CS 150 Lecture Slides

Motivation

- Automata = abstract computing devices

- Turing studied Turing Machines (= computers) before there were any real computers

- We will also look at simpler devices than Turing machines (Finite State Automata, Pushdown Automata, . . .), and specification means, such as grammars and regular expressions.

- NP-hardness = what cannot be efficiently computed

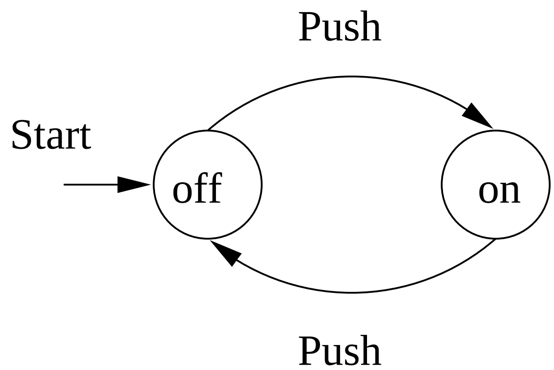
Finite Automata

Finite Automata are used as a model for

- Software for designing digital circuits
- Lexical analyzer of a compiler
- Searching for keywords in a file or on the web.
- Software for verifying finite state systems, such as communication protocols.

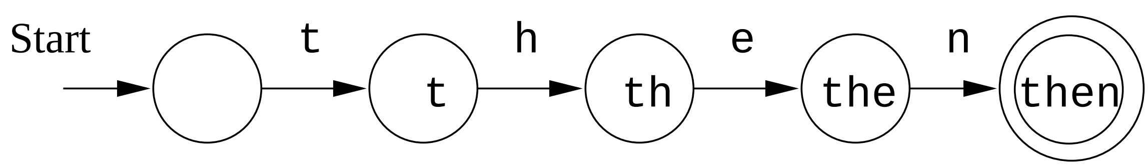
❈ Computer graphics and fractal compression.

- Example: Finite Automaton modelling an on/off switch



Model of Computation:
A program

- Example: Finite Automaton recognizing the string **`then`**



Model of Description:
A specification

Structural Representations

These are alternative ways of specifying a machine

Grammars: A rule like $E \Rightarrow E + E$ specifies an arithmetic expression

- $Lineup \Rightarrow Person.Lineup$ Recursion!

says that a lineup is a person in front of a lineup.

Regular Expressions: Denote structure of data, e.g.

`'[A-Z][a-z]*[][A-Z][A-Z]'`

matches `Ithaca NY`

does not match `Palo Alto CA`

Question: What expression would match `Palo Alto CA`

[A-Z][a-z]*[ ][A-Z][a-z]*[ ][A-Z][A-Z]

Central Concepts

Alphabet: Finite, nonempty set of symbols

Example: $\Sigma = \{0, 1\}$ binary alphabet

Example: $\Sigma = \{a, b, c, \ldots, z\}$ the set of all lower case letters

Example: The set of all ASCII characters

Strings: Finite sequence of symbols from an alphabet Σ, e.g. 0011001

Empty String: The string with zero occurrences of symbols from Σ

- The empty string is denoted ϵ

Length of String: Number of positions for symbols in the string.

$|w|$ denotes the length of string w

$|0110| = 4, |\epsilon| = 0$

Powers of an Alphabet: Σ^k = the set of strings of length k with symbols from Σ

Example: $\Sigma = \{0, 1\}$

$\Sigma^1 = \{0, 1\}$

$\Sigma^2 = \{00, 01, 10, 11\}$

$\Sigma^0 = \{\epsilon\}$

Question: How many strings are there in Σ^3

The set of all strings over Σ is denoted Σ^*

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \cdots$ E.g. {0,1}*

Also:

$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \cdots$

$\Sigma^* = \Sigma^+ \cup \{\epsilon\}$

Concatenation: If x and y are strings, then xy is the string obtained by placing a copy of y immediately after a copy of x

$x = a_1 a_2 \ldots a_i, y = b_1 b_2 \ldots b_j$

$xy = a_1 a_2 \ldots a_i b_1 b_2 \ldots b_j$

Example: $x = 01101, y = 110, xy = 01101110$

Note: For any string x

$$x\epsilon = \epsilon x = x$$

Languages:

If Σ is an alphabet, and $L \subseteq \Sigma^*$
then L is a language

Examples of languages:

- The set of legal English words
- The set of legal C programs
- The set of strings consisting of n 0's followed by n 1's

$$\{\epsilon, 01, 0011, 000111, \ldots\}$$

$\{0^n 1^n | \text{ n} >= 0\}$

- The set of strings with equal number of 0's and 1's

$$\{\epsilon, 01, 10, 0011, 0101, 1001, \ldots\}$$

- $L_P =$ the set of binary numbers whose value is prime

$$\{10, 11, 101, 111, 1011, \ldots\}$$

- The empty language $\emptyset$

- The language $\{\epsilon\}$ consisting of the empty string

Note: $\emptyset \neq \{\epsilon\}$

Note2: The underlying alphabet Σ is always finite

Problem: Is a given string w a member of a language L? (Membership Question)

Example: Is a binary number prime = is it a member in L_P

Is $11101 \in L_P$? What computational resources are needed to answer the question.

Usually we think of problems not as a yes/no decision, but as something that transforms an input into an output.

Example: Parse a C-program = check if the program is correct, and if it is, produce a parse tree.

Let L_X be the set of all valid programs in prog lang X. If we can show that determining membership in L_X is hard, then parsing programs written in X cannot be easier.

Question: Why?

language == (decision) problem!

Finite Automata Informally

Protocol for e-commerce using e-money

Allowed events:

1. The customer can *pay* the store (=send the money-file to the store)

2. The customer can *cancel* the money (like putting a stop on a check)

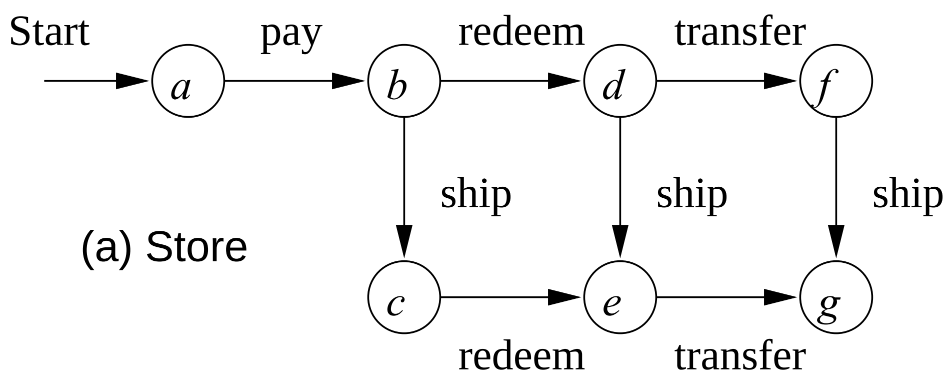
3. The store can *ship* the goods to the customer

