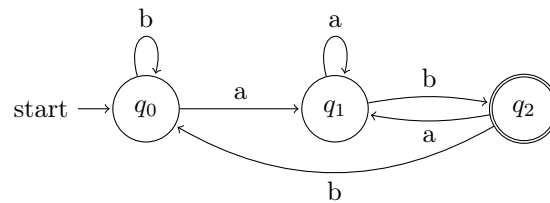
w	Accepted by A_1 ?	Accepted by A_2 ?
aaa	No	Yes
aab	Yes	No
aba	No	No
abb	No	No
baa	No	Yes
bab	No	No
bba	No	No
bbb	No	No

Language Descriptions

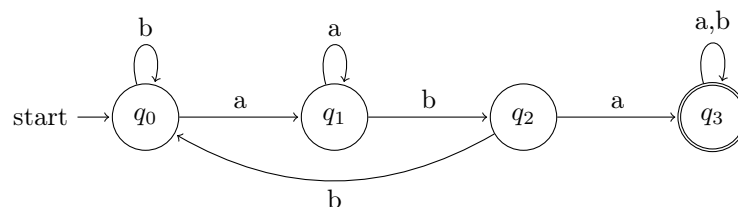
- $L(A_1)$: all strings over $\{a, b\}$ that start with a and end with an odd number of b 's.
- $L(A_2)$: all strings over $\{a, b\}$ that end with at least two consecutive a 's.

Exercise 2 – Designing DFAs

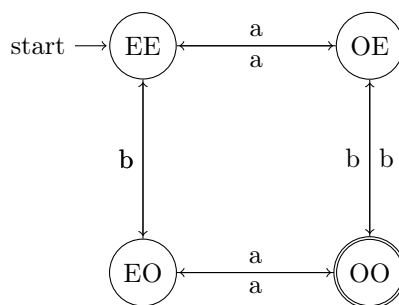
1. Words that end with ab



2. Words that contain aba

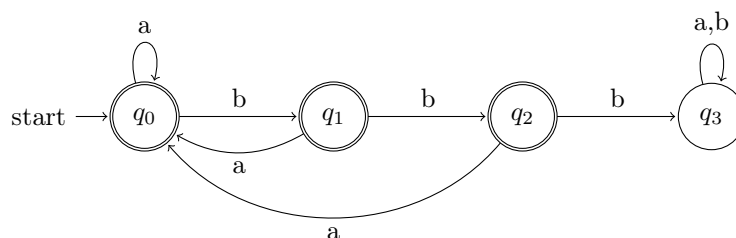


3. **Odd number of a 's and odd number of b 's** States represent parity: (a -parity, b -parity).

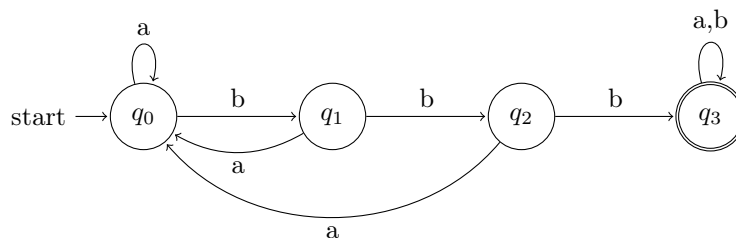


4. **Even number of a 's and odd number of b 's** Same automaton as above, but accepting state is EO.

5. **Any three consecutive characters contain at least one a** Equivalent to forbidding substring bbb .



6. **Words that contain bbb**



Observation

- Problems 3 and 4 use the same parity structure; only the accepting state changes.
- Problems 5 and 6 use the same “count consecutive b 's” structure; one treats reaching three b 's as rejection, the other as acceptance.
- Problems 1 and 2 track progress toward matching a pattern.

3.2 Week 2

HW2 – Operations on automata

Exercise 1 (Product automata). Throughout, $\Sigma = \{a, b\}$.

1. **Description of $L(A^{(1)})$ and $L(A^{(2)})$.**

$A^{(1)}$:

The accepting states are $\{2, 4\}$ and state 3 is a non-accepting sink (it loops on both a and b). From state 2, reading a goes to the sink; from state 4, reading b goes to the sink. Thus, once the first symbol is read, the letters must alternate in order to avoid the sink.

Therefore,

$$L(A^{(1)}) = \{w \in \{a, b\}^* : |w| \geq 1 \text{ and } w \text{ contains no substring } aa \text{ or } bb\}.$$

Equivalently,

$$L(A^{(1)}) = a(ba)^*(\varepsilon \mid b) \cup b(ab)^*(\varepsilon \mid a).$$

$A^{(2)}$:

The only accepting state is 2. From state 1, reading b leads to a sink. Reading a moves $1 \rightarrow 2$, and from 2 any symbol returns to 1.

Thus a word must:

- start with a ,
- have odd length,
- have a in every odd position.

