Part V.
Properties of Regular Languages

Pumping Lemma for RLs

Gist: Pumping lemma demonstrates an infinite iteration of some substring in RLs.

- Let L be a RL. Then, there is $k \geq 1$ such that **if** $z \in L$ and $|z| \geq k$, **then** there exist u,v,w: $z = uvw$, **1)** $v \neq \varepsilon$ **2)** $|uv| \leq k$ **3)** for each $m \geq 0$, $uv^m w \in L$

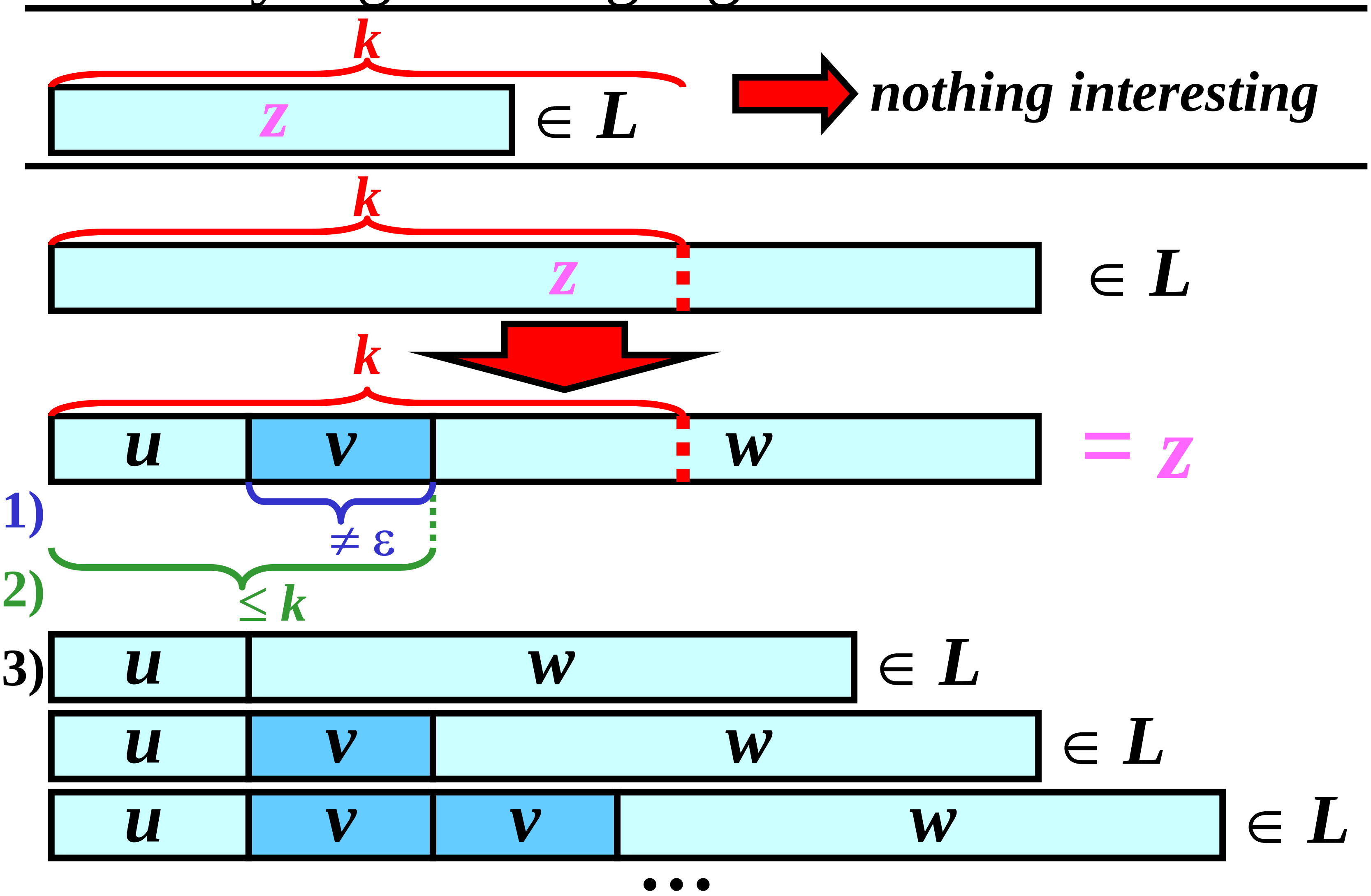
Example: for RE $r = ab^*c$, $L(r)$ is ***regular.***
There is $k = 3$ such that **1), 2)** and **3)** holds.

