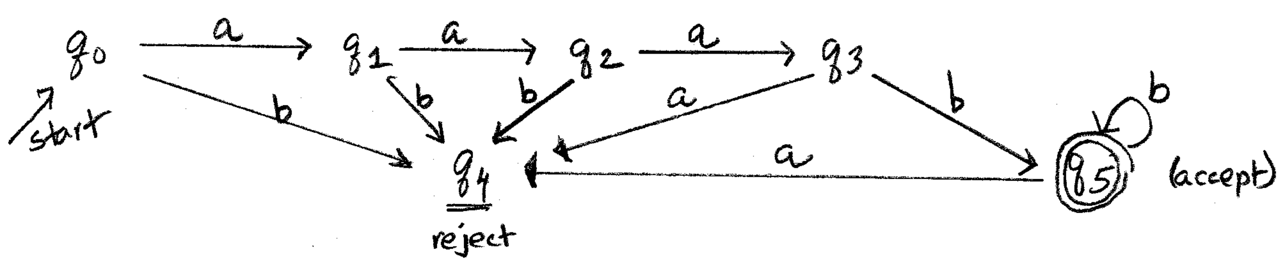
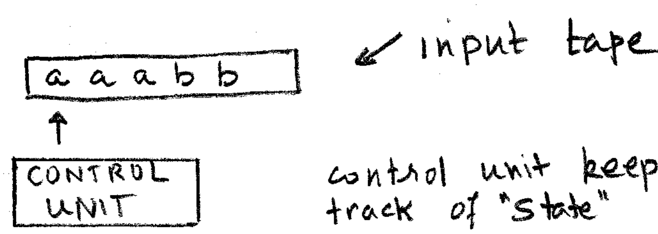
LECTURE 2
FINITE STATE MACHINES

Intuitive idea (deterministic machines)
You want to "recognize" $aaab^n \quad n \geq 1$
What kind of "machine" can do this?

Need to keep track of getting 3 a's and then an arbitrary number of b's.



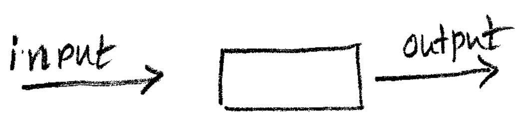
Another view



control unit keeps track of "state" and reads till the end of the tape

Note: control unit needs a finite # of states!

More generally (and still intuitively) look at the idea of input output behavior (I-O Behavior)



Without looking inside the box, we observe the I-O Behavior

Let X be the input alphabet, Y the output alphabet

The I-O Behavior is $R \subseteq X^+ \times Y^+$, i.e. $(x,y) \in R$ with $\ell(x) = \ell(y)$

$$x: a_1 a_2 \cdots a_n \longrightarrow \boxed{} \longrightarrow y: b_1 b_2 \cdots b_n$$

$\oplus$	0	1
0	0	1
1	1	0

xor table

Example

Let the I-O relation be R^M such that:

$(x_0 x_1 x_2 \cdots x_n \,,\, y_0 y_1 y_2 \cdots y_n) \in R^M$

iff for $t \geq 2$ $\quad y_t = x_{t-1} \oplus x_{t-2}$

x_0	x_1	x_2	x_3
1	0	0	1

y_0	y_1	y_2	y_3
?	?	1	0

$x_0 \oplus x_1 \qquad x_1 \oplus x_2$

State is what allows the determination of unique trajectories for the input.

This example machine can be viewed as having 4 states

x_{-1}	x_{-2}
0	0
0	1
1	0
1	1

Note: R^M is a "complete" behavior description. Experimentation can only determine a subset of R^M.

Formal Definition

A finite state (f.s.) sequential machine is a 6-tuple $M = \langle X, Y, Q, \delta, \lambda, q_0 \rangle$ where:

- X: is a finite set of inputs
- Y: is a finite set of outputs
- Q: is a finite set of states
- q_0: is the initial state
- δ: is a transition function $\delta: Q \times X \to Q$ (state transition)
- λ: is the output function $\qquad (q, x) \mapsto q'$

$\lambda : \quad Q \times X \to Y$ Mealy machine $q_1 \xrightarrow{a/0} q_2$

$\qquad (q, x) \mapsto y$

$\lambda : \quad Q \to Y$ Moore machine $q_1/1 \xrightarrow{a} q_2/0$

$\qquad q \mapsto y$

Input sequence $\quad x_0 \, x_1 \cdots x_n$

$\qquad q_0 \, q_1 \cdots q_n \qquad \delta(q_i, x_i) = q_{i+1}$

$\qquad y_0 \, y_1 \cdots y_n$

We define $\tilde{\beta}_{q_0} : X^+ \to Y^+$ extended behavior function from state q_0

