HW 6 FA '25

①

$(q_0, bacccddf, \varepsilon)$

$(q_1, bacccddf, \$)$

$(q_2, acccddf, N\$)$

$(q_1, acccddf, NN\$)$

$(q_2, cccddf, NNN\$)$

$(q_1, cccddf, NNNN\$)$

X

$(q_3, cccddf, NNN\$)$

$(q_4, cccddf, NNNN\$)$

$(q_4, ccddf, NNN\$)$

$(q_4, cddf, NN\$)$

$(q_4, ddf, N\$)$

X

② Give the set notation for PDA A

See #8

③ Draw the state machine for

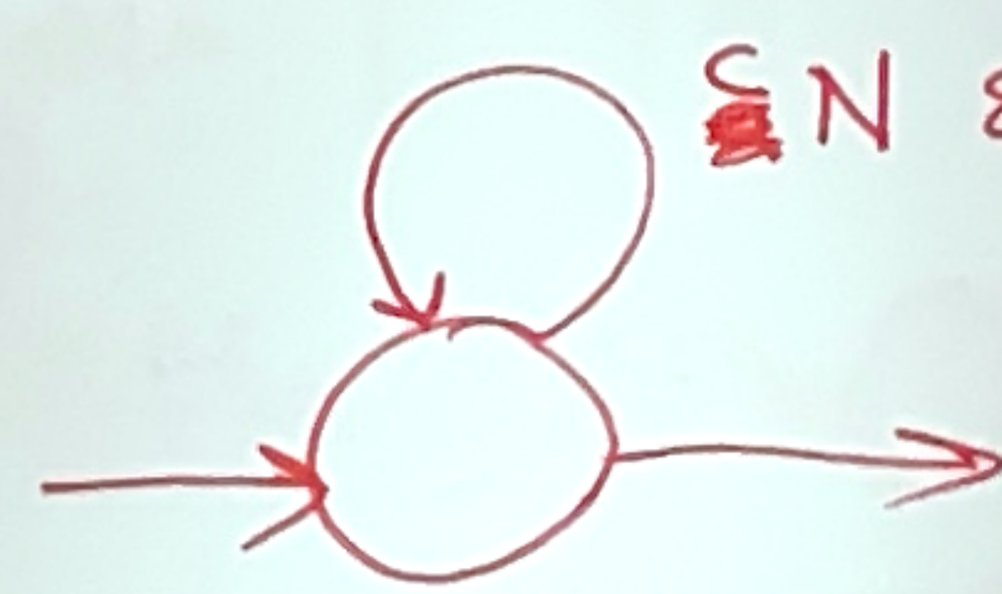
$$B = \{ w \text{ over } \{a, b, c\} \mid a^n b c^{n+2},\ n \geq 0 \}$$

First, observations:

- order is set (a's before b's before c's)
- there's only one b, and must be one b
- there's a relationship between the a's & c's
- the number of c's can be rewritten as $c^n c^2$
- there can be zero or more a's & c's AND there must be the two additional c's

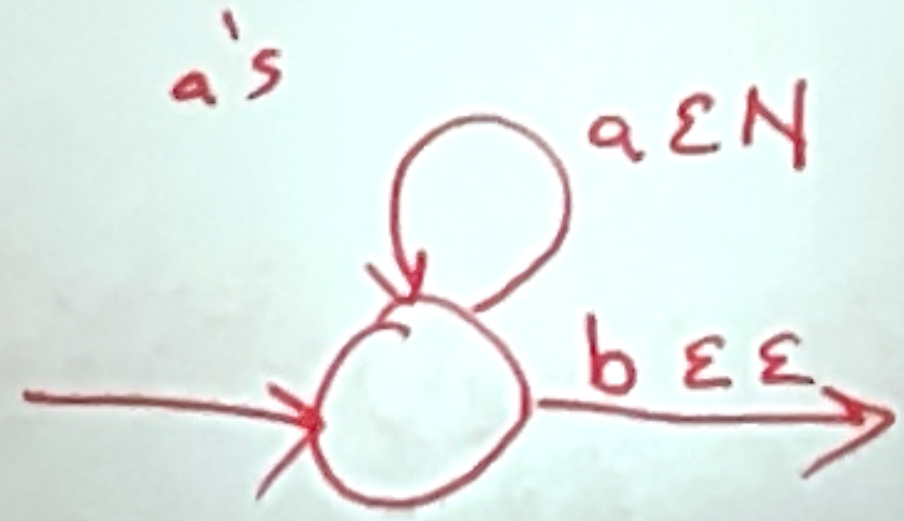
So, the state machine needs to count a's and check for c's