- for $z = abc$: $z \in L(r)$ & $|z| \geq 3$: $uv^0w = ab^0c = ac \in L(r)$
  $u = a$, $v = b$, $w = c$
  $uv^1w = ab^1c = abc \in L(r)$
  $uv^2w = ab^2c = abbc \in L(r)$
  ⋮
  $v \neq \varepsilon$, $|uv| = 2 \leq 3$
- for $z = abbc$: $z \in L(r)$ & $|z| \geq 3$: $uv^0w = abb^0c = abc \in L(r)$
  $u = a$, $v = b$, $w = bc$
  $uv^1w = abb^1c = abbc \in L(r)$
  $uv^2w = abb^2c = abbbc \in L(r)$
  ⋮
  $v \neq \varepsilon$, $|uv| = 2 \leq 3$

⋮

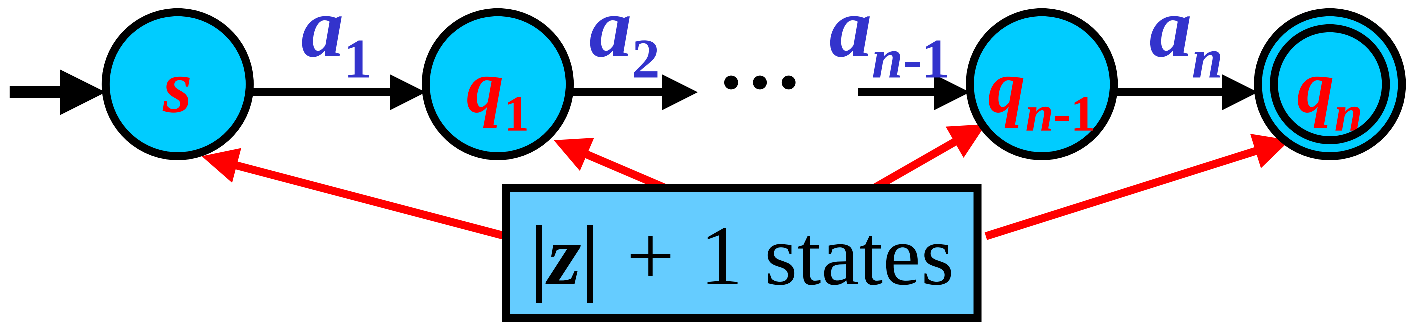
Pumping Lemma: Illustration

- L = any regular language:



Proof of Pumping Lemma 1/3

- Let L be a regular language. Then, there exists **DFA** $M = (Q, \Sigma, R, s, F)$, and $L = L(M)$.
- For $z \in L(M)$, M makes $|z|$ moves and M visits $|z| + 1$ states:
- for $z = a_1a_2\ldots a_n$:



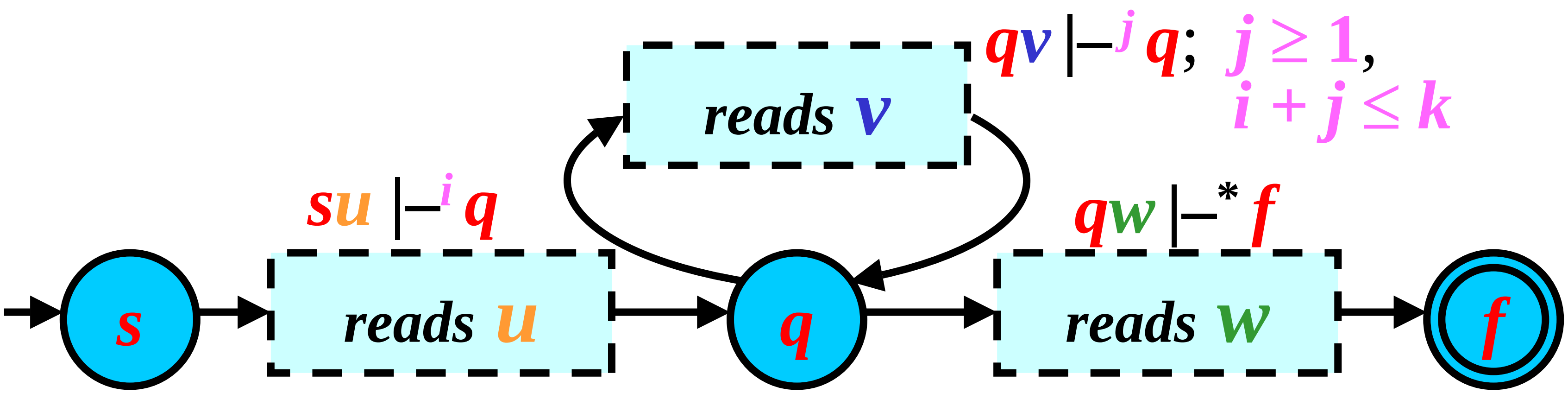
$$s\overbrace{a_1a_2\ldots a_n}^{|z|} \vdash q_1a_2\ldots a_n \vdash \ldots \vdash q_{n-1}a_n \vdash q_n$$

