CPSC 421/501, Oct 7, 2021

"Now we will use extension-by-zero, and you will sneer at it."

—

"Last class I mentioned extension-by-zero, and you sneered at it."

—

Running joke of Prof. Raoul Bott (1923 - 2005)

Is there a sub-linear time algorithm to factor an integer?

What is a "sub-linear time algorithm"?

Input:

7543127189

← factor?

just reasonable to look at each digit

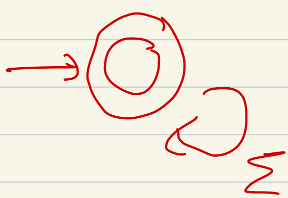
1^{st} pass: you don't actually have to look at all input digits!

e.g., $L = \{ s \in \Sigma^* \mid |s| \geq 2 \}$

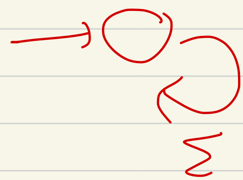
$L = \Sigma^*$ or $L = \emptyset$

DFA with one state

Σ^*

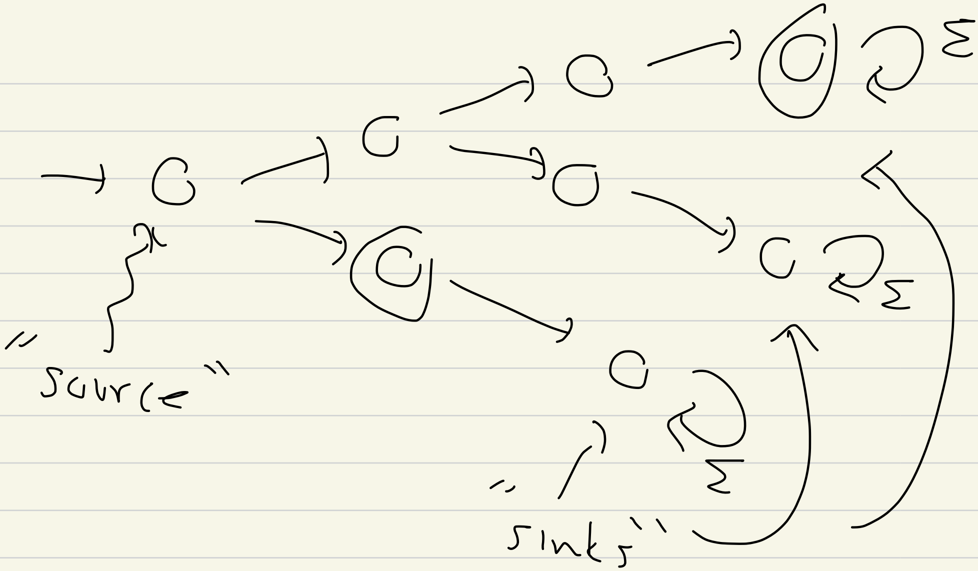


$\emptyset$



abbreviate

Say DFA is a partial ~~pe~~ order !



Could you factor a number $\mathbb{N} = \{1, 2, 3, \dots\}$ in sublinear time, or solve problem where you have to examine all the symbols/letters in the input strings/words ???

In Ch. 7, we talk about

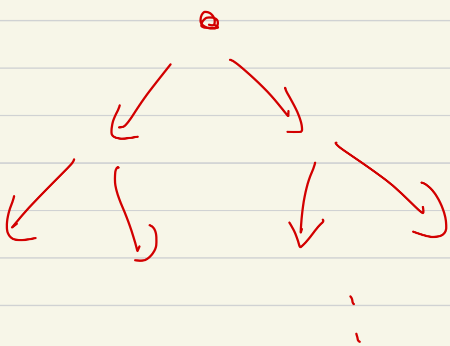
log-space reductions

poly-time "

=

You could talk about trees

representing computation



depth tree

Idea!

you can read input

input appears here $\underbrace{72158329}_{\text{string of size } n}$

you can read "input tape"

computation:

- DFA
- NFA
- Turing machine

Ch 1: "input"

poly log(n) computation

Could you factor an integer in sub-linear time?