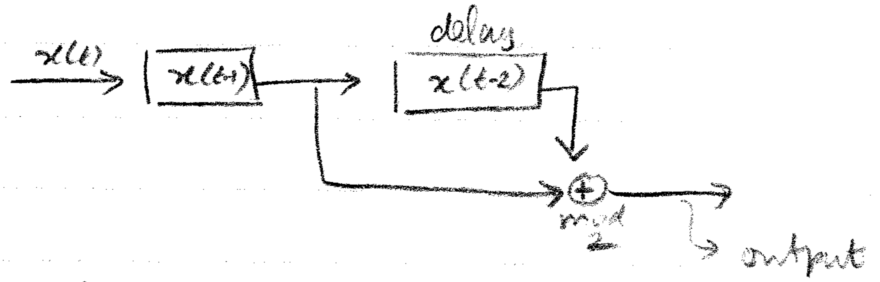
by: $\tilde{\beta}_{q_0}(x_0 \cdots x_n) = y_0 \cdots y_n$

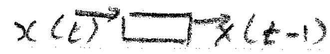
Mealy   where for Mealy: $\quad y_i = \lambda(q_i, x_i)$

(if concerned with physical time) $\quad y(t+1) = \lambda(q(t), x(t))$

Moore   where for Moore: $\quad y_i = \lambda(q_i)$

Realization of R^M defined earlier as Moore machine





$Q = \{(0,0), (0,1), (1,0), (1,1)\}$

1st delay 2nd delay

$Q = \{\hat{q}_0, \hat{q}_1, \hat{q}_2, \hat{q}_3\}$

$\lambda(\hat{q}_0) = 0 \qquad \lambda(\hat{q}_2) = 1$

$\lambda(\hat{q}_1) = 1 \qquad \lambda(\hat{q}_3) = 0$

Q \ X	0	1	λ
$\hat{q}_0$	$\hat{q}_0$	$\hat{q}_2$	0
$\hat{q}_1$	$\hat{q}_0$	$\hat{q}_2$	1
$\hat{q}_2$	$\hat{q}_1$	$\hat{q}_3$	1
$\hat{q}_3$	$\hat{q}_1$	$\hat{q}_3$	0