Proof of Pumping Lemma 2/3

- Let $\boldsymbol{k}$ = card(Q) (the number of states).

For each $z \in L$ and $|z| \geq k$, M visits $k + 1$ or more states. As $k + 1 >$ card(Q), there exists a state $\boldsymbol{q}$ that M visits at least twice.

- For z exist u, v, w such that $z = \boldsymbol{uvw}$:



Summary:

$$sz = suvw \vdash^i qvw \vdash^j qw \vdash^* f,\ f \in F$$

Proof of Pumping Lemma 3/3

- There exist moves:

(1.) $su \vdash^{i} q$; (2.) $qv \vdash^{j} q$; (3.) $qw \vdash^{*} f$, $f \in F$, so

- for $m = 0$, $uv^{m}w = uv^{0}w = uw$,

$$suw \overset{(1.)}{\vdash^{i}} qw \overset{(3.)}{\vdash^{*}} f,\ f \in F$$

- for each $m > 0$,

$$suv^{m}w \overset{(1.)}{\vdash^{i}} qv^{m}w \overset{(2.)}{\vdash^{j}} qv^{m-1}w \overset{(2.)}{\vdash^{j}} \dots \overset{(2.)}{\vdash^{j}} qw \overset{(3.)}{\vdash^{*}} f,\ f \in F$$

Summary:

1) $qv \vdash^{j} q$, $j \geq 1$; therefore, $|v| \geq \mathbf{1}$, so $v \neq \varepsilon$

2) $suv \vdash^{i} qv \vdash^{j} q$, $i + j \leq k$; therefore, $|uv| \leq k$

3) For each $m \geq 0$: $suv^{m}w \vdash^{*} f$, $f \in F$, therefore $uv^{m}w \in L$

QED

Pumping Lemma: Application I

- Based on the pumping lemma, we often make a proof by contradiction to demonstrate that a language is **not** regular

Assume that L is regular

Consider the PL constant $\boldsymbol{k}$ and select $\boldsymbol{z} \in L$, whose length depends on $\boldsymbol{k}$ so $|\boldsymbol{z}| \geq \boldsymbol{k}$ is surely true.

For all decompositions of $\boldsymbol{z}$ into $\boldsymbol{uvw}$, $v \neq \varepsilon$, $|uv| \leq k$, show:
there exists $m \geq 0$ such that $\boldsymbol{uv^m w} \notin \boldsymbol{L}$
from the pumping lemma, $\boldsymbol{uv^m w} \in \boldsymbol{L}$
} **contradiction**

false assumption

Therefore,
L **is not regular**

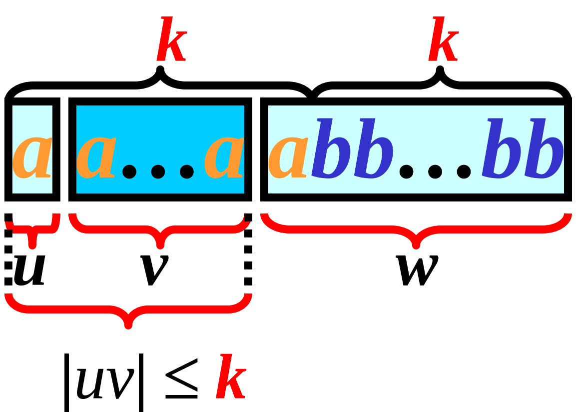
Pumping Lemma: Example

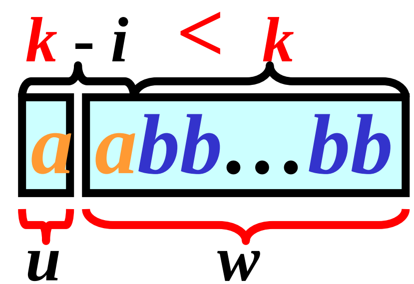
Prove that $L = \{a^n b^n : n \geq 0\}$ is not regular:

1) Assume that L is regular. Let $\boldsymbol{k} \geq 1$ be the pumping lemma constant for L.