4. The store can *redeem* the money (=cash the check)

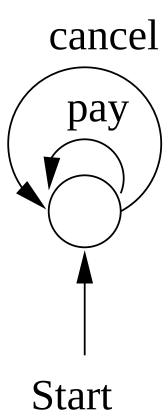
5. The bank can *transfer* the money to the store

e-commerce

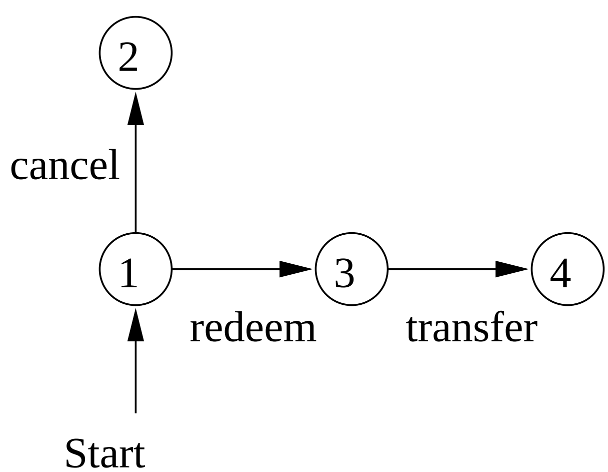
The protocol for each participant:



(a) Store

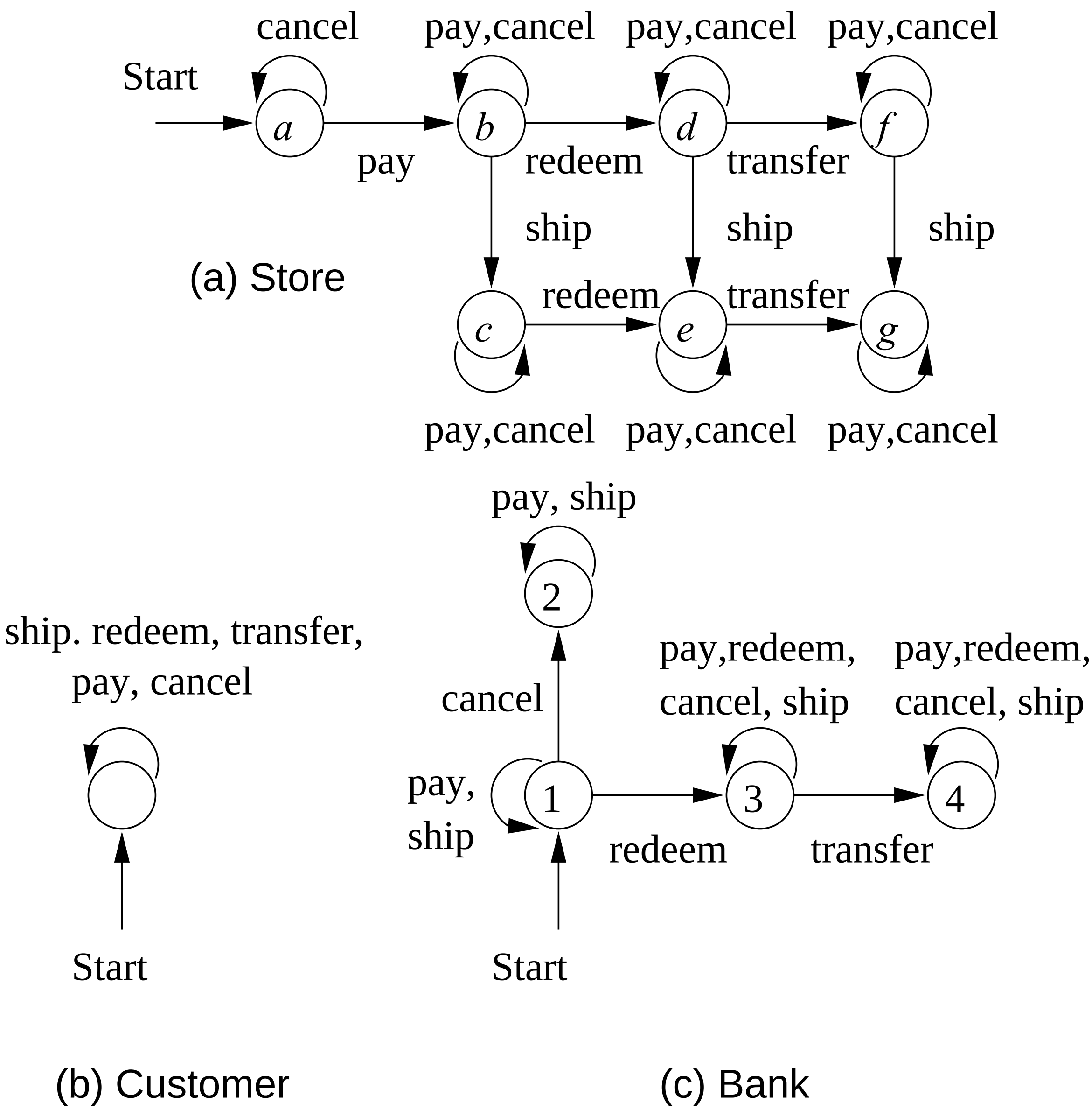


(b) Customer

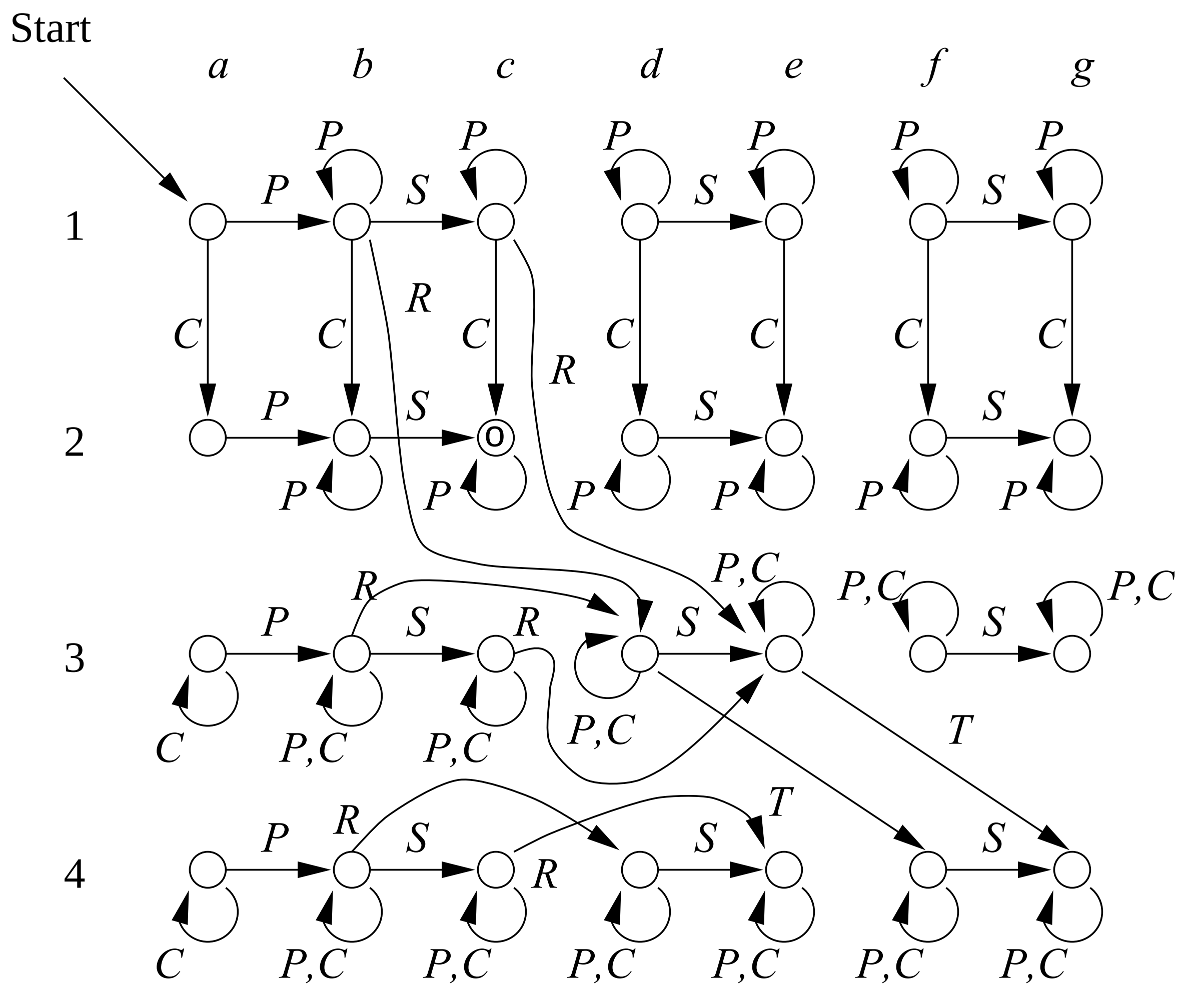


(c) Bank

Completed protocols:

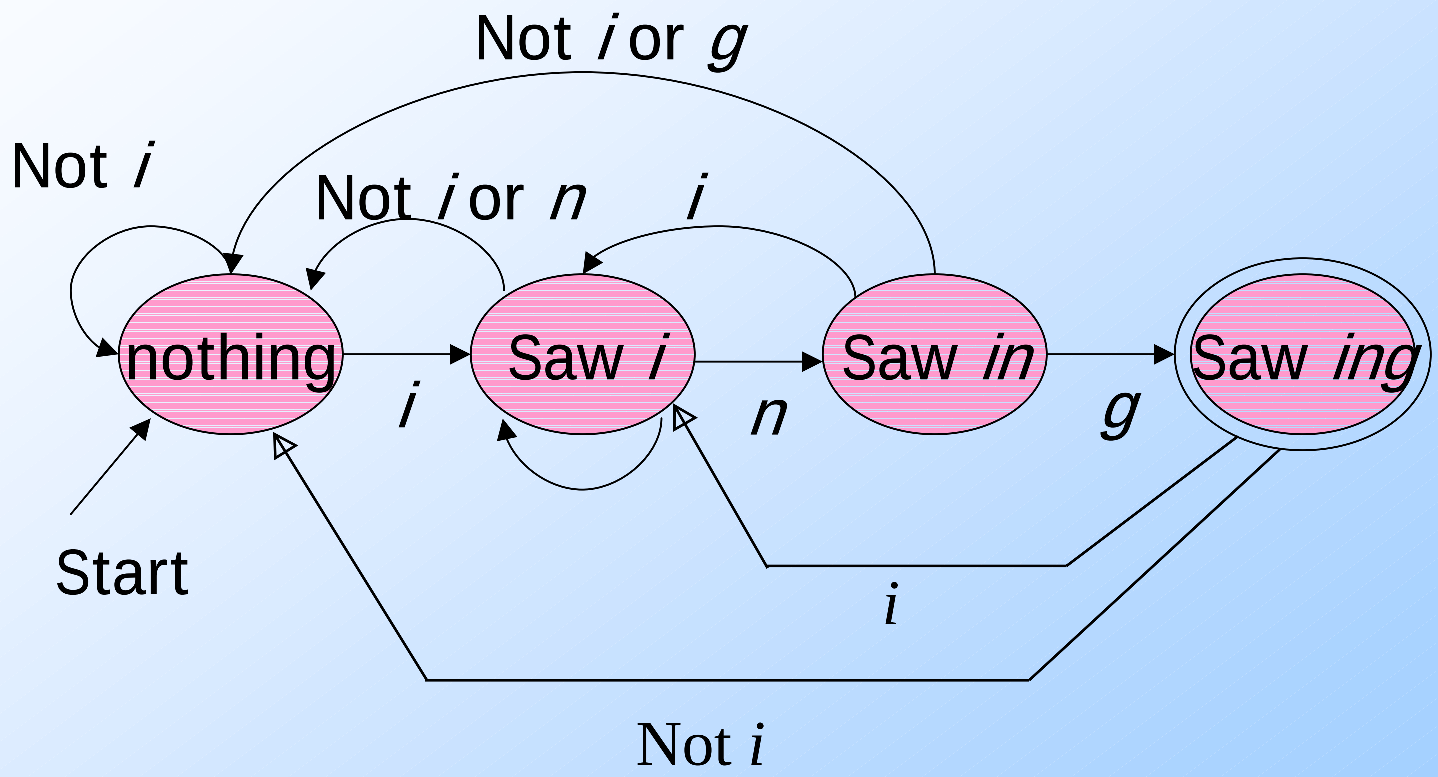


The entire system as an Automaton:



More applications of FA can be found in Linz, Ch. 1.3.

Example: Recognizing Strings Ending in "ing"



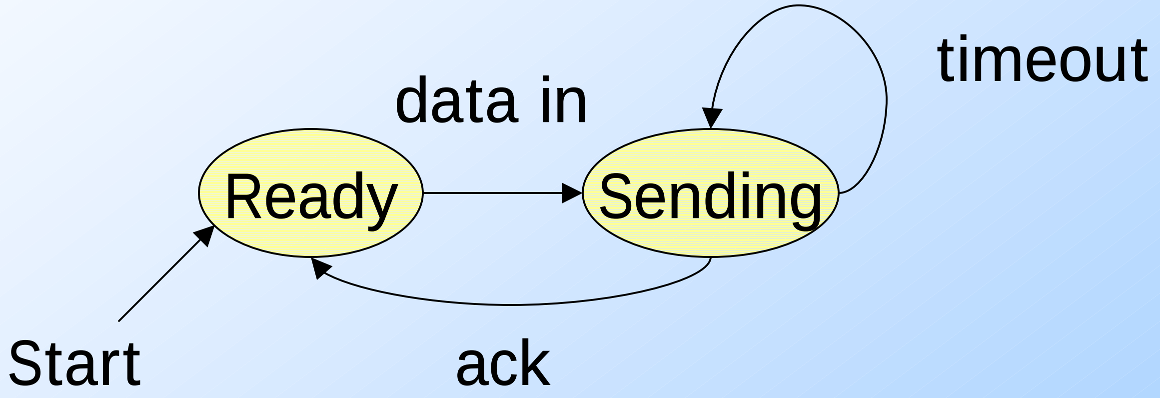
Automata to Code

- In C/C++, make a piece of code for each state. This code:
  1. Reads the next input.
  2. Decides on the next state.
  3. Jumps to the beginning of the code for that state.