$\tilde{\beta}_{\hat{q}_1}(0\;1\;0\;0\;1)$

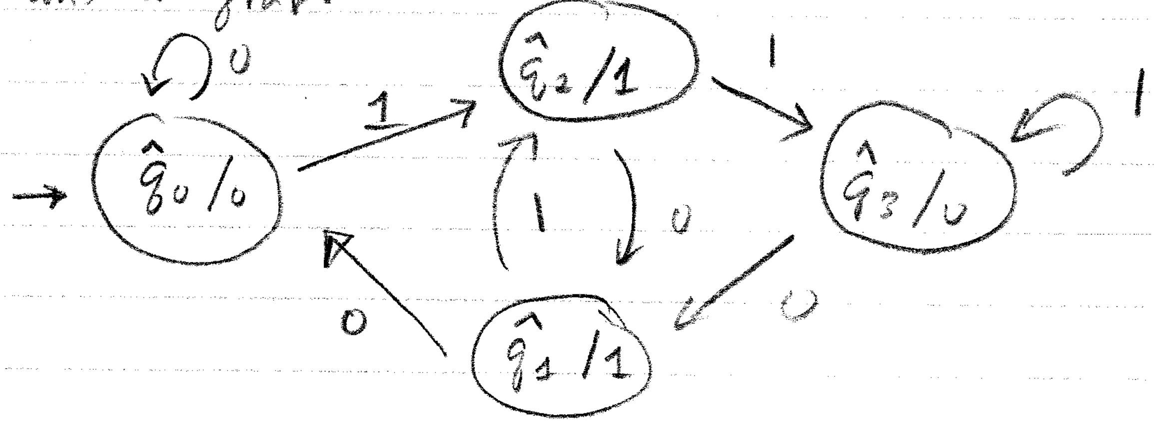
$x_0\; x_1\; x_2\; x_3\; x_4$

$\hat{q}_1\; \hat{q}_0\; \hat{q}_2\; \hat{q}_1\; \hat{q}_0 \quad \hat{q}_2$

$1\; 0\; 1\; 1\; 0$

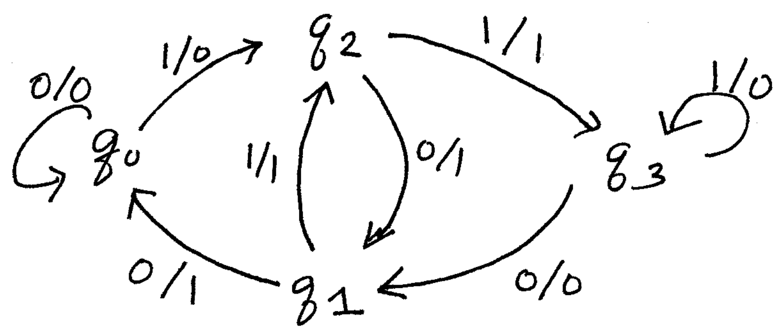
not used for $\tilde{\beta}$

transition graph



Defining a Mealy Machine

$Q \backslash X$	0	1
q_0	$q_0/0$	$q_2/0$
q_1	$q_0/1$	$q_2/1$
q_2	$q_1/1$	$q_3/1$
q_3	$q_1/0$	$q_3/0$



$\tilde{\beta}_{q_0}(111) = 010$

x_0	x_1	x_2	q_0	q_1	q_2	y_0	y_1	y_2
1	1	1	q_0	q_2	q_3	0	1	0

Note that $R^M = \bigcup_{q \in Q} \tilde{\beta}_q$

M_1 is <u>indistinguishable</u> from $M_2 \iff R^{M_1} = R^{M_2}$

M_1 has the same behavior as M_2

$\iff \{\tilde{\beta}_q \mid q \in Q_1\} = \{\tilde{\beta}_q \mid q \in Q_2\}$

Note : M_1 has same behavior as M_2

$\Rightarrow$ M_1 indistinguishable M_2

Is the reverse of this statement true ?

Given a sequential machine $\langle X, Y, Q, q_0, \delta, \lambda \rangle$ we define the extended transition function $\tilde{\delta}$ by:

$\tilde{\delta} : Q \times X^* \rightarrow Q$ s.t.

$\tilde{\delta}(q, a) = \delta(q, a)$ for $a \in X$

$\tilde{\delta}(q, xa) = \delta(\tilde{\delta}(q, x), a)$ for $x \in X^+, a \in X$

$\tilde{\delta}(q, \lambda) = q$

We define the extended output function $\tilde{\lambda}$ by:

<u>Mealy</u> $\tilde{\lambda} : Q \times X^+ \rightarrow Y$ s.t.

$\tilde{\lambda}(q, a) = \lambda(q, a)$ for $a \in X$

$\tilde{\lambda}(q, xa) = \lambda(\tilde{\delta}(q, x), a)$ for $x \in X^+, a \in X$.

<u>Moore</u> $\tilde{\lambda} : Q \times X^* \rightarrow Y$ s.t.

$\tilde{\lambda}(q, x) = \lambda(\tilde{\delta}(q, x))$ for $x \in X^*$

Defining the behavior function β_q from state q

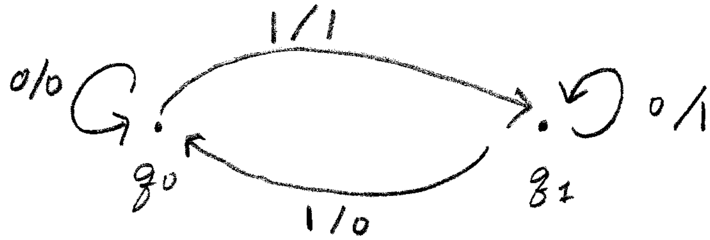
<u>Mealy</u> $\beta_q : X^+ \rightarrow Y$ "output starting from state q obtained on <u>last</u> transition"

$\beta_q(xa) = \tilde{\lambda}(q, xa)$ $x \in X^*, a \in X$

$\beta_q : X^* \rightarrow Y$ "output starting from state q obtained from last state entered"