What if: 12 base 10

you write

a a a a a a a a a a a a

= 12 in unary

work space

12 in decimal writes then run usual naive factoring...

Unary! always "bloats" the input size...

Do not <u>sneer</u> at

$$\Sigma = \{a\}$$

a single-(letter / symbol) alphabet

We'll spend more time this year on

$$\Sigma = \{a\}$$ single letter

CPSC 421/501 Oct 7

- My office hours now 3:45-5:15pm on Tuesday.

- NEW to 2021:

More discussion of DFA's and regular languages over $\Sigma = \{a\}$.

(Don't sneer at a one-letter alphabet)

(Wait until we make use of unary notation to give a short proof that

NP-SNEAKY

$=$

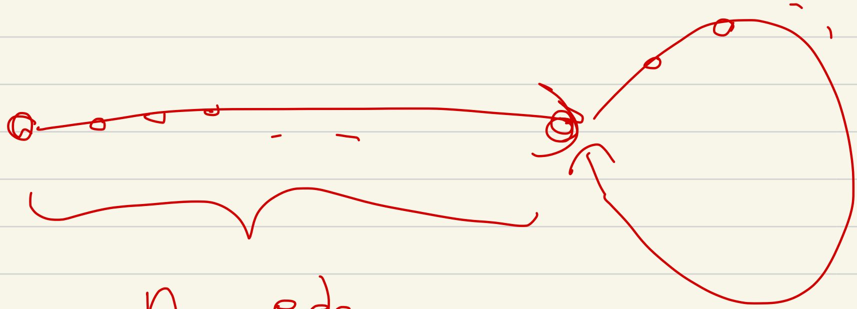
$\{ \langle M, w, 1^t \rangle \mid$ M is a non-det TM that accepts w within time t $\}$

is NP-complete)

t expressed in UNARY, i.e. over a one-letter alphabet. Does not work in binary, base 10, etc.

Review

DFA on $\Sigma = \{a\}$



n_0 edges

period p

\# states $= n_0 + p$

Today: mostly talk about

$\{a^3, a^5\}^*$ set

something regular but could requires

many more states.

<u>Thm</u> If L is regular, then so is L^*.

The most convenient to prove this is non-deterministic FA, NFA's (Section 1.2).

Last time:

DIV-BY-3: over

$\Sigma = \{0, 1, \ldots, 9\}$.

