then m(S,0) is the 'next' state.
Current state S, and current input character a (a \in \Sigma)
m is state transition function with two variables,
S is a finite set of states

system \( M = (S,\Sigma, m, s_0, F) \) where

Formal definition: A FSM over the alphabet \( \Sigma \) is a

Example: 00, 01, 11, 10 are states.

number of internal configurations or states
The system may be in any one of a finite

of a system with discrete inputs and

A finite state machine is a mathematical model

State diagram.
The diagram is called

100, 110 and called words.

\( F = \{ 111 \} \)

in is the pumping

\( S_0 = \text{shaded } 1003 \)

\( S = \{ 00, 01, 10, 11 \} \)

The alphabet \( \Sigma = \{ 0, 1 \} \)

Two buttons 0, 1. Two outputs locked or unlocked

A digital lock with a combination of 100
Ex. $S$ is a set of messages $\{1, 2\}$ and $z$ is a regular event.

$$\forall y \in S, \forall x \in S \exists z \in S$$

Also, the product is associative if $y, x \in S$, and $S$ is a non-empty set.

Def. 4: A semigroup is a system $(S, \cdot)$ where $S$ is a non-empty set and $\cdot$ is a binary operation.

The same language $L \in \text{FSM}$ (one to many) redundant states to change FSM but still define FSM. Interestingly, we can always add some redundant states to change FSM but still define a language. A regular language can have more than one

Note: A regular language $L$ is a regular language.

Ex. $\{00\}$ is a regular language.

is an accepting state.

Simple words: $L$ is a subject of $\Sigma^* S$, the final state.

For $W = \{s, m, so, f\}$, $L(W) = \{x \in \Sigma^* : m(\Sigma^*, x) \in F\}$

A language is a subset of $\Sigma^* \in \Sigma$. Ex. $\{00, 11\}$.

Let $\Sigma$ be the set of all words over $\Sigma$, $Ex.: \{101, 111\}$

Ex. $\{0, 1\}$ is the set of accepting states.

So $S$ is the initial state.
The set of all equivalence classes is denoted by \( x / \theta \).

\[ x / \theta \cdot y / \theta = (x \cdot y) / \theta \]

The multiplication of equivalence classes is denoted by \( \cdot \).

\[ x / \theta = \{ x : x \in \theta \} \]

\( x / \theta \) is a collection of element \( x \).

A permutation class is a collection of elements \( x \).

For a group \( G \), and \( \theta, \eta \in \text{Perm}(G) \), and all \( x, y, z \in G \),

\[ x \cdot (y \cdot z) = (x \cdot y) \cdot z \]

Define \( G = \text{Sym}(G) \), the symmetric group of an equivalence relation.

For the subset property of an equivalence relation.

Ex: Let \( S \) be \( \mathbb{Z} \),

with empty word \( e \),

concatenation of words, then \( (\mathbb{Z}, \cdot) \) is a monoid.

Ex: Let \( S \) be \( \mathbb{Z} \),

with empty word \( e \),

concatenation of words, then \( (\mathbb{Z}, \cdot) \) is a monoid with \( e = 1 \).

Let \( (\mathbb{Z}, \cdot) \) be monoid with \( e = 1 \).

For the rest of talk, I'm going to prove two theorems to show that there exists a unique FSA with minimal states for a regular language.
The index is 3.

Keep this.

\[ x \notin \theta \Delta \forall z \in \theta \]

For \( \exists \in (\text{set}) \),

This is a right stable relation.

There are 3 equal classes. Full words 0.

\[ a \, \text{character} \, 0, \, 1, \, 2 \]

\[ a \, \text{equal}: \, (0, \, 1, \, 2 \, \subseteq \, \{0, \, 1, \, 2\}, \, \exists) \]

Greater than equality relation on \((\text{set}, \, \cdot)\).

Let \( z = \{0, \, 1, \, 2\} \Rightarrow \exists \times = \{0, \, 1, \, 2\} \times \exists \]

Let \( x \in \exists \theta \forall z \in \exists \theta \)

1. For all right stable (or left stable respectively) if for all right index of an equal relation \( \theta \) on a semigroup \( G \) called class.

Index of an equal relation is the number of eq.

Remark?
So it has finite mocc.

2. classes: L and its complement.

3. The relation to partition 0, 1, 2[x

and L is the union of two cell classes.

We found (on last page) that the mocc is 3.

For the language.

There exists an FSM

Example: \[ L = \{ \lambda \} \subseteq \mathcal{X}^* \]

has finite mocc.

