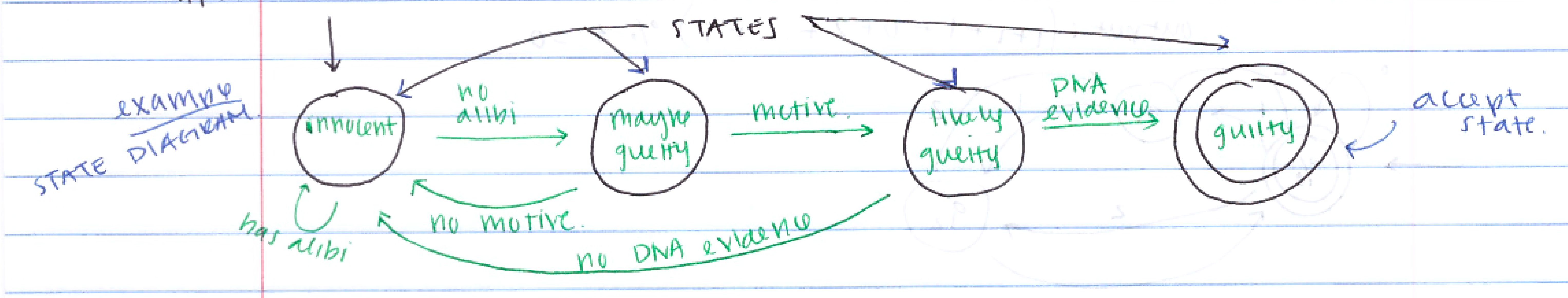
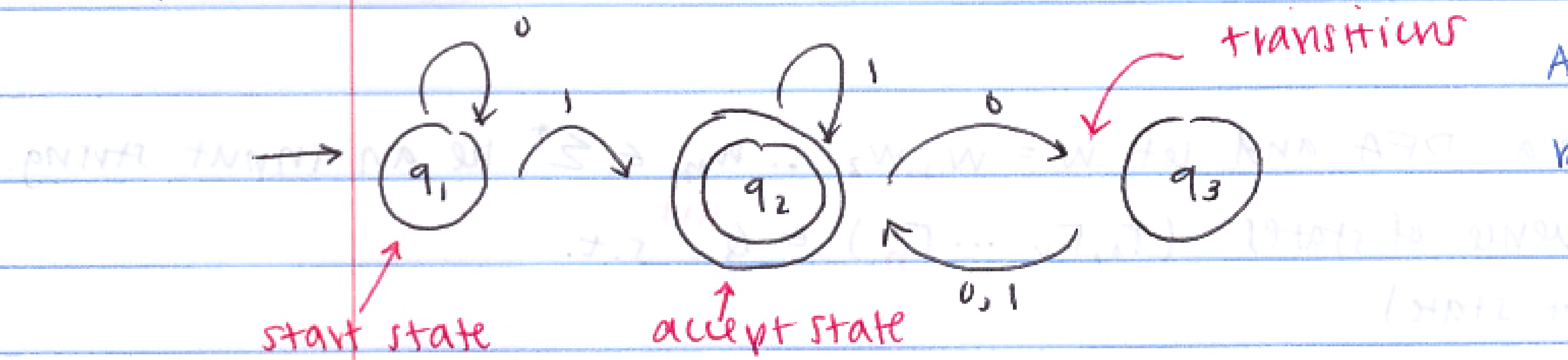
AUGUST 31, 2017.
Bauchau christina Tran.
class Notes.

Finite automata → pushdown automata → Turing machines.

input: alibi? motive? DNA evidence. output = T or F.



formal example



Accepts: 0000001, 01

rejects: ε, 0

such that.