Hence,

$$L(A^{(2)}) = a((a \mid b)a)^*.$$

2. Intersection automaton $A = A^{(1)} \times A^{(2)}$.

- States: $Q = Q_1 \times Q_2$
- Start state: $(1, 1)$
- Accepting states:

$$F = F_1 \times F_2 = \{2, 4\} \times \{2\}$$

- Transition function:

$$\delta((p, q), x) = (\delta_1(p, x), \delta_2(q, x))$$

The reachable accepting state is $(2, 2)$.

3. Proof that $L(A) = L(A^{(1)}) \cap L(A^{(2)})$.

For any word w , by induction on prefixes, after reading w the product automaton is in state

$$(\delta_1(1, w), \delta_2(1, w)).$$

Thus A accepts w iff both component automata accept w . Therefore,

$$L(A) = L(A^{(1)}) \cap L(A^{(2)}).$$

4. Construction of union automaton A' .

Keep the same product states and transitions, but define the accepting set as

$$F' = (F_1 \times Q_2) \cup (Q_1 \times F_2).$$

Then A' accepts whenever at least one component accepts, so

$$L(A') = L(A^{(1)}) \cup L(A^{(2)}).$$

Exercise 2 (More automata). 1. Description of $L(B^{(1)})$ and $L(B^{(2)})$.

$B^{(1)}$:

State p_0 is accepting. Reading a cycles

$$p_0 \rightarrow p_1 \rightarrow p_2 \rightarrow p_0,$$

while b loops at each state.

Thus $B^{(1)}$ accepts exactly when the number of a 's is divisible by 3:

$$L(B^{(1)}) = \{ w \in \{a, b\}^* : \#_a(w) \equiv 0 \pmod{3} \}.$$

$B^{(2)}$:

State q_0 is accepting. The automaton goes to a sink q_2 if the substring aa occurs. Also, q_1 is non-accepting, so a word ending in a is rejected.

Hence

$$L(B^{(2)}) = \{ w \in \{a, b\}^* : w \text{ has no substring } aa \text{ and does not end with } a \}.$$

Equivalently,

$$L(B^{(2)}) = (b \mid ab)^*.$$

2. Intersection automaton $B = B^{(1)} \times B^{(2)}$.

- Start state: (p_0, q_0)
- Accepting states:

$$F = \{(p_0, q_0)\}$$

- Transition function:

$$\delta((p_i, q_j), x) = (\delta_1(p_i, x), \delta_2(q_j, x))$$

3. Proof that $L(B) = L(B^{(1)}) \cap L(B^{(2)})$.

After reading any word w , the product automaton is in state

$$(\delta_1(p_0, w), \delta_2(q_0, w)).$$

Thus B accepts w iff both component automata accept w . Therefore,

$$L(B) = L(B^{(1)}) \cap L(B^{(2)}).$$

4. Construction of union automaton B' .

Using De Morgan's law:

$$L(B^{(1)}) \cup L(B^{(2)}) = \overline{\overline{L(B^{(1)})} \cap \overline{L(B^{(2)})}}.$$

1. Complement $B^{(1)}$ and $B^{(2)}$ by swapping accepting and non-accepting states.
2. Construct their product automaton for intersection.
3. Complement the resulting automaton.

This yields B' such that

$$L(B') = L(B^{(1)}) \cup L(B^{(2)}).$$

Exercise 2.2.7 Let A be a DFA and let q be a state such that $\delta(q, a) = q$ for all input symbols $a \in \Sigma$. We prove by induction on $|w|$ that for all strings $w \in \Sigma^*$, $\hat{\delta}(q, w) = q$.

Proof. Base case: $|w| = 0$, so $w = \varepsilon$. By definition of $\hat{\delta}$, $\hat{\delta}(q, \varepsilon) = q$.

Inductive step: Assume that for some $n \geq 0$, for every string $x \in \Sigma^*$ with $|x| = n$ we have $\hat{\delta}(q, x) = q$. Let $w \in \Sigma^*$ be any string with $|w| = n + 1$. Then w can be written as $w = xa$ where $|x| = n$ and $a \in \Sigma$ is the last symbol. Using the recursive definition of $\hat{\delta}$,

$$\hat{\delta}(q, xa) = \delta(\hat{\delta}(q, x), a).$$

By the inductive hypothesis, $\hat{\delta}(q, x) = q$, so

$$\hat{\delta}(q, xa) = \delta(q, a) = q,$$

where the last equality uses the assumption that $\delta(q, a) = q$ for all $a \in \Sigma$. Thus $\hat{\delta}(q, w) = q$ for all w of length $n + 1$.

By induction, $\hat{\delta}(q, w) = q$ for all $w \in \Sigma^*$. □

Interesting Question for Discord (from §2.2.4) Can the same automaton be setup to accept multiple languages? If so, can we use the same automaton hardware to create a multifunctional dfa?

4 Synthesis

5 Evidence of Participation

6 Conclusion

References

[BLA] Nathan Carnnahan, [CPSC-406 Report](#), Nathan Carnnahan, 2026.