You could say DIV-BY-3

dont have ε

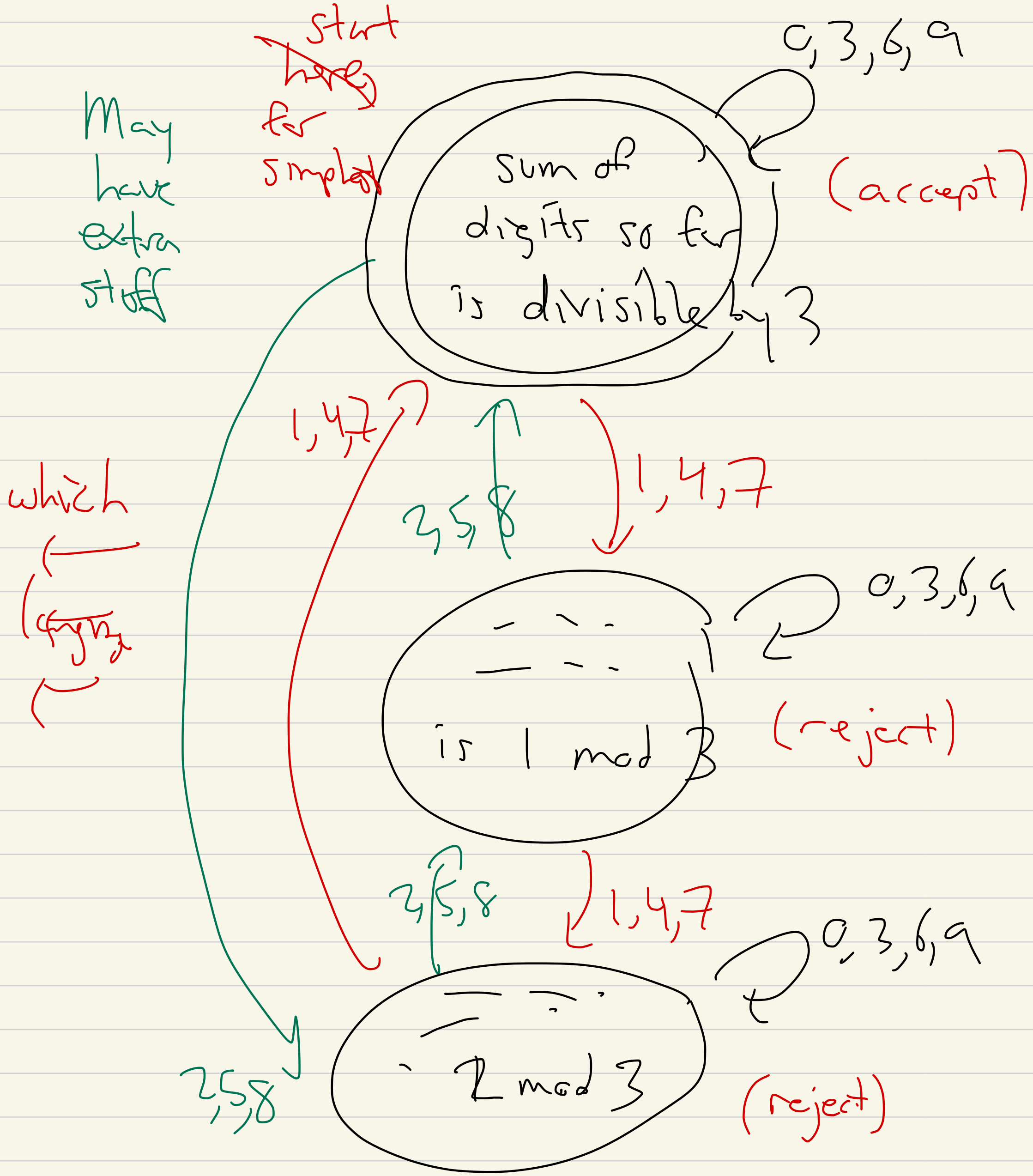
$\{3, 6, 9, 12, 15, 18, \ldots\}$

$\{0, 3, 6, 9, 12, \ldots\}$

$\{\varepsilon, 0, 3, 6, 9, \ldots\}$

$\{\varepsilon, 0, 3, 6, 9, 03, 06, 09, 12, \ldots\}$

Simplest machine



5 min break, 10:12 — 10:17,

Group HW #4!

6.1.1 – 6.1.5, EXERCISE section in Myhill-Nerode handout

Which is simpler?

$\{ \epsilon, a^3, a^6, a^9, \ldots \}$

or

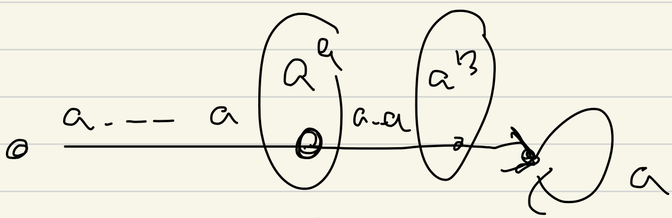
$\{ a^3, a^6, a^9, \ldots \}$

Roughly analogue of DIV-BY-3 question

Question:

$L = \{a^9, a^{13}\}$ looks

easy to understand ...