Deterministic Finite automata [DFA]

Definition: A finite automata is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ s.t.:

1) Q is a finite set of states
2) Σ is a finite set called alphabet
3) $\delta : Q \times \Sigma \longmapsto Q$ is transition function.
4) $q_0 \in Q$ is start state
5) $F \subseteq Q$ is a set of accept states

Note: $F = \emptyset$ has no accept states

Example - For our state diagram: $Q = \{q_1, q_2, q_3\}$ $q_0 = q_1$

$\Sigma = \{0, 1\}$ $F = \{q_2\}$

δ	0	1
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_2	q_2

language of a machine M:

don't want to restrict Σ

$L(M) = \{x \mid x \text{ is accepted by } M\} \Rightarrow \boxed{\{x \in \Sigma^* \mid x \text{ is accepted by } M\}}$

example: Entire course: Given $x \in \Sigma^*$ & language L, is $x \in L$?

↳ any YES/NO problem can be phrased this way.

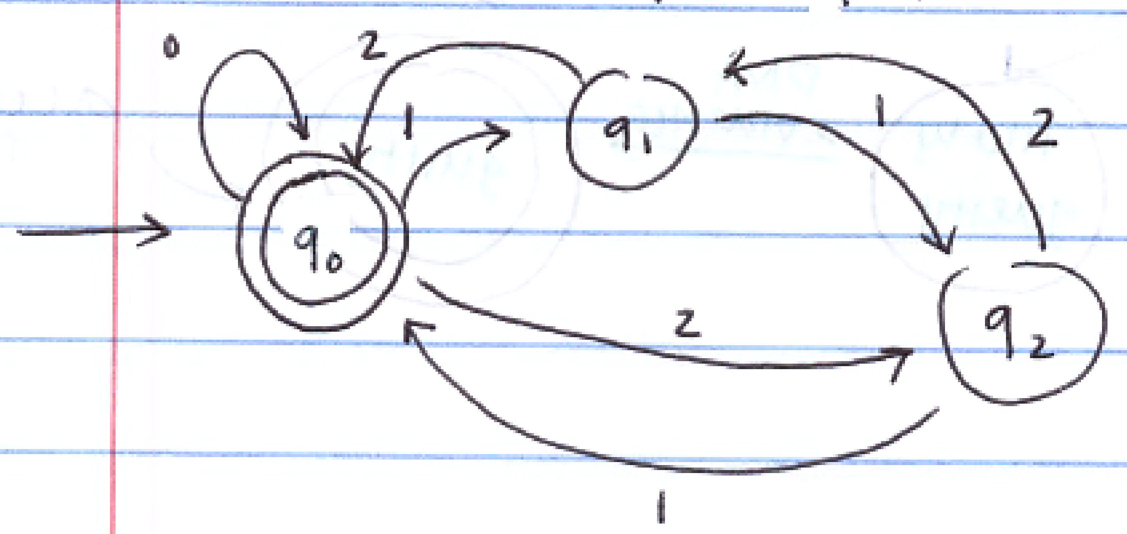
Σ^* - take 0 or more symbols from Σ strings

Design DFA M s.t.