2) Let $\boldsymbol{z} = \boldsymbol{a^k b^k}$: $\boldsymbol{a^k b^k} \in L$, $|\boldsymbol{z}| = |\boldsymbol{a^k b^k}| = 2\boldsymbol{k} \geq \boldsymbol{k}$

3) All decompositions of $\boldsymbol{z}$ into $\boldsymbol{uvw}$, $v \neq \varepsilon$, $|uv| \leq \boldsymbol{k}$:



- pumping lemma: $uv^0w \in L$
- $uv^0w = uw =$  $\notin L$

Contradiction!

4) Therefore, L is not regular

Note on Use of Pumping Lemma

- **Pumping lemma:**

if *L* is regular **then** ⇒ exist $k \geq 0$ and ...

Main application of the pumping lemma:

- proof by contradiction that *L* is **not** regular.

- **However, the next implication is incorrect:**

~~**if** exist $k > 0$ and ... **then** ⇒ *L* is regular~~

- **We cannot use the pumping lemma to prove that *L* is regular.**

Pumping Lemma: Application II. 1/3

- We can use the pumping lemma to prove some other theorems.

Illustration:

- Let M be a DFA and k be the pumping lemma constant (k is the number of states in M). Then, $L(M)$ is infinite $\Leftrightarrow$ there exists $z \in L(M)$, $k \leq |z| < 2k$

Proof:

1) there exists $z \in L(M)$, $k \leq |z| < 2k \Rightarrow L(M)$ is infinite**:**

if $z \in L(M)$, $k \leq |z|$, then by PL:

$z = uvw$, $v \neq \varepsilon$, and for each $m \geq 0$: $uv^m w \in L(M)$

$L(M)$ is infinite

Pumping Lemma: Application II. 2/3

2) $L(M)$ is infinite $\Rightarrow$ there exists $z \in L(M)$, $k \leq |z| < 2k$**:**

- We prove by contradiction, that

$L(M)$ is infinite —**a)**→ there exists $z \in L(M)$, $|z| \geq k$ —**b)**→ there exists $z \in L(M)$, $k \leq |z| < 2k$

---

a) Prove by contradiction that

- **$L(M)$ is infinite $\Rightarrow$ there exists $z \in L(M)$, $|z| \geq k$**

Assume that **$L(M)$ is infinite** and there exists no $z \in L(M)$, $|z| \geq k$

↓

for all $z \in L(M)$ holds $|z| < k$

↓

$L(M)$ is finite

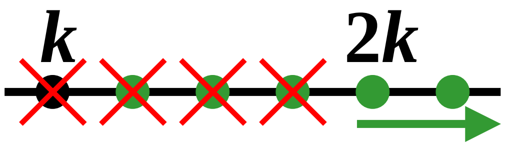
Contradiction !

Pumping Lemma: Application II. 3/3

b) Prove by contradiction

- **there exists $z \in L(M)$, $|z| \geq k \Rightarrow$ there exists $z \in L(M)$, $k \leq |z| < 2k$**

Assume that **there is $z \in L(M)$, $|z| \geq k$** and **there is no $z \in L(M)$, $k \leq |z| < 2k$**



Let z_0 be **the shortest string** satisfying $z_0 \in L(M)$, $|z_0| \geq k$

Because there exists no $z \in L(M)$, $k \leq |z| < 2k$, so $|z_0| \geq 2k$

If $z_0 \in L(M)$ and $|z_0| \geq k$, the PL implies: $z_0 = uvw$, $|uv| \leq k$, and for each $m \geq 0$, $uv^mw \in L(M)$