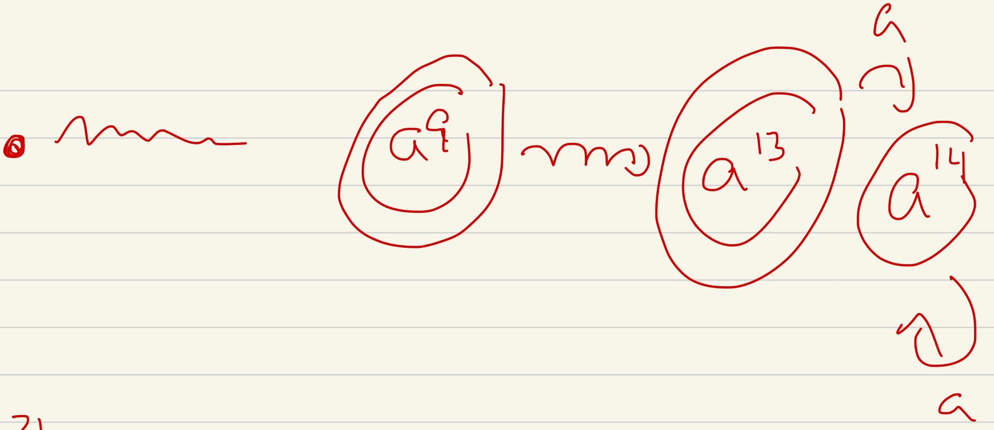


14 edges

what about recognizing

$L^* = \{a^9, a^{13}\}^*$ as a

DFA



a^{31} $\in L$?

$\notin L$?

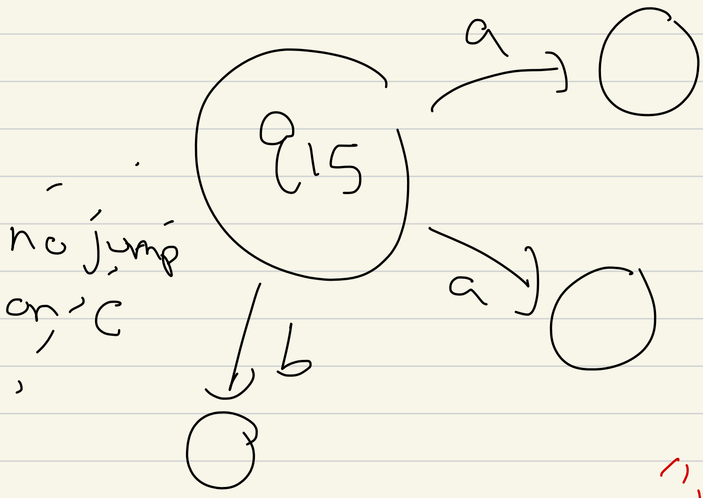
a^{95} $\in L$?

$\notin L$?

$L^* = \{$ words that are concatenations of (a^9)'s (a^{13})'s $\} =$???

Say that

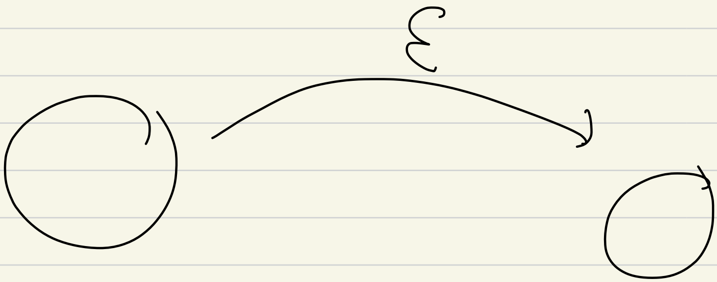
1. we can jump to more than one place on a given symbol, or to places



"non-determinism"