$\beta_q(x) = \tilde{\lambda}(q, x)$ $x \in X^*$

<u>Example Mealy</u>



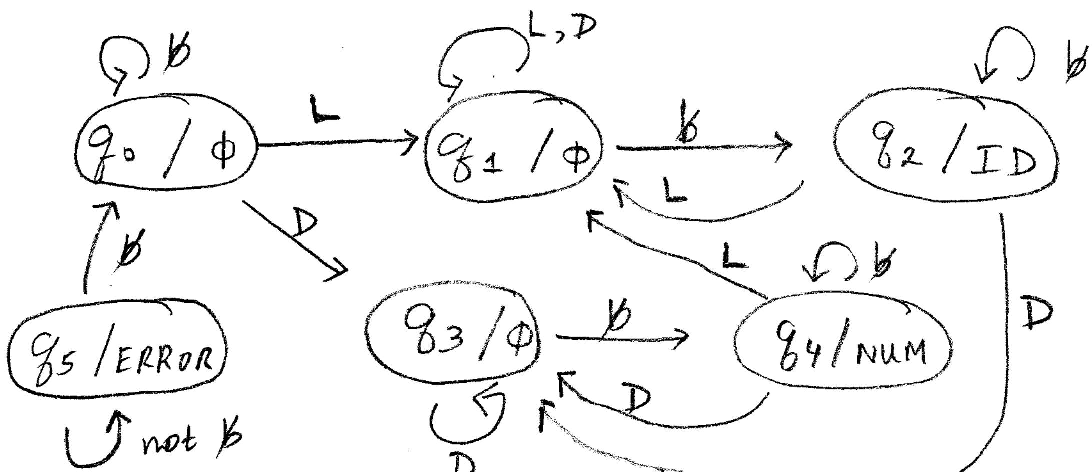
$\beta q_0(011010) = 1$
$\beta q_1(011010) = 0$

For Mealy we have: $\tilde{\beta}_q(x_1 x_2 \cdots x_n) = \beta_q(x_1)\beta_q(x_1 x_2) \cdots \beta_q(x_1 x_2 \cdots x_n)$
$\tilde{\beta}_{q_0}(011) = 010$
$\hat{\beta}_{q_0}(011010) = 010011$

<u>Example Moore</u>

Let $L \triangleq A|B|C \cdots |Z$ (letter)
$D \triangleq 0|1|2 \cdots |9$ (digit)
$\square, b\!\!\!/ \triangleq$ "blank" $\quad \sigma \triangleq$ "other character"

Let $X = \{L, D, b\!\!\!/, \sigma\}$ and $Y = \{\phi, ID, NUM, ERROR\}$



All other transitions go to q_5 (error) from q_0, q_1, q_2, q_3, q_4

$\beta_{q_0}(\text{A25}\square\square) = \text{ID}$
$\beta_{q_0}(123\square) = \text{NUM}$
$\beta_{q_0}(12?4) = \text{ERROR}$

Note that for Moore we have:
$\tilde{\beta}_q(x_1 x_2 \cdots x_n) = \beta_q(\lambda)\beta_q(x_1)\cdots\beta_q(x_1 \cdots x_{n-1})$
we make sure length of input & output match.

Synthesis of Sequential Machines

Let $|X|$ be finite and $|Y|$ be finite

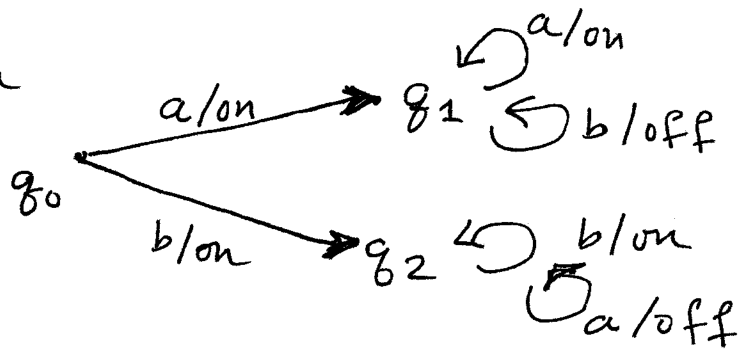
Given $f : X^+ \to Y$ we say that a Mealy machine M realizes f if there exists a state $q \in Q$ s.t. $\beta_q = f$

Example: A "blackbox" turns on the light if the first and last elements of a sequence are the same.

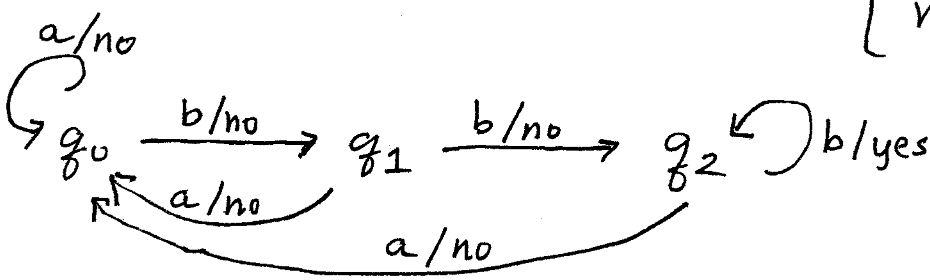
$X = \{a, b\}$ $Y = \{\text{on}, \text{off}\}$