Example: Automata to Code

```
2: /* i seen */
  c = getNextInput();
  if (c == ’n’) goto 3;
  else if (c == ’i’) goto 2;
  else goto 1;
3: /* ”in” seen */
  . . .
```

Example: Protocol for Sending Data



Extended Example

- Thanks to Jay Misra for this example.
- On a distant planet, there are three species, a, b, and c.
- Any two different species can mate. If they do:
  1. The participants die.
  2. Two children of the third species are born.

Strange Planet – (2)

- Observation: the number of individuals never changes.
- The planet *fails* if at some point all individuals are of the same species.
  - Then, no more breeding can take place.
- *State* = sequence of three integers – the numbers of individuals of species a, b, and c.

Strange Planet – Questions

- In a given state, must the planet eventually fail?
- In a given state, is it possible for the planet to fail, if the wrong breeding choices are made?

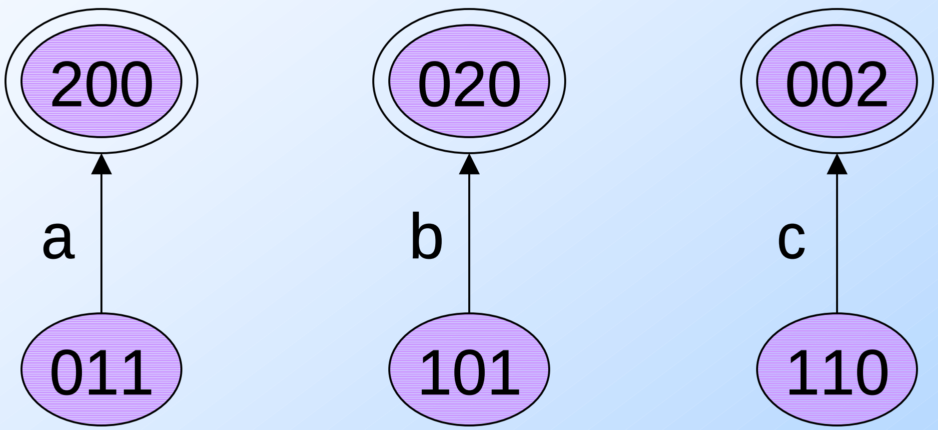
Questions – (2)

- These questions mirror real ones about protocols.
  - "Can the planet fail?" is like asking whether a protocol can enter some undesired or error state.
  - "Must the planet fail" is like asking whether a protocol is guaranteed to terminate.
    - Here, "failure" is really the good condition of termination.

Strange Planet – Transitions

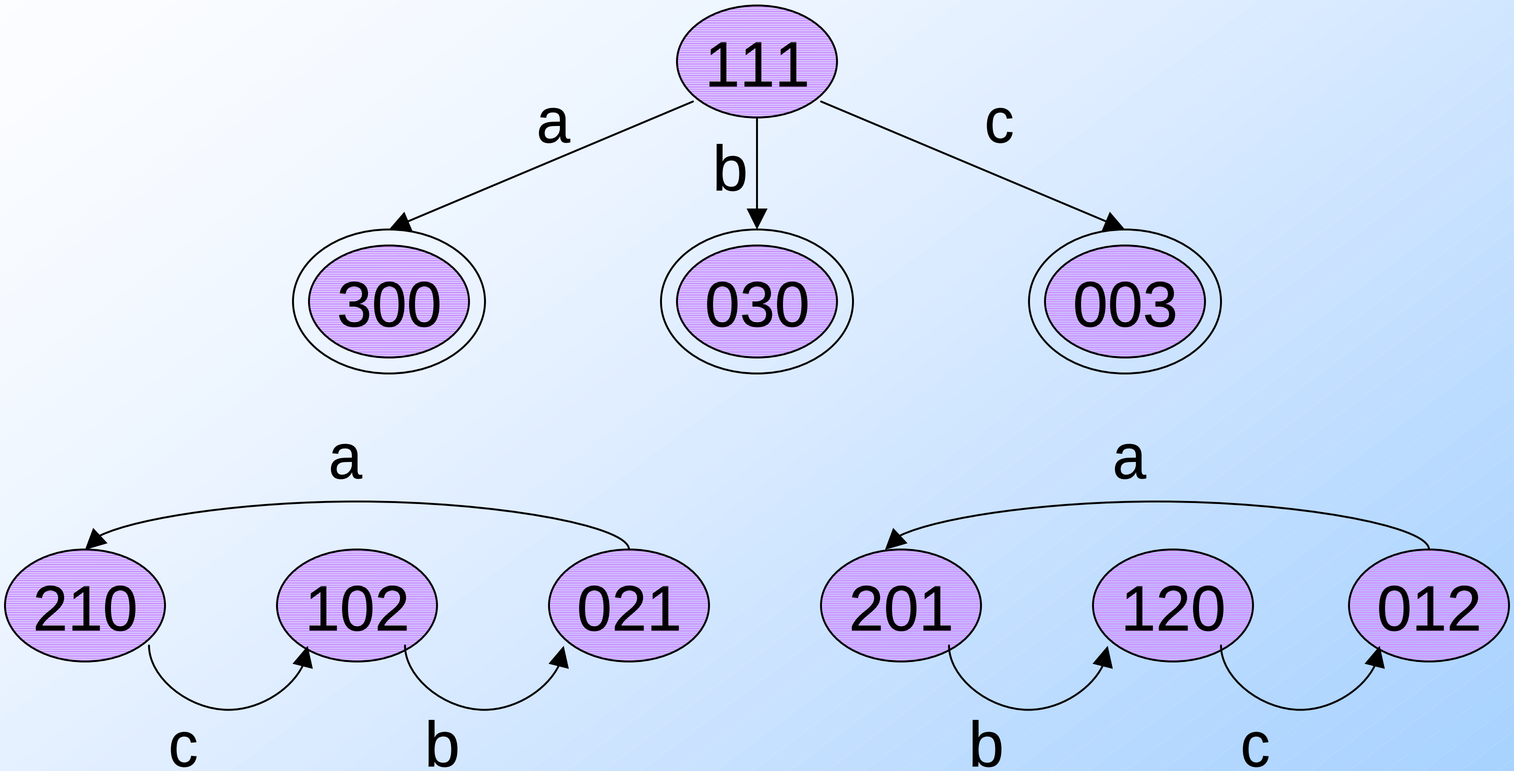
- An *a-event* occurs when individuals of species b and c breed and are replaced by two a's.
- Analogously: b-events and c-events.
- Represent these by symbols a, b, and c, respectively.

Strange Planet with 2 Individuals



Notice: all states are "must-fail" states.