for $m = 0$: $uv^mw = uw \in L(M)$

$$|uw| = \underbrace{|z_0|}_{\geq 2k} - \underbrace{|v|}_{\leq k} \geq k$$

Summary: $uw \in L(M)$, $|uw| \geq k$ and $|uw| < |z_0|$**!**

z_0 is not the shortest string satisfying $z_0 \in L(M)$, $|z_0| \geq k$

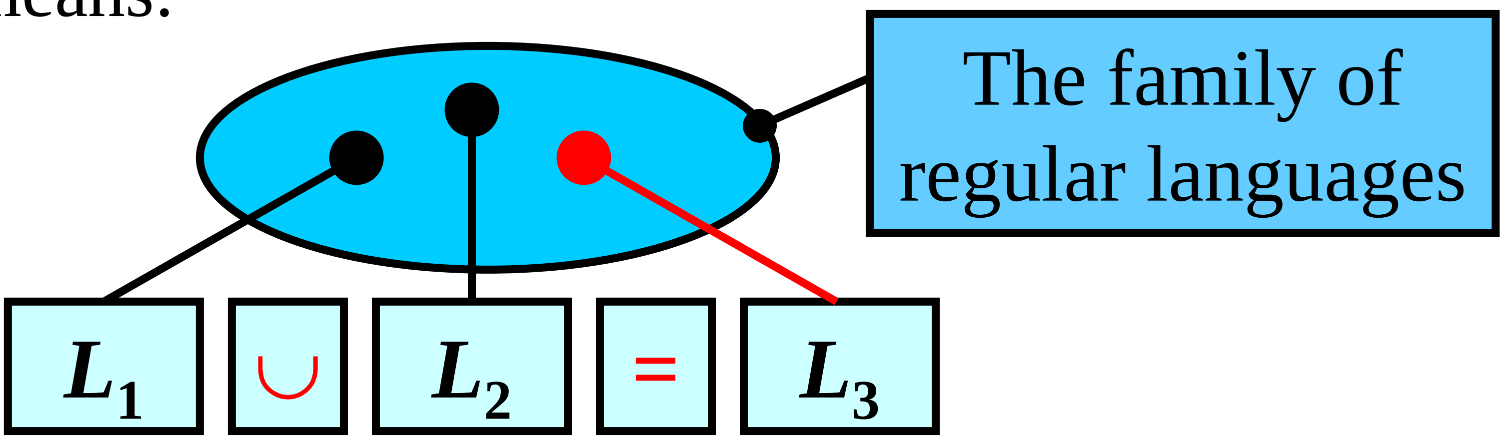
Contradiction !

Closure properties 1/2

Definition: The family of regular languages is closed under an operation ***o*** if the language resulting from the application of ***o*** to any regular languages is also regular.

Illustration:

- The family of regular languages is closed under ***union***. It means:



Closure properties 2/2

Theorem: The family of regular languages is closed under **union**, **concatenation**, **iteration**.

Proof:

- Let $\boldsymbol{L_1}$, $\boldsymbol{L_2}$ be two **regular languages**
- Then, there exist two REs r_1, r_2: $L(r_1) = \boldsymbol{L_1}$, $L(r_2) = \boldsymbol{L_2}$;
- By the definition of regular expressions:
  - $r_1.r_2$ is a RE denoting $\boldsymbol{L_1 L_2}$
  - $r_1 + r_2$ is a RE denoting $\boldsymbol{L_1 \cup L_2}$
  - r_1^* is a RE denoting $\boldsymbol{L_1^*}$
- Every RE denotes regular language, so $\boldsymbol{L_1 L_2}$, $\boldsymbol{L_1 \cup L_2}$, $\boldsymbol{L_1^*}$ are a **regular languages**