$f : X^+ \to Y$ $f(x) = \begin{cases} \text{on} & \text{if 1st \& last symbols same} \\ \text{off} & \text{otherwise} \end{cases}$

Solution



<u>Example</u> $f : X^+ \to Y$ $f(x) = \begin{cases} \text{yes, if } x \text{ ends in bbb} \\ \text{no, otherwise} \end{cases}$

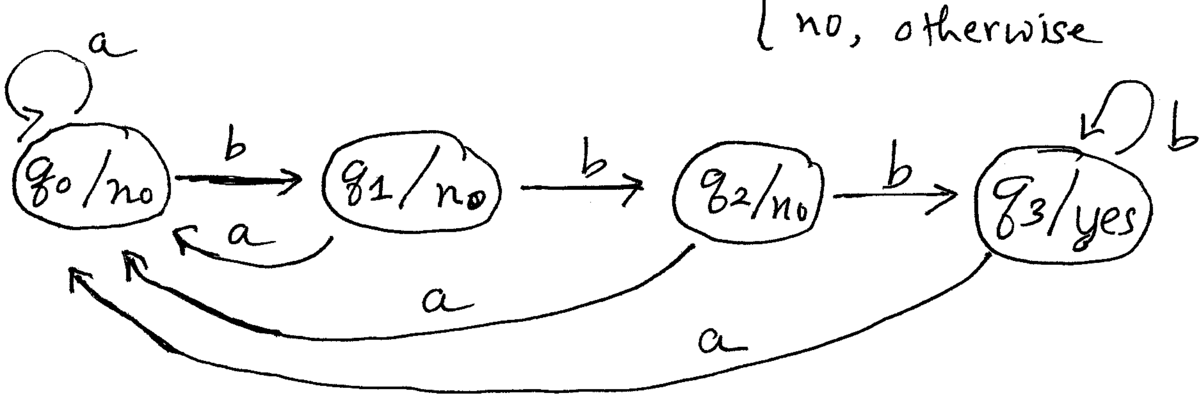


Synthesis for Moore Machines

$f : X^* \to Y$

Moore machine M realizes f if $\exists$ state $q \in Q$ s.t. $\beta_q = f$

<u>Example</u> $f : X^* \to Y$ $f(x) = \begin{cases} \text{yes, if } x \text{ ends in bbb} \\ \text{no, otherwise} \end{cases}$

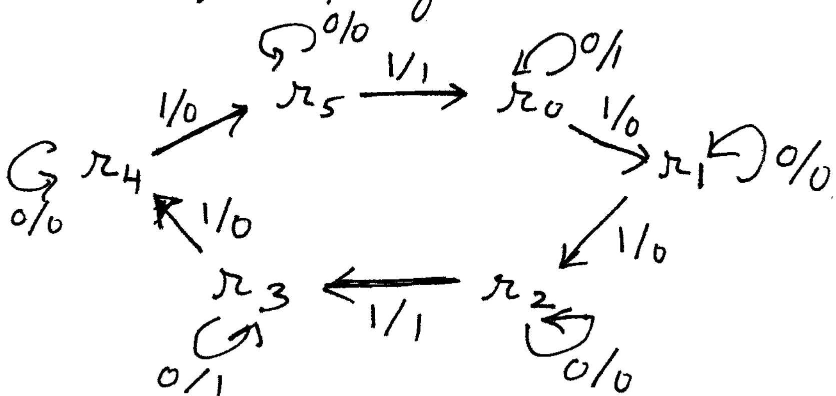


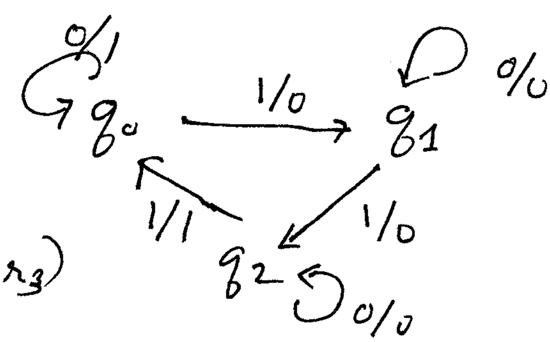
Equivalence of Machine Behaviors

Def. Let M and M' have state sets Q and Q' respectively. q is <u>equivalent</u> to q' $(q \equiv q')$ if $\beta_q = \beta_{q'}$

Def $M \equiv M' \iff \{\beta_q \mid q \in Q\} = \{\beta_{q'} \mid q' \in Q'\}$

Example of equivalent machines





Note that

q_0 corresponds to r_0, r_3 $(q_0 \equiv r_0 \equiv r_3)$

Similarly $q_1 \equiv r_1 \equiv r_4$

$q_2 \equiv r_2 \equiv r_5$

Def M is <u>reduced</u> if $q \equiv r \Rightarrow q = r$

alternatively M is reduced if $\beta_q = \beta_r \Rightarrow q = r$

Equivalence Relation

Note that $\equiv$ or $q \equiv r$ is an equivalence relation.