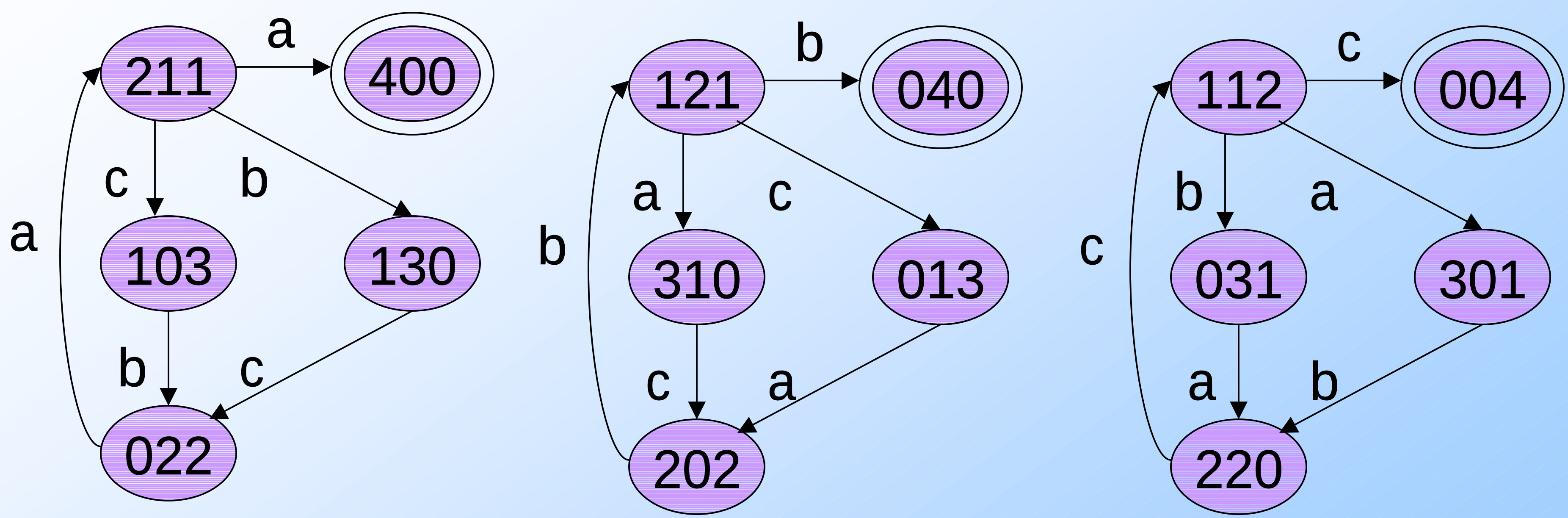
Strange Planet with 3 Individuals



Notice: four states are "must-fail" states.
The others are "can't-fail" states.

State 111 has several transitions.

Strange Planet with 4 Individuals

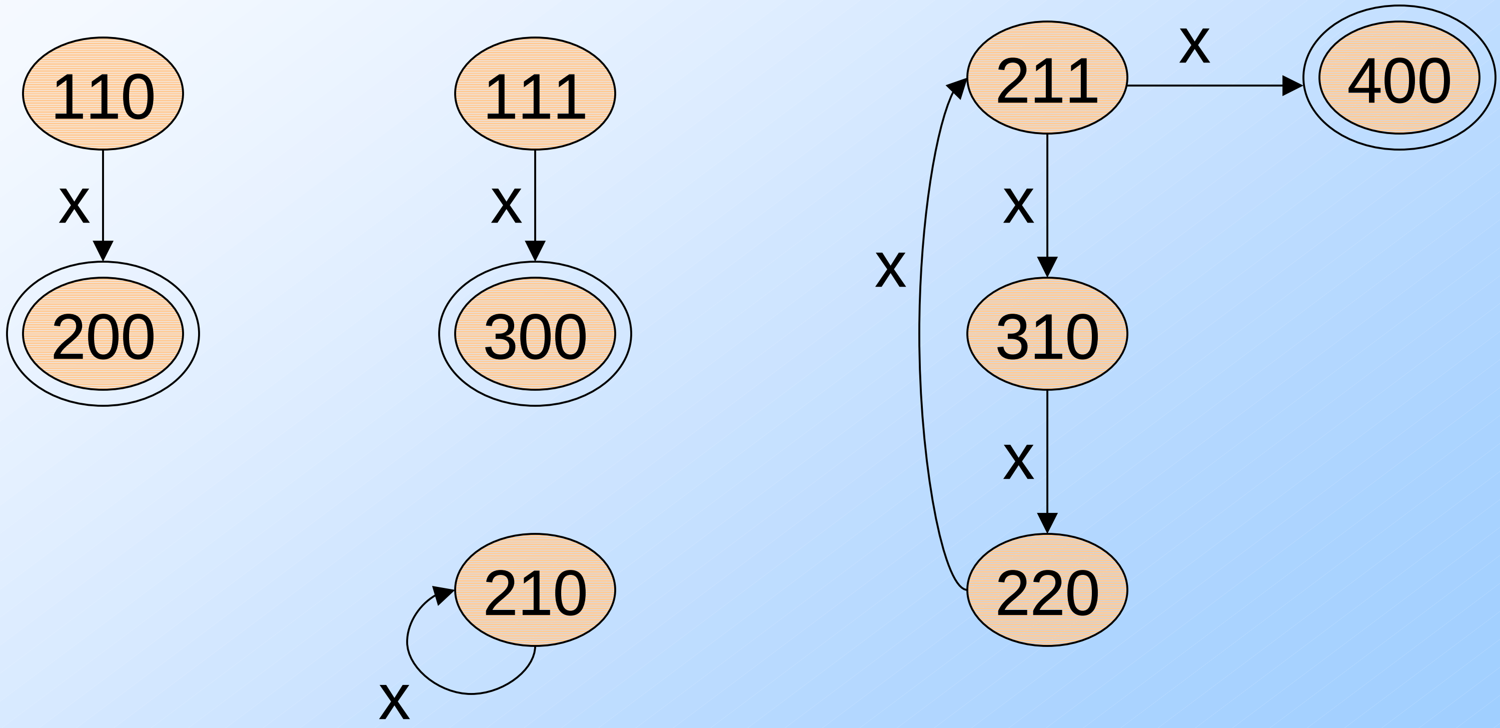


Notice: states 400, etc. are must-fail states.
All other states are "might-fail" states.