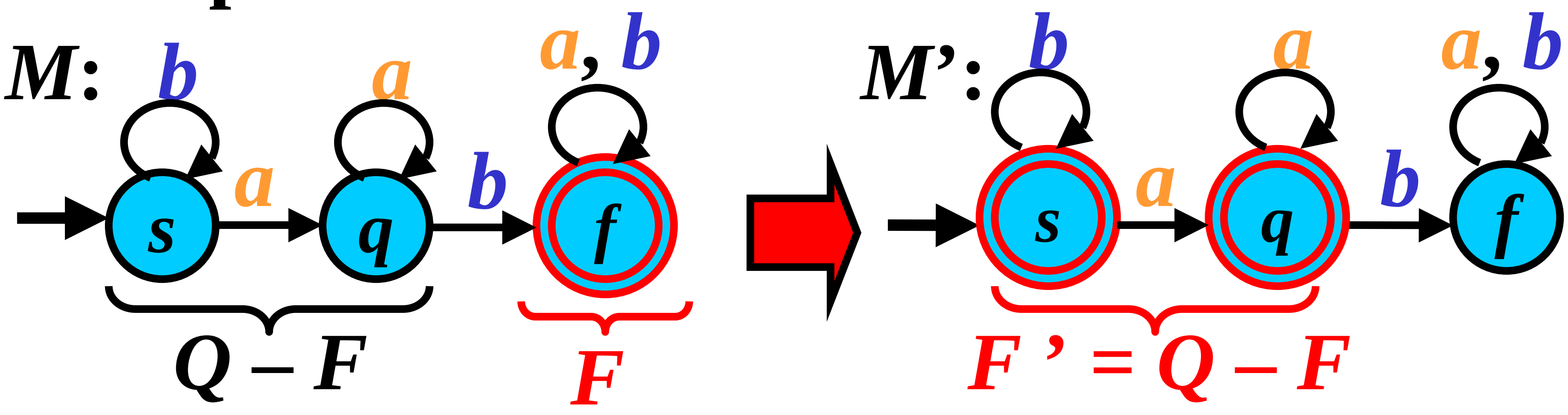
Algorithm: FA for Complement

- **Input:** Complete FA: $M = (Q, \Sigma, R, s, F)$
- **Output:** Complete FA: $M' = (Q, \Sigma, R, s, F')$,

$$L(M') = \overline{L(M)}$$

- **Method:**
- $F' := Q - F$

Example:



$L(M) = \{x$: *ab* is a *substring* of $x\}$; $L(M') = \{x$: *ab* is no *substring* of $x\}$

FA for Complement: Problem

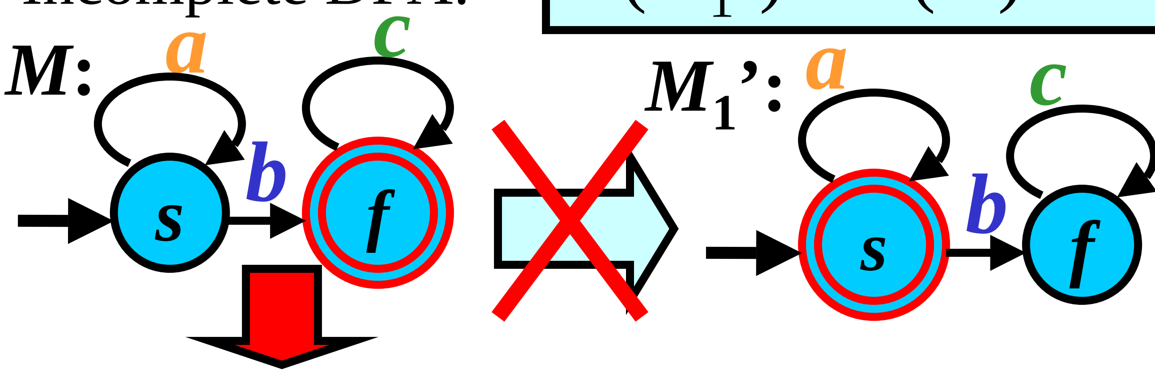
- Previous algorithm requires a **complete** FA
- If M is incomplete FA, then M must be converted to a complete FA before we use the previous algorithm

---

Example:

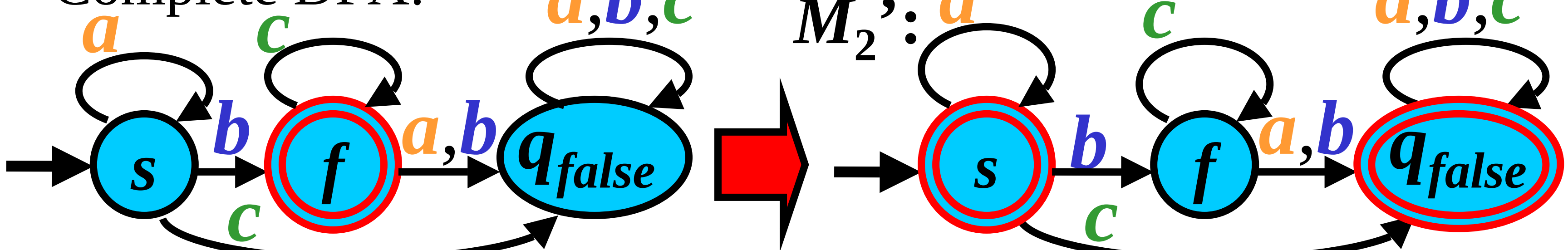
Incomplete DFA:

$L(M_1') \neq \overline{L(M)}$**!** - $c \notin L(M)$, $c \notin L(M_1')$



$L(M_2') = \overline{L(M)}$

Complete DFA:



Closure properties: Complement

Theorem: The family of regular languages is closed under **complement**.

