Math 327, Lecture 35, 11-18.

- Finite State Automata (FSA)

- The languages accepted by (defined by) these machines.

A FSA is a model of a very primitive computer:

- input device that reads strings of characters (words)
- One memory location - possible large.
- Can output a yes/no determination.

A FSA is deterministic. The future is determined by the current state and the next input character.

An FSA, M, has the following parts:

- A – a set called the alphabet
- S – a finite set – the set of states.
- $\delta: S \times A \longrightarrow S$, called the transition function.
- q_0 – an element of S called the start state.
- F – a subset of S, called the set of final states.

If $\delta(s,a) = t$ we intend that if the machine is in state s when the character a is input, then the machine moves to state t.

```
char c = nextchar();
State = δ(State, t);
```

Even for modestly-sized state sets, an FSA can describe a usefull, albiet low-level, language.

Example: $A = \{a, b\}$, $S = \{0, 1, 2\}$;

M:

δ	a	b
0	1	0
1	2	1
2	2	2

$q_0 = 0$;

$F = \{2\}$

On this input: aba

The machine follows these states: (0,) 1, 1, 2.

2 is in F, so we say M accepts the word aba.

On this: abb

Follow this: (0,) 1, 1, 1

$1 \notin F$ so M rejects abb.

Example. $S = \{0,1\}$; $A = \{0,1\}$;

N:

δ	0	1
0	0	1
1	1	0

$q_0 = 0$; $F = \{0\}$.

If N is in state j:

an input of 0 preserves the state;

an input of 1 flips the state.

The words accepted by N are exactly those strings in 0 & 1 with an even number of ones.

This example was easy because δ is the operation from a semigroup. ((FSA $\Leftrightarrow$ semigroups))

related ??

Need a visual aid. (Modified digraphs.)

Draw $\textcircled{s}$ to represent s in S;

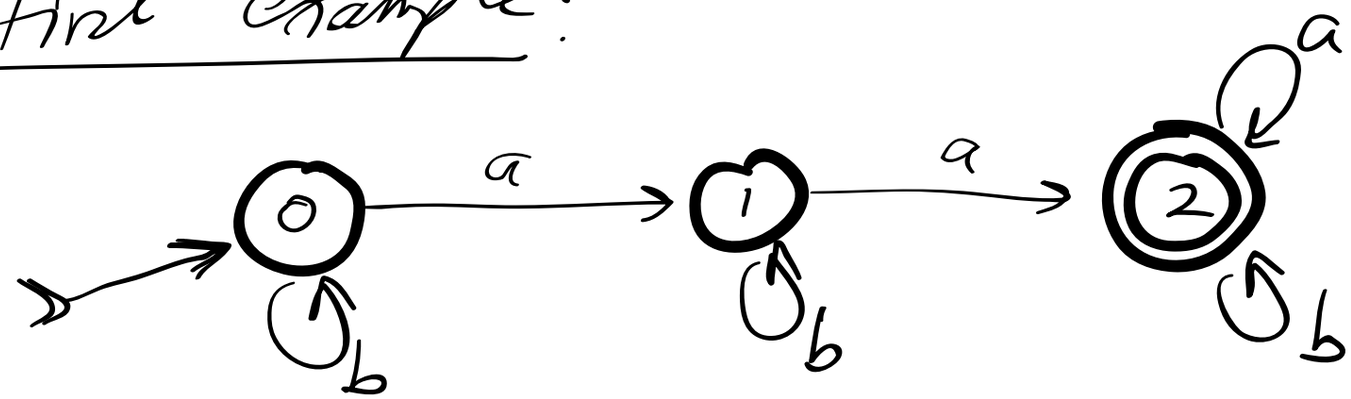
Use $\twoheadrightarrow$ (or whatever) to mark q_0;

Use $\textcircled{\textcircled{t}}$ when $t \in F$;

Use $\textcircled{s} \xrightarrow{a} \textcircled{t}$ when $\delta(s, a) = t$.