$\Sigma = \{a, b, c\}$

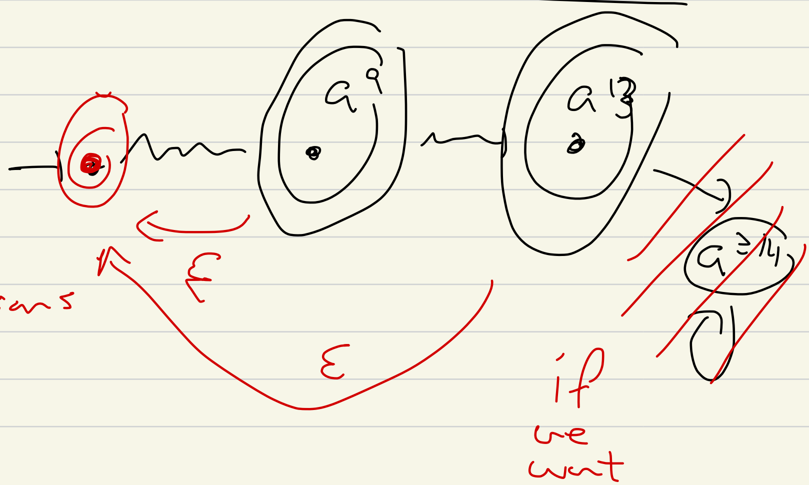
(2) also we have an ε-jump



meaning you reading anything

So

L :

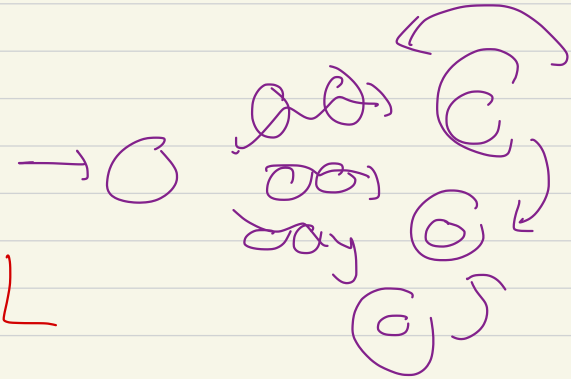


New conventions

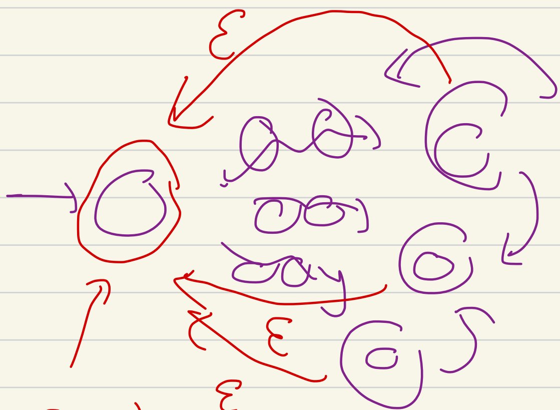
L^*

More generally:

if L has DFA:



L



L^*

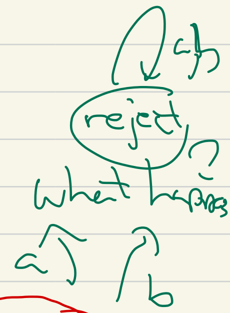
empty word

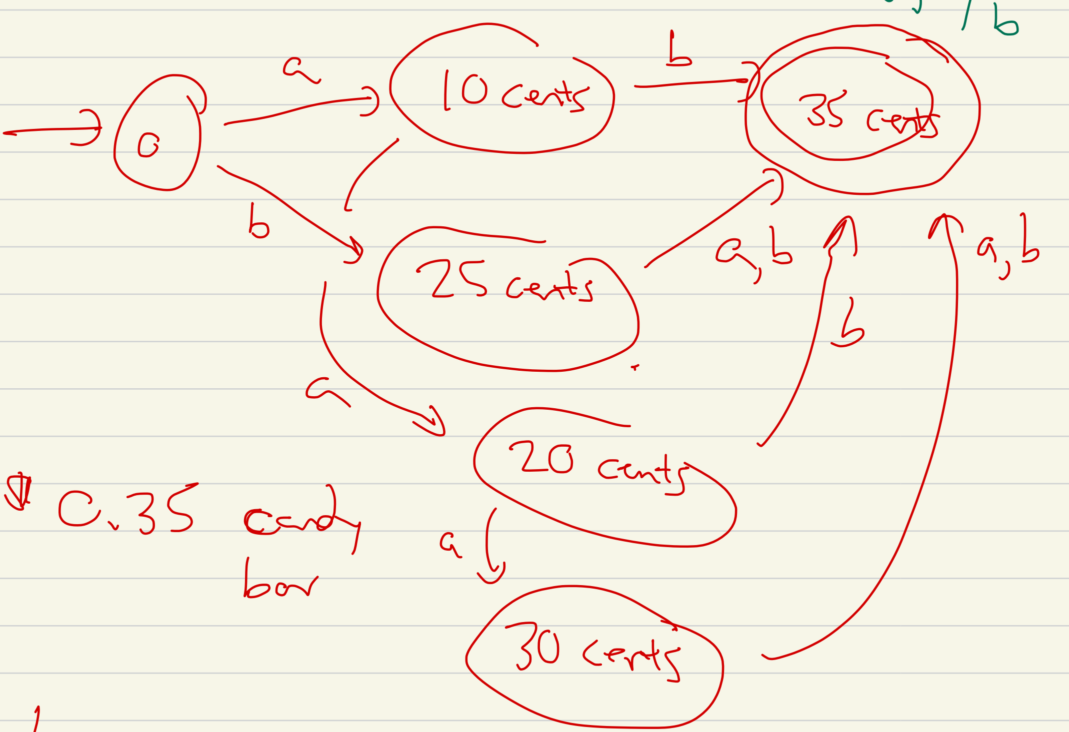
$\varepsilon \in L^*$

Example 2: Vending Machine

Input coins \$0.10 = a

\$0.25 = b

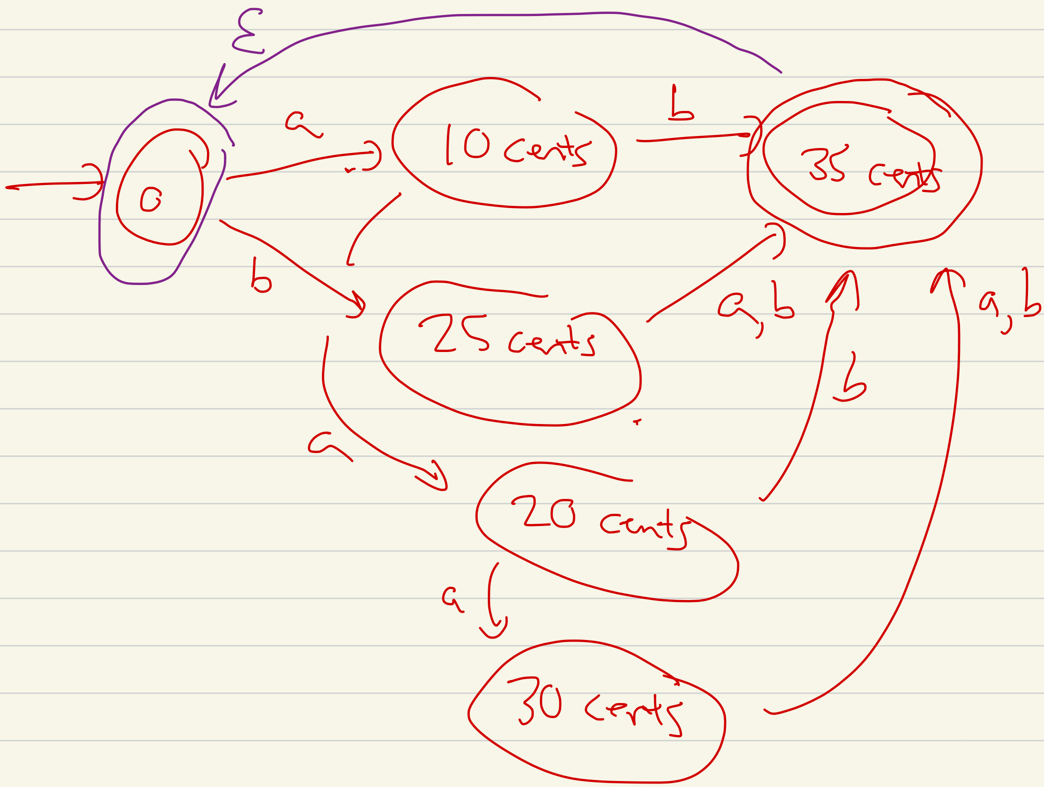