$L(M) = \{x \in \{0,1,2\}^*$ s.t. sum of digits of x is divisible by 3.$\}$

↳ comp. prob : input : 111 02 ... 1

output : $1+1+1+0+2+\ldots+1 \ \% \ 3 = 0$



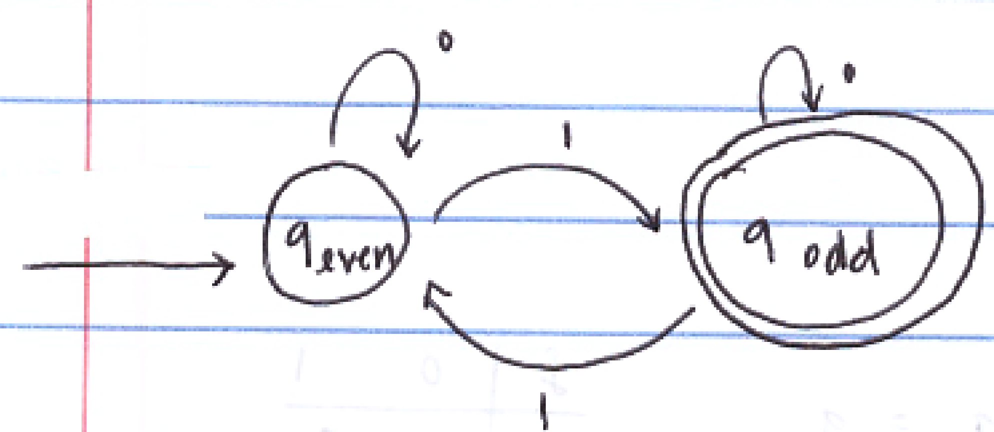
Formalizing the accept criterion

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA and let $w = w_1 w_2 \ldots w_n \in \Sigma^*$ be an input string.
Then M accepts w if $\exists$ a sequence of states $(r_0, r_1, \ldots r_n) \in Q^{n+1}$ s.t.

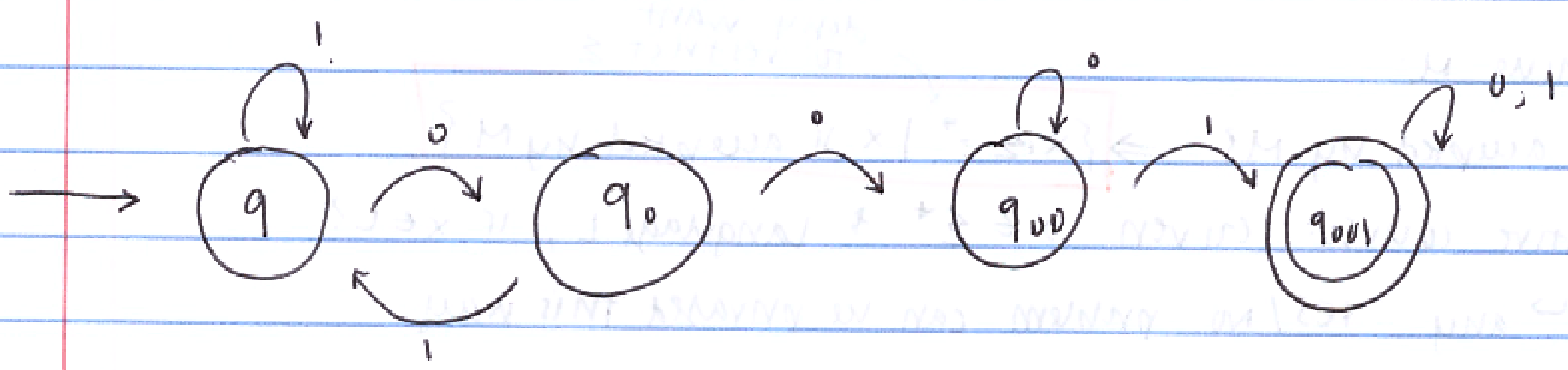
1) $r_0 = q_0$ (start in start state)
2) $\delta(r_i, w_{i+1}) = r_{i+1}$ for $i = 0, \ldots, n-1$ (obey transition function)
3) $r_n \in F$ (end in an accept states)

Designing DFAs

a) $L(M) = \{x \in \{0,1\}^* \mid x$ has odd # of 1's.$\}$



b) $L(M) = \{x \in \{0,1\}^* \mid x$ contains substring 001$\}$ input = 110100$\underline{1}$110



Regular languages: language L is regular if $\exists$ DFA M recognizing L, i.e. $L(M) = L$

regular languages

Q: Given 2 regular languages A & B, which operations applied to A & B produce a new regular language C?

Regular operations

1. union: $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$
2. concatenation: $A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$
3. star: $A^* = \{x_1 x_2 \ldots x_k \mid k \geq 0 \text{ and } x_i \in A \ \forall \ 1 \leq i \leq k\}$

Note: $\varepsilon \in A^* \ \forall A$ (pick $k=0$)

example

$S = \{a, b\}$ $T = \{0, 1\}$, $\Sigma = \{a, b, 0, 1\}$

finite $\begin{cases} S \cup T = \{a, b, 0, 1\} \\ S \circ T = \{a0, a1, b0, b1\} \end{cases}$

infinite $\{$ $S^* = \{\varepsilon, a, b, aa, aaa \ldots, bb, abab \ldots, \ldots\}$