What is an equivalence relation? It is a relation R that satisfies three properties:

1. reflexive property $\forall x$, $(x,x) \in R$ or $x \equiv x$
2. symmetric property $x \equiv y \Rightarrow y \equiv x$
3. transitive property $x \equiv y$ and $y \equiv z \Rightarrow x \equiv z$

Example

mod 5 defines an equivalence relation on the integers

$0 \equiv 5 \equiv 10 \equiv 15 \quad \ldots$

$1 \equiv 6 \equiv 11 \quad \ldots$

$2 \equiv 7 \equiv 12 \quad \ldots$

Other examples

"in the same classroom" is an equivalence relation

"likes" is not an equivalence relation

Why is $q \equiv r$ an equivalence relation?

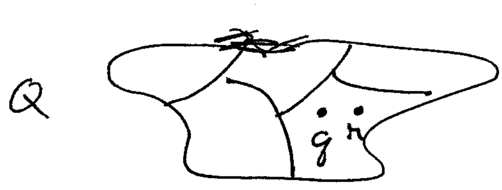
Easy to show using definition $\beta_q = \beta_r$

$\beta_q = \beta_r$ can be shown to be an equivalence relation.

$$\beta_q = \beta_q \quad \forall q$$

$$\beta_q = \beta_r \Rightarrow \beta_r = \beta_q$$

$$\beta_q = \beta_r \ \& \ \beta_r = \beta_s \Rightarrow \beta_q = \beta_s$$



We often partition a set Q into partition classes Π_q, Π_r based on an equivalence relation $[q] = [r]$

Theorem For any machine M, there is a reduced machine M_R s.t. $M \equiv M_R$

Proof Let $M = (X, Y, Q, \delta, \lambda)$
Define $M_R = (X, Y), Q_R, \delta_R, \lambda_R)$ as follows:

Let $Q_R = \{ [q] \mid q \in Q \}$ be the equivalence classes of Q where $[q]$ is the set of states equivalent to q. That is:
$q \equiv r$ if $\beta_q = \beta_r$.

Let $\delta_R : Q_R \times X \longrightarrow Q_R$ by
$$\delta_R : ([q], a) \longmapsto [\delta(q,a)]$$

Note that we must show this definition of δ_R is well-defined!
That is, we need to show that if $q \equiv r$ then $\delta(q,a) \equiv \delta(r,a)$
Assume $q \equiv r$. Then $\beta_q = \beta_r$
$\therefore \beta_q(ax) = \beta_r(ax) \quad \forall a \in X, x \in X^+$
$\therefore \beta_{\delta(q,a)}(x) = \beta_{\delta(r,a)}(x) \quad \forall x \in X^+$
$\Rightarrow [\delta(q,a)] = [\delta(r,a)]$ or $\delta(q,a) \equiv \delta(r,a)$.
An equivalence relation satisfying this transition property is called right invariant.

We call also define
$$\lambda_R : Q_R \times X \to Y$$
$$([q], a) \mapsto [\lambda(q,a)]$$
(We need to also show that this function is well-defined)

We now claim that (1) $M \equiv M_R$

(2) M_R is reduced

Proof

(1) $\forall q \in Q$, $\beta_q \equiv \beta_{[q]} \therefore M \equiv M_R$ (proof of part 1)

(2) Suppose $[q] \equiv [q']$, that is $\beta_{[q]} = \beta_{[q']}$

By definition $\beta_q = \beta_{[q]} = \beta_{[q']} = \beta_{q'}$

$\therefore q \equiv q'$ or $[q] = [q']$

This complete proof that for any M, there is a reduced machine equivalent to M.

Describing things behaviorally.

$$\xrightarrow{x \in X^+} \boxed{M} \xrightarrow{y \in Y} \qquad \beta : X^+ \to Y$$

Can we capture the idea of "state"

Yes, state is "past history"!

Nerode Equivalence

Define an equivalence relation E_β on X^*

by $x E_\beta y$ $(x \equiv y)$

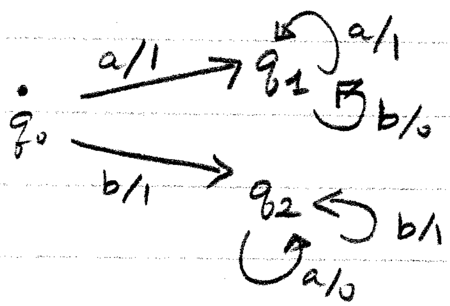
if $\beta(xz) = \beta(yz)$ for all $z \in X^+$.