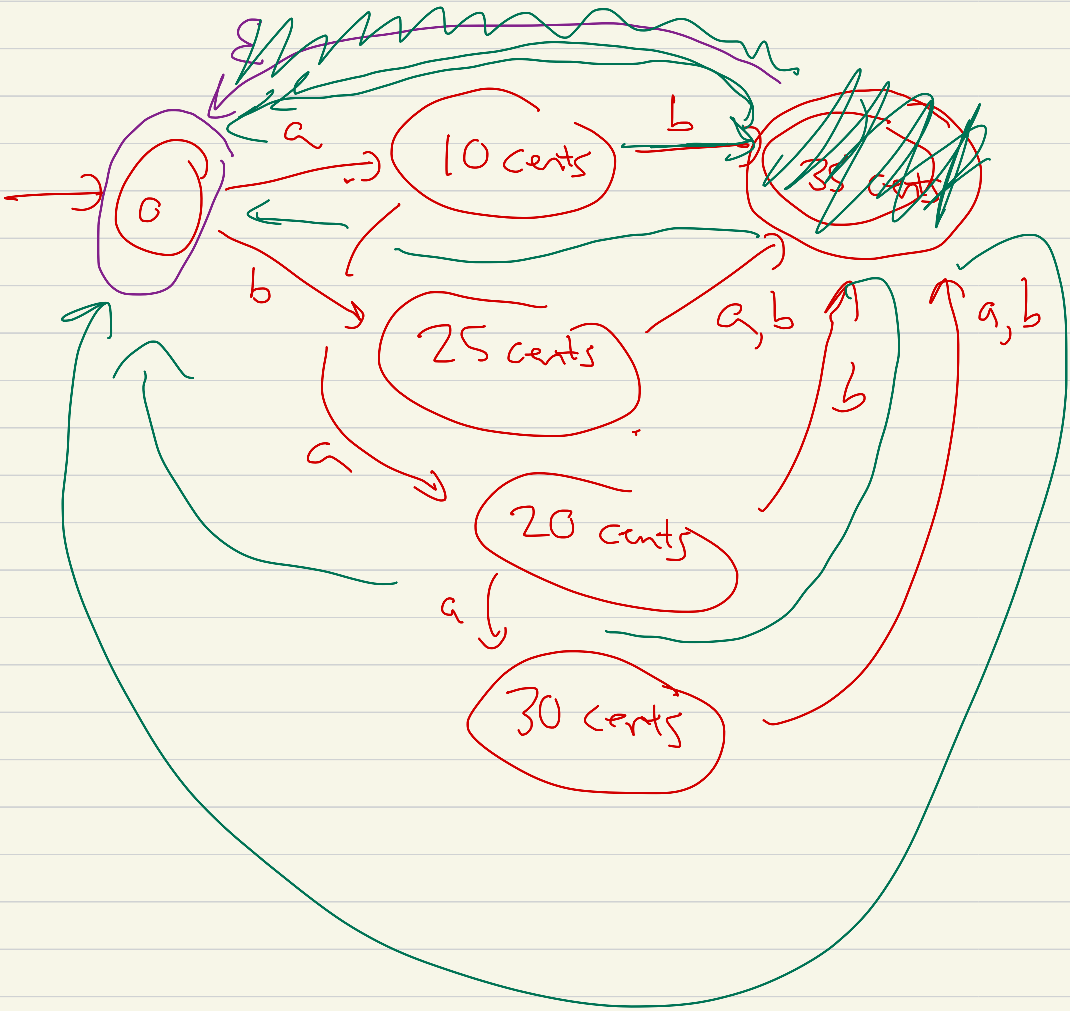


\$0.35 candy bar

L

$L^* = \{$ words in $\{a, b\}^*$ above where machine is not asking for more money, i.e. showing 0 cents $\}$





0
10 cents
25 cents
20 cents
30 cents
35 cents
a
b
a
b
a, b
a
a
b
a, b