Taking Advantage of Symmetry

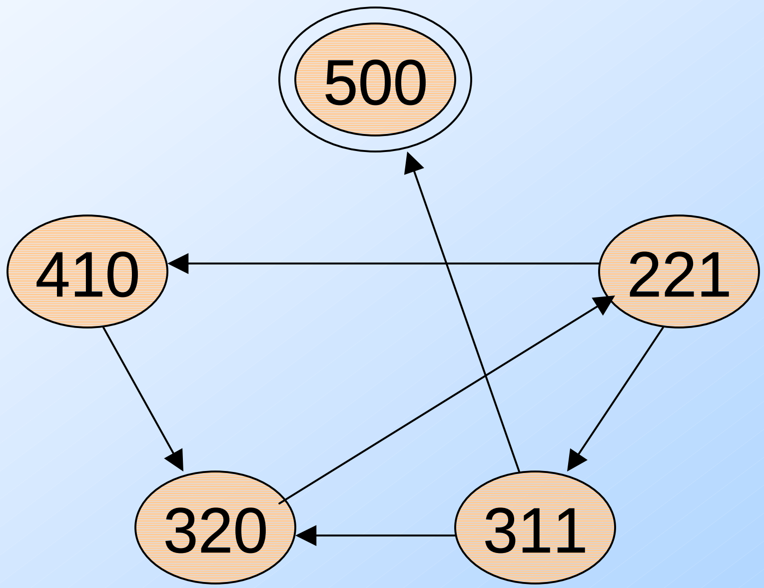
- The ability to fail depends only on the *set* of numbers of the three species, not on which species has which number.
- Let's represent states by the list of counts, sorted by largest-first.
- Only one transition symbol, *x*.

The Cases 2, 3, 4



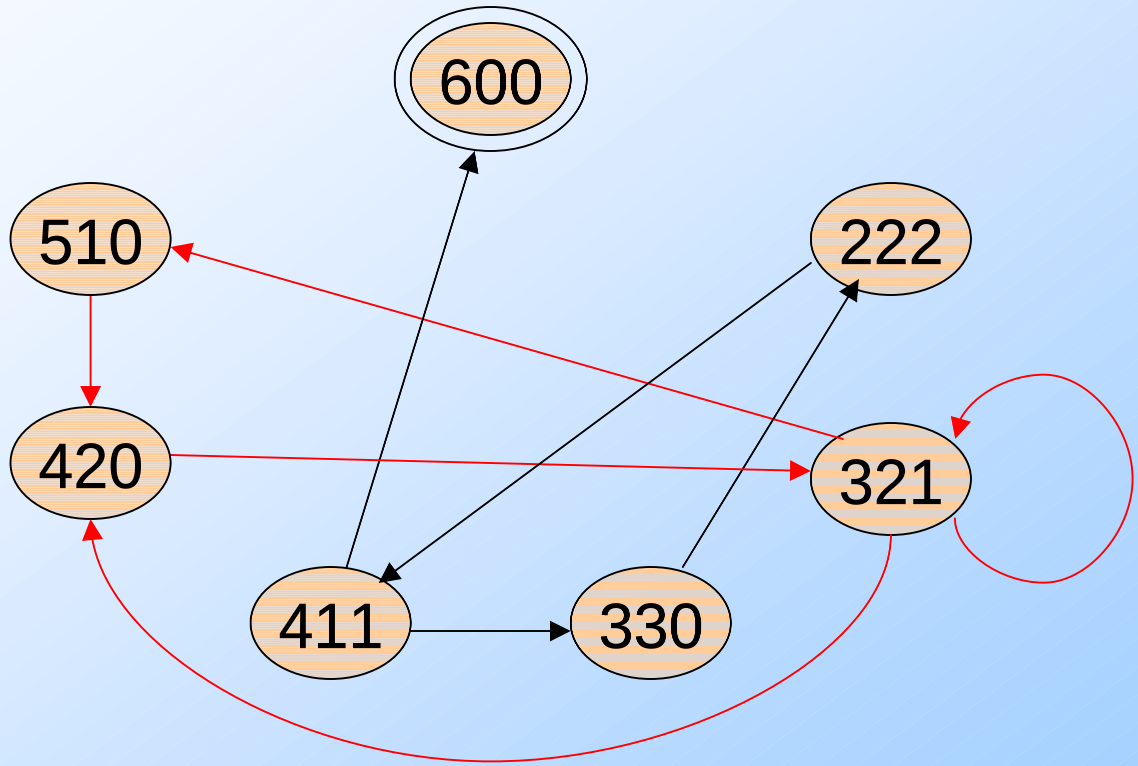
Notice: for the case n = 4, there is *nondeterminism* : different transitions are possible from 211 on the same input.

5 Individuals



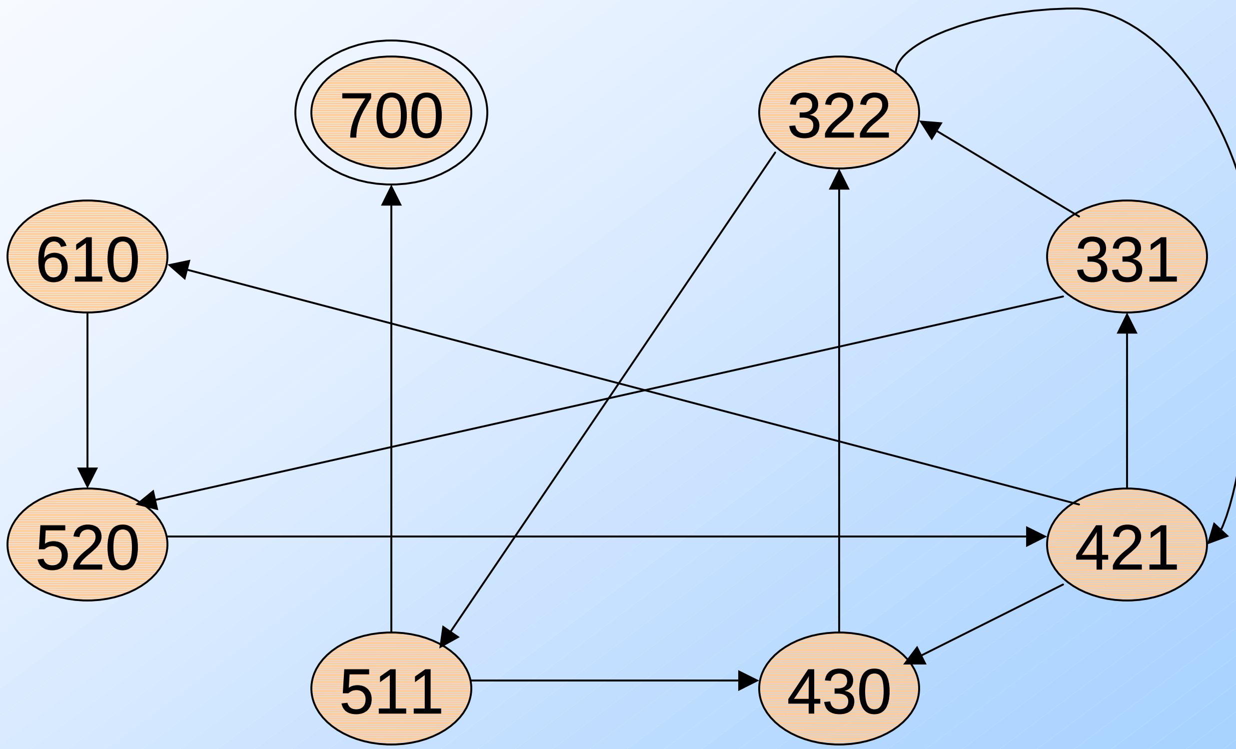
Notice: 500 is a must-fail state; all others are might-fail states.