We write $[x]_\beta$ for the equivalence class

Two input strings are equivalent if their "future behavior is the same."

<u>Example</u> $X = \{a, b\}$ $Y = \{0, 1\}$

$$\beta(x) = \begin{cases} 1 & \text{if 1st \& last symbols of } x \text{ the same} \\ 0 & \text{otherwise} \end{cases}$$



Partitioning X^*

$[\lambda] = \{\lambda\}$

$[a] = \{x \mid x \text{ begins with } a\}$

$[b] = \{x \mid x \text{ begins with } b\}$

<u>Def</u> If $\beta : X^+ \to Y$ then the Nerode machine of β is $M_\beta = (X, Y, Q_\beta, \delta_\beta, \lambda_\beta)$

where $Q_\beta = \{ [x]_\beta \mid x \in X^* \}$

$\delta_\beta([x]_\beta, a) = [xa]_\beta$

$\lambda_\beta([x]_\beta, a) = \beta(xa)$

Can you prove this is well defined?

Use definitions! Note will just write $[x]$ for $[x]_\beta$ if context is obvious.

Def: Let $l_x : X^+ \to X^+$ $x \in X^*$

by $z \mapsto xz$

Note: $\beta \circ l_x$ is a well defined function.

$\beta \circ l_x(z) = \beta(l_x(z)) = \beta(xz)$

Note

$\beta_{[x]_\beta} = \beta \circ l_x$ Let $z = ya$ $a \in X, y \in X^*$

$$\therefore \beta_{[x]_\beta}(z) = \lambda_\beta(\tilde{\delta}_\beta([x]_\beta, y), a)$$
$$= \lambda_\beta([xy]_\beta, a)$$
$$= \beta(xya) = \beta(xz) = \beta \circ l_x(z)$$

Theorem If M_β is the Nerode machine of β then
(1) M_β realizes β.
(2) M_β is reduced.

Proof (1) Consider state $[\lambda]$. $\beta_{[\lambda]} = \beta l_\lambda = \beta$.
(2) Suppose $[x] \equiv [y]$

$$\Rightarrow \beta_{[x]} = \beta_{[y]} \Rightarrow \beta(xz) = \beta(yz) \quad \forall z \in X^+$$
$$\Rightarrow x \, E_\beta \, y \Rightarrow [x] = [y].$$

Note: $[x] = [y] \Leftrightarrow \beta l_x = \beta l_y$

Corollary.
A function $\beta : X^+ \to Y$ is finite state realizable $\Leftrightarrow$ E_β has a finite # of classes.

Def For $\beta : X^+ \to Y$, the machine of β is

$$M(\beta) = (X, Y, Q_\beta, \delta_\beta, \lambda_\beta)$$
$$Q_\beta = \{ \beta \ell_x \mid x \in X^* \}$$
$$\delta_\beta(\beta\ell_x, a) = \beta\ell_{xa}$$
$$\lambda_\beta(\beta\ell_x, a) = \beta(xa)$$

Similar to Nerode machine; just a different point of view. Looking at functions instead of equivalence classes of strings.

Myhill-Nerode Theorem If $\beta : X^+ \to Y$ the the following are equivalent:

1. β is finite state realizable
2. E_β has a finite number of classes
3. $|\{ \beta\ell_x \mid x \in X^* \}| < \infty$ (ie. finite)

Example $X = Y = \{0,1\}$, $\beta : X^+ \to Y$ $\beta(x) = \begin{cases} 1 \text{ if } \# \text{ 1's in } X \geq \# \text{ 0's in } X \\ 0 \text{ otherwise} \end{cases}$

Not finite state realizable

z	$\beta(z)$	$\beta\ell_0(z)$	$\beta\ell_{00}(z)$	$\beta\ell_{000}(z)$	...
1	1	1	0	0	0
11	1	1	1	0	0
111	1	1	1	1	0
1111	1	1	1	1	1
⋮					

etc.

$\therefore$ Infinite (ie. not finite) # of different functions $\beta\ell_x$.

Algorithm for minimizing state in a f.s. seq machine

Idea: find partition of state space that satisfies $\forall a \in X$, $\delta(q,a)$ & $\delta(r,a)$ in same class if q & r are in the same partition class.