Any \( y \in \mathcal{X}^* \) (\( x \in \mathcal{X} \subseteq \mathbb{Z} \))

is defined on \( \mathcal{X} \).

The expression right graph equivalence relation.

The union of some \( \Theta \) equivalence classes

the monoid \( (\mathcal{X}, \cdot) \) has finite mocc, \( L \in \Theta \)

For some right stable equivalence relation \( \Theta \)

(on or \( L \) is regular).

\[ L = \text{L(M)} \] for some finite-state machine.

For a set \( L \subseteq \mathcal{X}^* \) of words, the following one

(Thoreum 1: (Myhill–Nerode theorem))
Thus \( \theta_1 \leq \theta_2 \Leftrightarrow \theta_1 \leq \theta_2 \).

Let \( L \) be the union of \( \Theta \) classes. We have \( x \in \Theta \) if and only if \( x \in L \) for some \( \theta \in \Theta \). Suppose \( \Theta \) is union of \( \Theta \)-classes for some \( \theta \in \Theta \).

If \( \theta \in \Theta \), then \( m(50,0,\theta) = m(50,0,\theta) \in \Theta \). If \( \theta \in \Theta \), then \( m(50,0,\theta) \leq m(50,0,\theta) \).

Also, \( \theta \in \Theta \) if \( \theta \in \Theta \).

The number of classes should be at most one because we partition the set by states.

(2) \[ x \in \Theta \leq m(50,0,\theta) \in \Theta \]

(3) \[ x \in \Theta \leq m(50,0,\theta) \in \Theta \]

To prove this statement.

Equality is on an equivalence relation as we only need to prove (2) where \( \Theta \) is the set of \( \Theta \)-classes.

(Proof: (2) \( \Rightarrow \) \( \Theta \) is a partition.)
Def. 8. For a \( M = (S, m, S_0, F_0) \) to be isomorphic to \( M = (S', m', S_0', F_0') \), we need to show that \( X \in L \leftrightarrow X \in L' \).

To establish \( \sim \) we need to show that \( X \equiv X' \mod \) and

\[
\forall x \in X \iff x \in X'.
\]

Note: It is easier to think \( X \equiv X' \mod \) and

\[
\forall x \in X \iff x \in X'.
\]

Since \( L \subset L' \), since \( L \subset L' \)

\[
F_{L} = \{ x \mid x \in L \} \subset L \subset L'.
\]

Thus, we have obtained a clue to right...
(8.16) F correspond to states in $F_{\pi}$

c) for all $s \in S$, if $s \rightarrow s_0$, then for any
    $\sigma \in \Sigma$, $m(s, \sigma) \rightarrow m_\pi(s_0, \sigma)$.

Simple words: $\exists$ a 1-1 correspondence between
    two sets of machine and if we rename the
    states of one by this correspondence, it becomes
    the same machine as the other.

Theorem 2: A minimum FSM defining a given
    regular language is unique up to isomorphism.
    and is described by the machine $M_\pi$ of theorem 1.

Proof: 1 prove $M_\pi$ is minimum.
    In theorem 1, for any FSM $M$ for $L$,
    $\exists$ a $\Theta'$ on $\Sigma^*$ s.t. $\Theta \subseteq \Theta'$.

1. Index $\pi_0 \leq$ Index $\Theta'$ (for any FSM)

We also show index of $\Theta' \leq |S'|$, the # of states of $M$

2. Index $\pi_0 = |S_{\pi_0}| \leq$ Index $\Theta' \leq |S|$

3. $|S_{\pi_0}| \leq |S|$ for any FSM $M$
Let $\text{m}(x, y) \in \text{m}(50, x)$ and $\forall x \in (F, G)$.

\[
\text{m}(x, y) = \text{m}(50, y) = \text{m}(50, x)
\]

We can deduce this in an accessible state. Let us find a 1-1 correspondence with

\[
|\text{m}| = |\text{m}|
\]

\[
|\text{m}| = |\text{m}|
\]

For any state in $S$, $x \in \text{m}(50, x, y)$ such that $\text{m}(x, y) = \text{m}(50, y) = \text{m}(50, x)$.

To yield a morphism, we need to find a 1-1 correspondence with

\[
\text{m}(x, y) = \text{m}(50, y) = \text{m}(50, x)
\]

\[
|\text{m}| = |\text{m}|
\]

\[
|\text{m}| = |\text{m}|
\]

\[
|\text{m}| = |\text{m}|
\]