Proof:

- Let $\boldsymbol{L}$ be a **regular language**
- Then, there exists a complete DFA M: $L(M) = \boldsymbol{L}$
- We can construct a complete DFA M': $L(M') = \overline{\boldsymbol{L}}$ by using the previous algorithm
- Every FA defines a regular language, so $\overline{\boldsymbol{L}}$ is a **regular language**

Closure properties: Intersection

Theorem: The family of regular languages is closed under **intersection**.

Proof:

- Let L_1, L_2 be two **regular languages**
- $\overline{L_1}$, $\overline{L_2}$ are **regular languages**

(the family of regular languages is closed under complement)

- $\overline{L_1} \cup \overline{L_2}$ is a **regular language**

(the family of regular languages is closed under union)

- $\overline{\overline{L_1} \cup \overline{L_2}}$ is a **regular language**

(the family of regular languages is closed under complement)

- $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$ is a **regular language** (DeMorgan's law)

Boolean Algebra of Languages

Definition: Let a family of languages be closed under union, intersection, and complement. Then, this family represents a ***Boolean algebra of languages***.

Theorem: The family of regular languages is a Boolean algebra of languages.

Proof:

- The family of regular languages is closed under union, intersection, and complement.

Main Decidable Problems

1. Membership problem:

- **Instance:** FA M, $w \in \Sigma^*$; **Question:** $w \in L(M)$**?**

2. Emptiness problem:

- **Instance:** FA M; **Question:** $L(M) = \emptyset$**?**

3. Finiteness problem:

- **Instance:** FA M; **Question:** Is $L(M)$ finite**?**

4. Equivalence problem:

- **Instance:** FA M_1, M_2; **Question:** $L(M_1) = L(M_2)$**?**

Algorithm: Membership Problem

- **Input:** DFA $M = (Q, \Sigma, R, s, F)$; $w \in \Sigma^*$
- **Output:** **YES** if $w \in L(M)$

  NO if $w \notin L(M)$

- **Method:**
- **if** $sw \vdash^* f$, $f \in F$ **then** write ('**YES**')

  else write ('**NO**')

Summary:

The membership problem for FAs is decidable

Algorithm: Emptiness Problem

- **Input:** FA $M = (Q, \Sigma, R, s, F)$;
- **Output:** **YES** if $L(M) = \emptyset$

  NO if $L(M) \neq \emptyset$

- **Method:**
- **if** *s* is nonterminating **then** write ('**YES**')

  else write ('**NO**')

Summary:

The emptiness problem for FAs is decidable

Algorithm: Finiteness Problem

- **Input:** DFA $M = (Q, \Sigma, R, s, F)$;
- **Output:** **YES** if $L(M)$ is finite
  NO if $L(M)$ is infinite

- **Method:**
- Let k = card(Q)
- **if** there exist $z \in L(M)$, $k \leq |z| < 2k$ **then** write ('**NO**')
  else write ('**YES**')

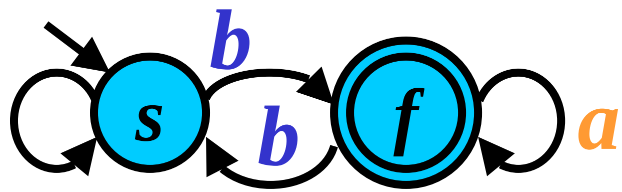
Note: This algorithm is based on
$L(M)$ is infinite $\Leftrightarrow$ there exists z: $z \in L(M)$, $k \leq |z| < 2k$

Summary:

The finiteness problem for FAs is decidable

Decidable Problems: Example

***M*:**



Question: ***ab*** $\in L(M)$ **?**

sab |– ***sb*** |– ***f***, ***f*** $\in$ ***F***

Answer: YES because ***sab*** $\vdash^*$ ***f***, ***f*** $\in$ ***F***

Question: $L(M) = \emptyset$ **?**

$Q_0 = \{f\}$

1. $qa' \rightarrow f$; $q \in Q$; $a' \in \Sigma$: $sb \rightarrow f$, $fa \rightarrow f$

$Q_1 = \{f\} \cup \{s, f\} = \{f, s\}$ … ***s*** is terminating

Answer: NO because ***s*** is terminating

Question: Is $L(M)$ finite?