Example

$Q \backslash X$	0	1	λ
0	0	1	1
1	1	2	0
2	2	3	0
3	3	4	1
4	4	5	0
5	5	0	0

<u>1st</u> partition by output λ. $\{0,3\}$, $\{1,2,4,5\}$

states are distinguishable across classes

2nd, $\{1,2,4,5\}$ splits. $\{0,3\}$, $\{1,4\}$ $\{2,5\}$

3rd. No more splitting all states in a class are indistinguishable. (equivalent)

∴ More precisely <u>Machine reduction algorithm</u> is:
first partition states into Π_r corresponding to outputs. Then find the coarsest partition that refines Π_r and satisfies the SP property:

SP property: if q & r are in the same partition class, then for all $a \in X$, $\delta(q,a)$ and $\delta(r,a)$ are in the same partition class.

Note: this is an $O(n^2)$ algorithm. $\exists$ an $O(n \log n)$ algorithm for machine reduction, where n is the number of states.

Example

	a_0	a_1
q_0	$q_1/0$	$q_2/1$
q_1	$q_3/1$	$q_1/0$
q_2	$q_2/0$	$q_4/1$
q_3	$q_3/1$	$q_3/0$
q_4	$q_1/1$	$q_2/0$

$\Pi_1 = \Pi_r$

		a_0	a_1
class 1	q_0	$q_1 : 2$	$q_2 : 1$
	q_2	$q_2 : 1$	$q_4 : 2$
class 2	q_1	$q_3 : 2$	$q_1 : 2$
	q_3	$q_3 : 2$	$q_3 : 2$
	q_4	$q_1 : 2$	$q_2 : 1$

q_0 & q_2 of class 1 in different classes
q_4 in a different class from q_1 & q_3

		a_0	a_1
(1)	q_0	✓	✓
(2)	q_2	✓	✓
(3)	q_1	$q_3 : 3$	$q_1 : 3$
	q_3	$q_3 : 3$	$q_3 : 3$
(4)	q_4	✓	✓

Done!

Shift attention to realization for
f (or β) $\quad f: X^* \to Y$

Example

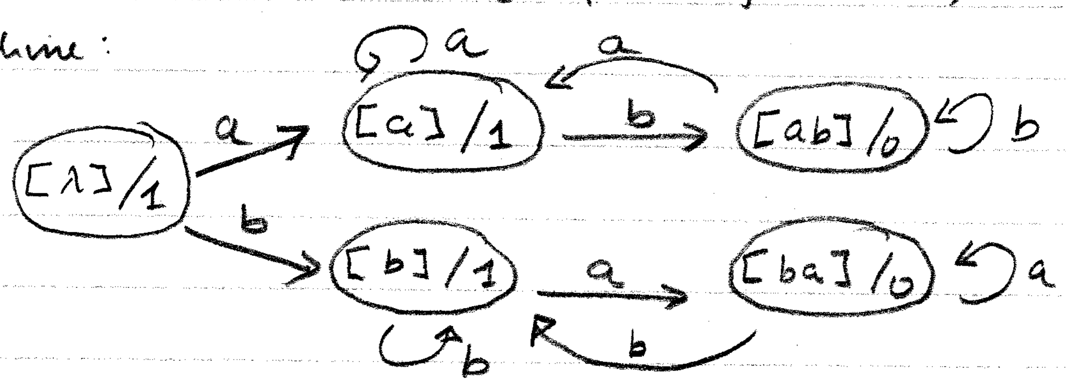
$\beta: X^* \to Y \qquad \begin{cases} 1 & \text{if 1st \& last symbol the same.} \\ 0 & \text{otherwise} \end{cases}$

Note: let $\beta(\lambda) = 1$

Nerode equivalence: $x \equiv y \iff \beta(xz) = \beta(yz)$ for $z \in X^*$

$\therefore$ classes are
$[\lambda] = \{\lambda\}$
$[a] = \{x \mid x \text{ begins and ends with } a\}$
$[b] = \{x \mid x \text{ begins and ends with } b\}$
$[ab] = \{x \mid x \text{ begins with } a, \text{ ends with } b\}$
$[ba] = \{x \mid x \text{ begins with } b, \text{ ends with } a\}$

Moore machine:



Consider $\beta: X^* \to Y$. Let $Y = \{y_1, \ldots, y_n\}$.
Look at $\beta^{-1}(y_1), \beta^{-1}(y_2), \ldots, \beta^{-1}(y_n)$.

Def: If $L \subseteq X^*$, then L is a <u>finite state language</u> if there exists a finite state Moore machine $M = \langle X, Y, Q, \delta, \lambda \rangle$ s.t. $\beta_q^{-1}(y) = L$ for some $q \in Q$, $y \in Y$.