Back to first example:

M:

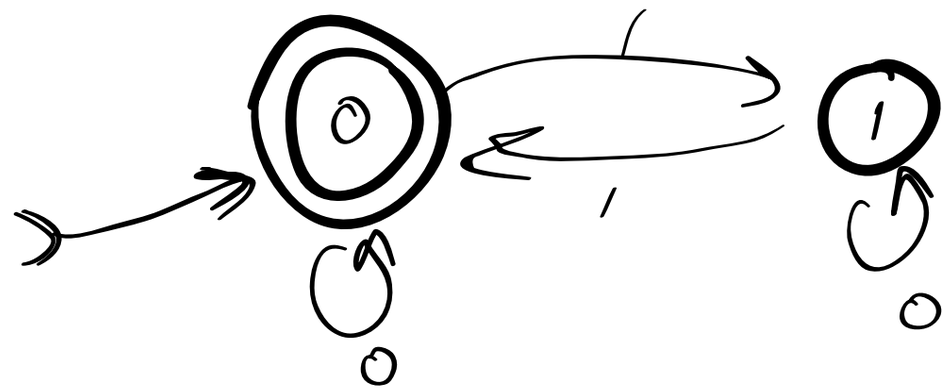


An 'a' moves us left, if possible.
A 'b' leaves us standing still.
Start on left, hope to end on right.

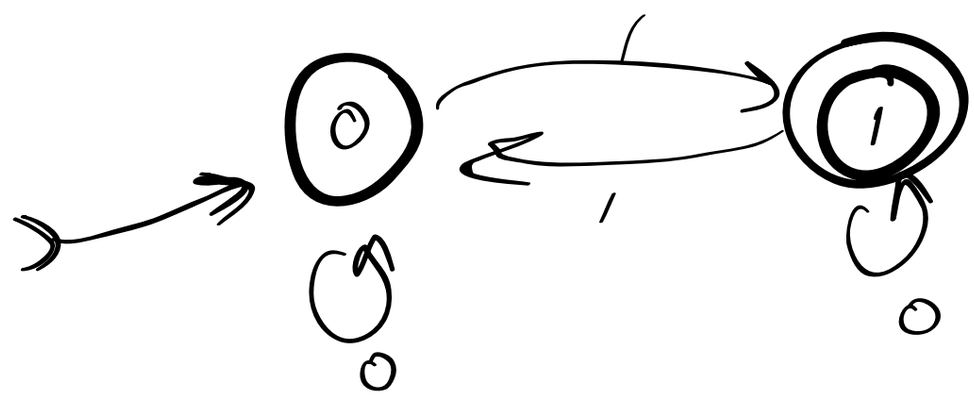
$\therefore$ M accepts those words in $\{a,b\}$ which have at two 'a's in them.

2nd example:

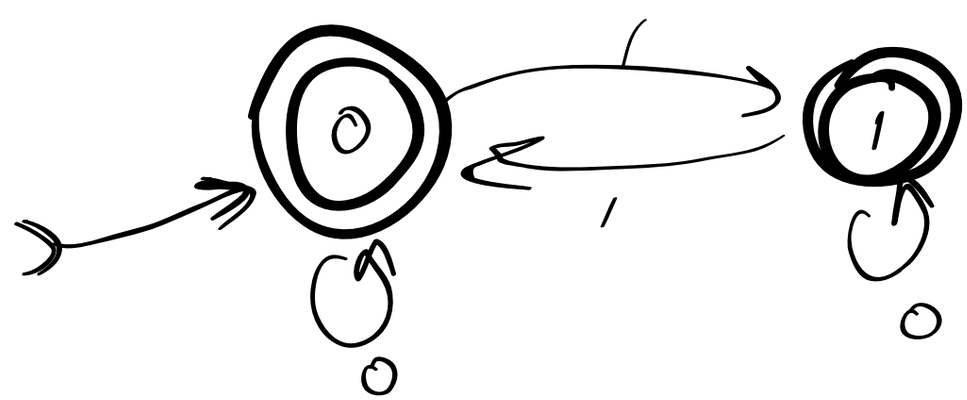
N:



N is a parity checker.



accepts odd # of ones



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

accepts all words in 0 and 1.