$k = \text{card}(Q) = 2$

All strings $z \in \Sigma^*$: $2 \leq |z| < 4$: ***aa***, ***bb***, ***ab*** $\in$ ***L*(*M*)**, …

Answer: NO because there exist $z \in L(M)$, $k \leq |z| < 2k$

Algorithm: Equivalence Problem

- **Input:** Two minimum state FA, $M_1$and M_2
- **Output:** **YES** if $L(M_1) = L(M_2)$

  NO if $L(M_1) \neq L(M_2)$

- **Method:**
- **if** M_1 coincides with M_2 except for the name of states

  then write ('**YES**')

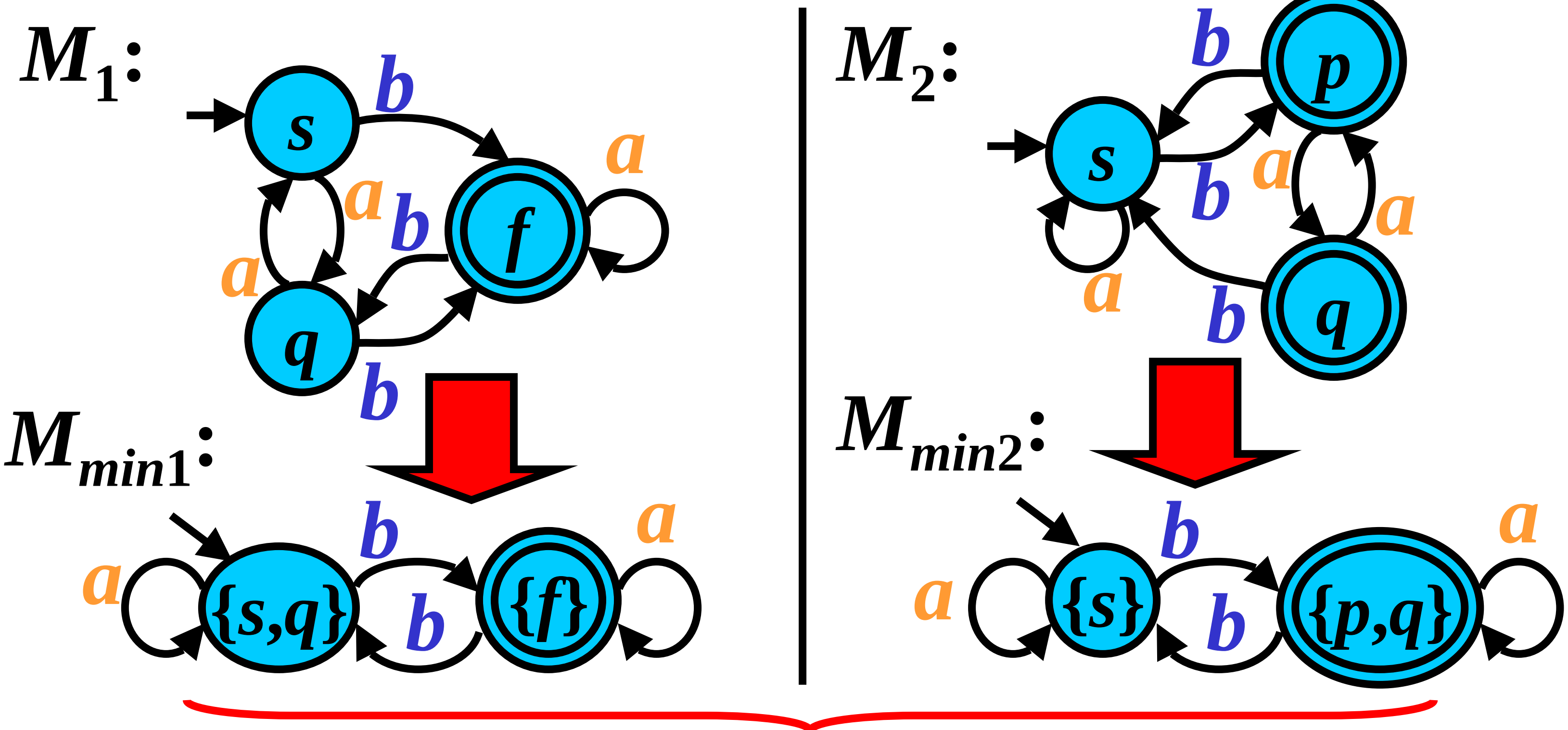
  else write ('**NO**')

Summary:

The equivalence problem for FA is decidable

Equivalence Problem: Example

Question: $L(M_1) = L(M_2)$?



A minimum state FA

Answer: **YES** because M_{min1} coincides with M_{min2}