6 Individuals



Notice: 600 is a must-fail state; 510, 420, and 321 are can't-fail states; 411, 330, and 222 are "might-fail" states.

7 Individuals



Notice: 700 is a must-fail state; All others are might-fail states.

Questions for Thought

1. Without symmetry, how many states are there with *n* individuals?
2. What if we use symmetry?
3. For *n* individuals, how do you tell whether a state is “must-fail,” “might-fail,” or “can’t-fail”?

Deterministic Finite Automata

A DFA is a quintuple

$$A = (Q, \Sigma, \delta, q_0, F)$$

- Q is a finite set of *states*
- Σ is a *finite alphabet* (=input symbols)
- δ is a *transition function* $(q, a) \mapsto p$
- $q_0 \in Q$ is the *start state*
- $F \subseteq Q$ is a set of *final states*

Example: An automaton A that accepts

$$L = \{x01y : x, y \in \{0,1\}^*\}$$

The automaton $A = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_1\})$ as a *transition table*:

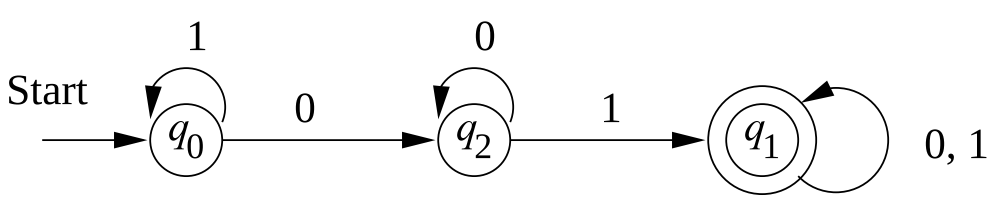
	0	1
$\rightarrow q_0$	q_2	q_0
$\star q_1$	q_1	q_1
q_2	q_2	q_1

$\hat{\delta}(q_0, 00) = q_2$

$\hat{\delta}(q_0, 01) = q_1$

$\hat{\delta}(q_2, 011) = q_1$

The automaton as a *transition diagram*:



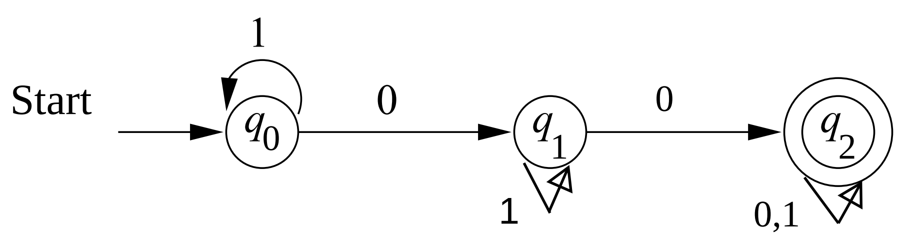
An FA *accepts* a string $w = a_1 a_2 \cdots a_n$ if there is a path in the transition diagram that

1. Begins at a start state

2. Ends at an accepting state
or final

3. Has sequence of labels $a_1 a_2 \cdots a_n$

Example: The FA



accepts e.g. the string 01101 and 1010, but not 110 or 0111

• The transition function δ can be extended to $\widehat{\delta}$ that operates on states and strings (as opposed to states and symbols)

Basis: $\widehat{\delta}(q, \epsilon) = q$

Induction: $\widehat{\delta}(q, xa) = \delta(\widehat{\delta}(q, x), a)$

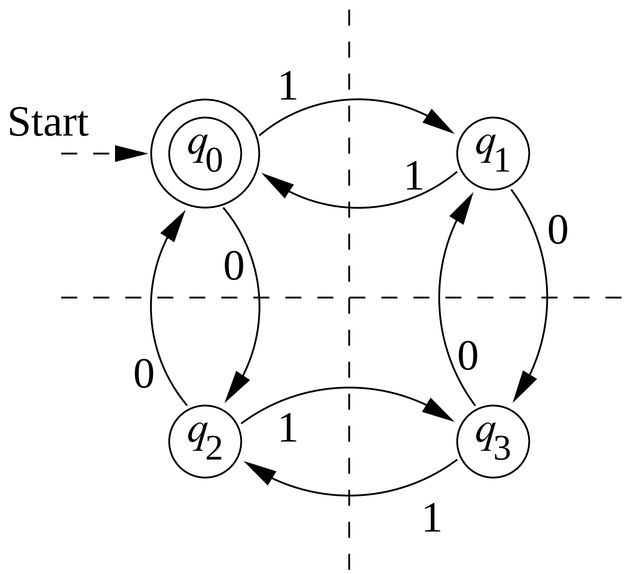
• Now, fomally, the *language accepted by* A is

$$L(A) = \{w : \widehat{\delta}(q_0, w) \in F\}$$

no more!
no less!

• The languages accepted by FA s are called *regular languages*

Example: DFA accepting all and only strings with an even number of 0's and an even number of 1's

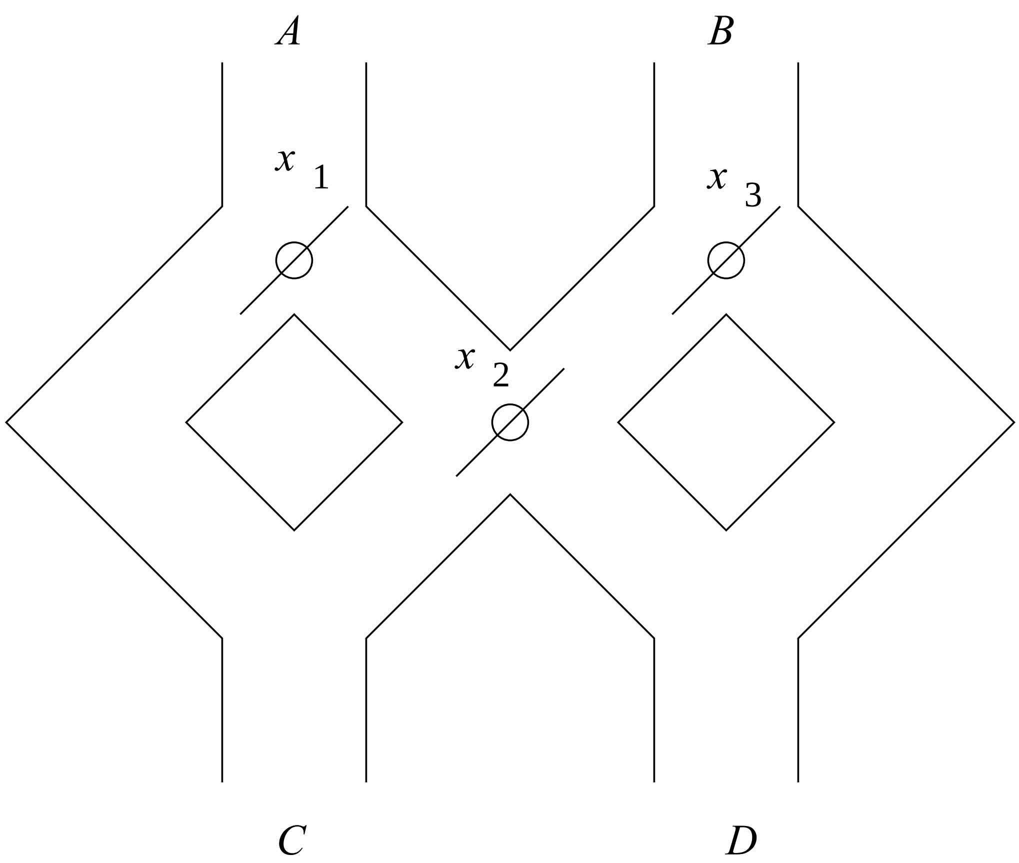


Tabular representation of the Automaton

	0	1
$\star \rightarrow q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Example

Marble-rolling toy from p. 53 of textbook



Ex. L_0 = {binary numbers divisible by 2}
L_1 = {binary numbers divisible by 3}
L_2 = {x | x in {0,1}*, x does not contain 000 as a substring}

A state is represented as sequence of three bits followed by r or a (previous input *rejected* or *accepted*)

For instance, 010a, means
left, right, left, accepted

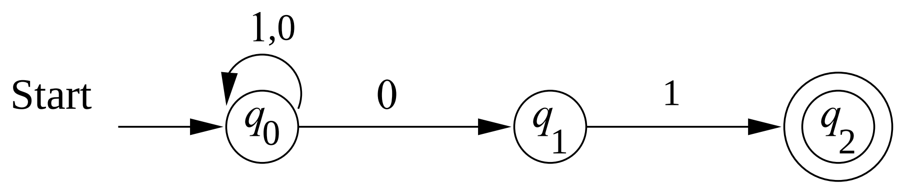
Tabular representation of DFA for the toy

	A	B
$\rightarrow 000r$	$100r$	$011r$
$\star 000a$	$100r$	$011r$
$\star 001a$	$101r$	$000a$
$010r$	$110r$	$001a$
$\star 010a$	$110r$	$001a$
$011r$	$111r$	$010a$
$100r$	$010r$	$111r$
$\star 100a$	$010r$	$111r$
$101r$	$011r$	$100a$
$\star 101a$	$011r$	$100a$
$110r$	$000a$	$101a$
$\star 110a$	$000a$	$101a$
$111r$	$001a$	$110a$

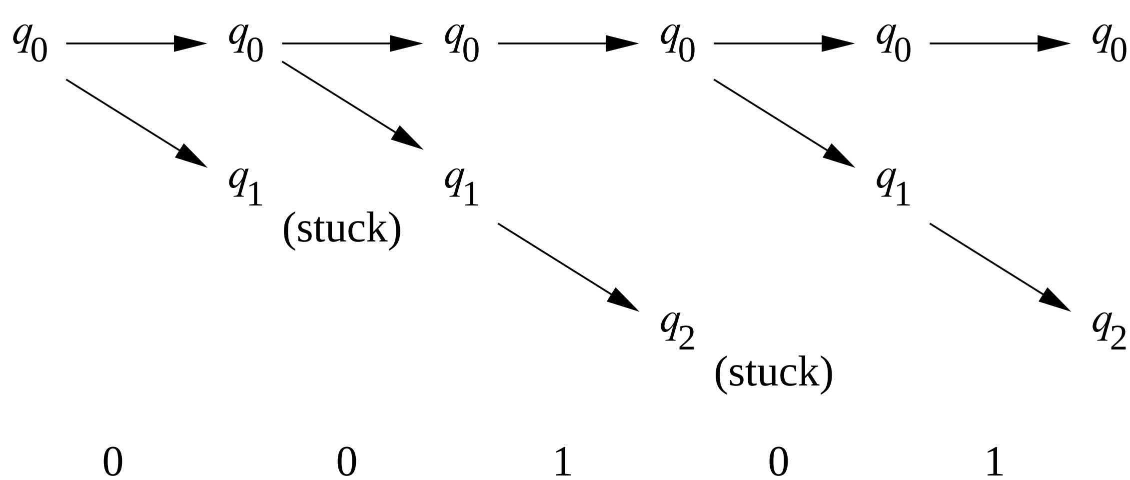
Nondeterministic Finite Automata

An NFA can be in several states at once, or, viewded another way, it can "guess" which state to go to next

Example: An automaton that accepts all and only strings ending in 01.



Here is what happens when the NFA processes the input 00101



Formally, an NFA is a quintuple

$$A = (Q, \Sigma, \delta, q_0, F)$$

- Q is a finite set of states

- Σ is a finite alphabet

- δ is a transition function from $Q \times \Sigma$ to the *powerset* of Q

- $q_0 \in Q$ is the *start state*

- $F \subseteq Q$ is a set of *final states*

Example: The NFA from the previous slide is

$$(\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$$

where δ is the transition function

	0	1
$\to q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
$\star q_2$	$\emptyset$	$\emptyset$

Extended transition function $\hat{\delta}$.

Basis: $\hat{\delta}(q, \epsilon) = \{q\}$

Induction:

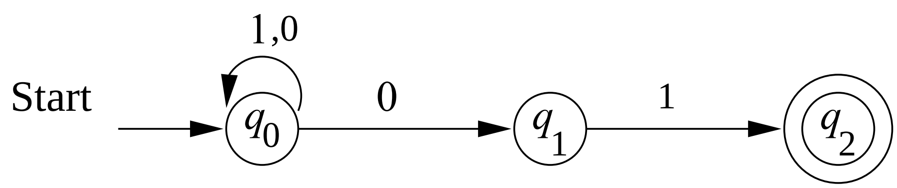
$$\hat{\delta}(q, xa) = \bigcup_{p \in \hat{\delta}(q,x)} \delta(p, a)$$

Example: Let's compute $\hat{\delta}(q_0, 00101)$ on the blackboard. How about $\hat{\delta}(q_0, 0010)$?

- Now, fomally, the *language accepted by A* is

$$L(A) = \{w : \hat{\delta}(q_0, w) \cap F \neq \emptyset\}$$

Let's prove formally that the NFA



accepts the language $\{x01 : x \in \Sigma^*\}$. We'll do a mutual induction on the three statements below

0. $w \in \Sigma^* \Rightarrow q_0 \in \widehat{\delta}(q_0, w)$

1. $q_1 \in \widehat{\delta}(q_0, w) \Leftrightarrow w = x0$

2. $q_2 \in \widehat{\delta}(q_0, w) \Leftrightarrow w = x01$

Basis: If $|w| = 0$ then $w = \epsilon$. Then statement (0) follows from def. For (1) and (2) both sides are false for ϵ

Induction: Assume $w = xa$, where $a \in \{0, 1\}$, $|x| = n$ and statements (0)–(2) hold for x. We will show on the blackboard in class that the statements hold for xa.

Ex. Design an NFA for

L = {x | x in {0,1}*, the 3rd last bit of x is a 1}

How many states would be required in the DFA for L?