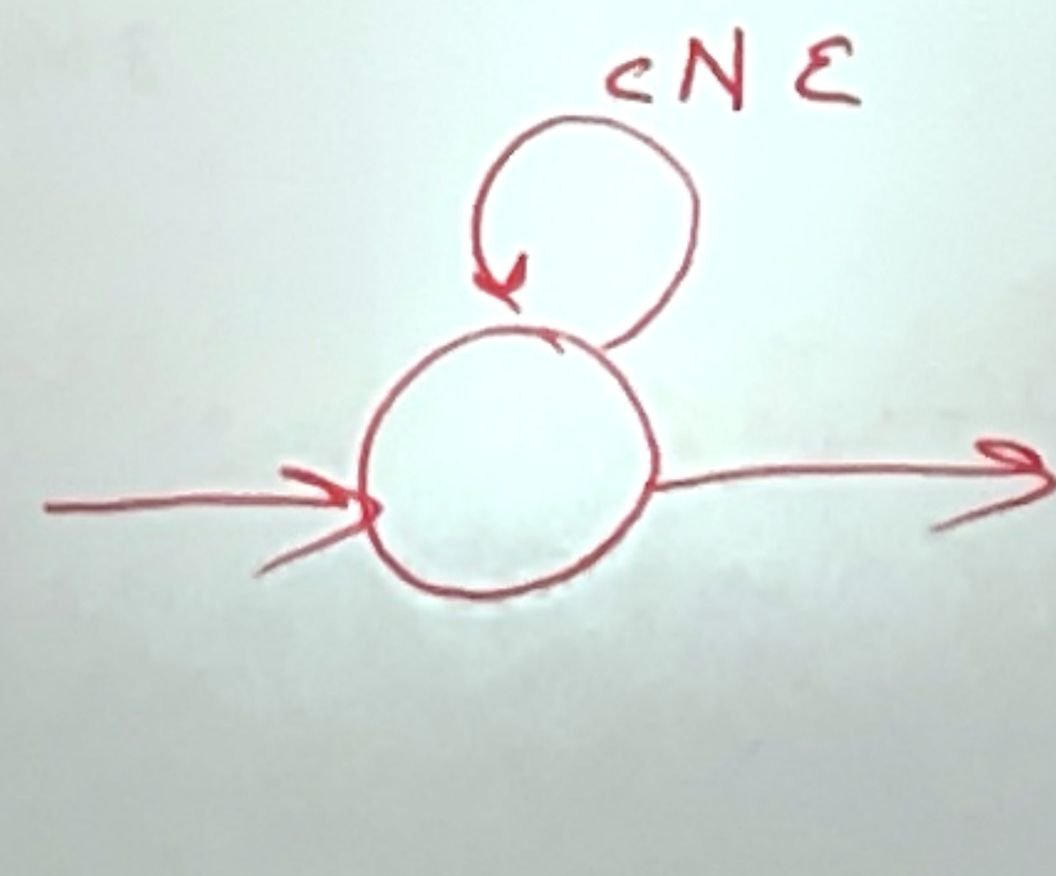
//count a's



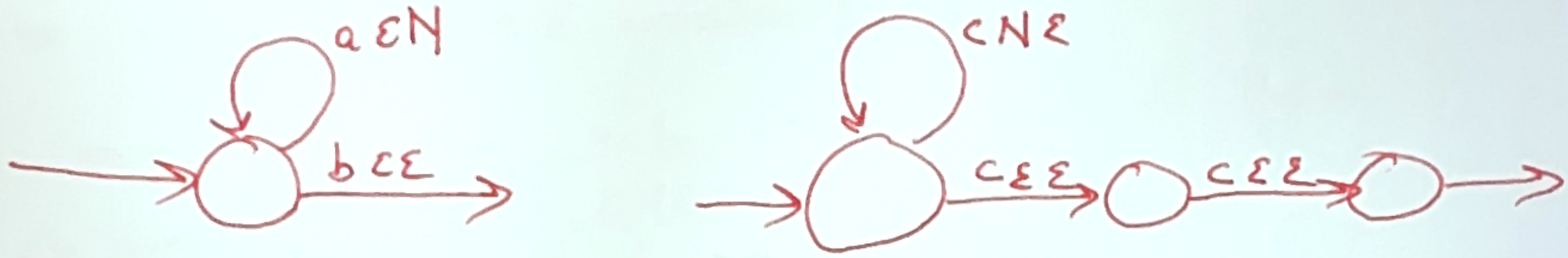
//check c's

The single "b" will go on a transition after counting the a's

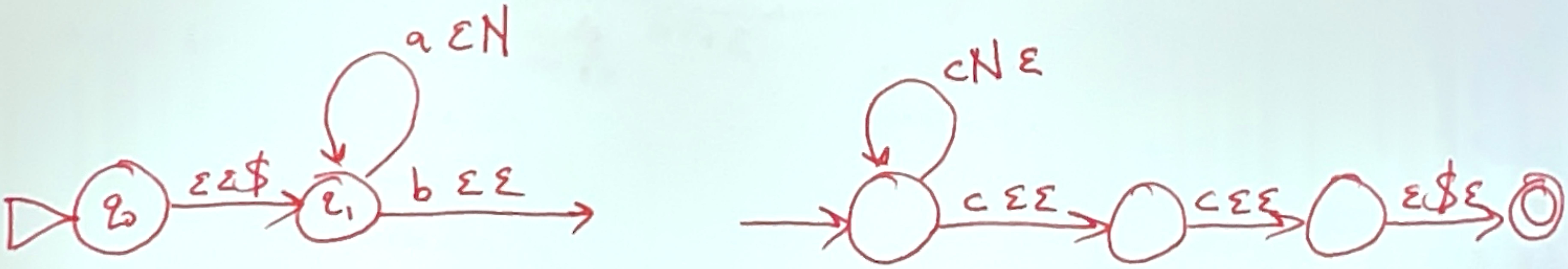




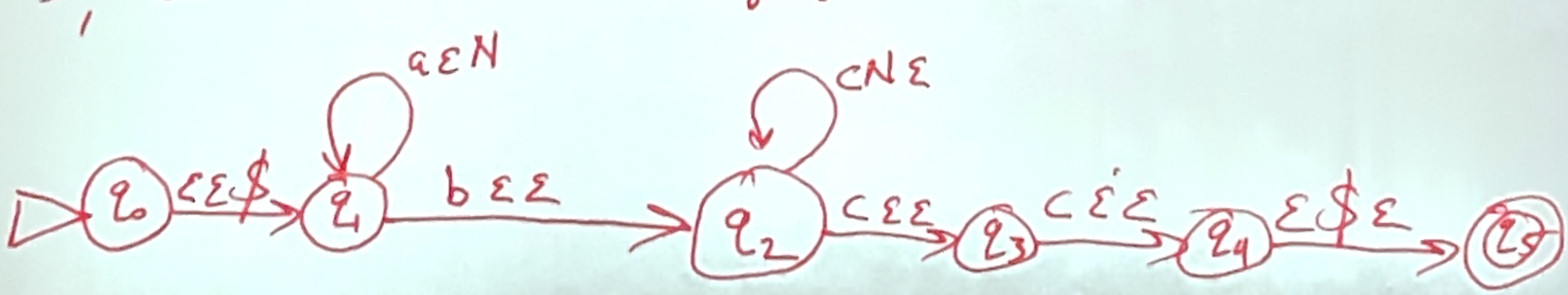
The two additional c's can just go on transitions either before or after the "checking for c's" self-loop.



Now we need a way to be sure we have the same # a's and c's. I need to know that I've had a "c" for each "a". Will do that by pushing on a $ first thing and checking for it at the end.



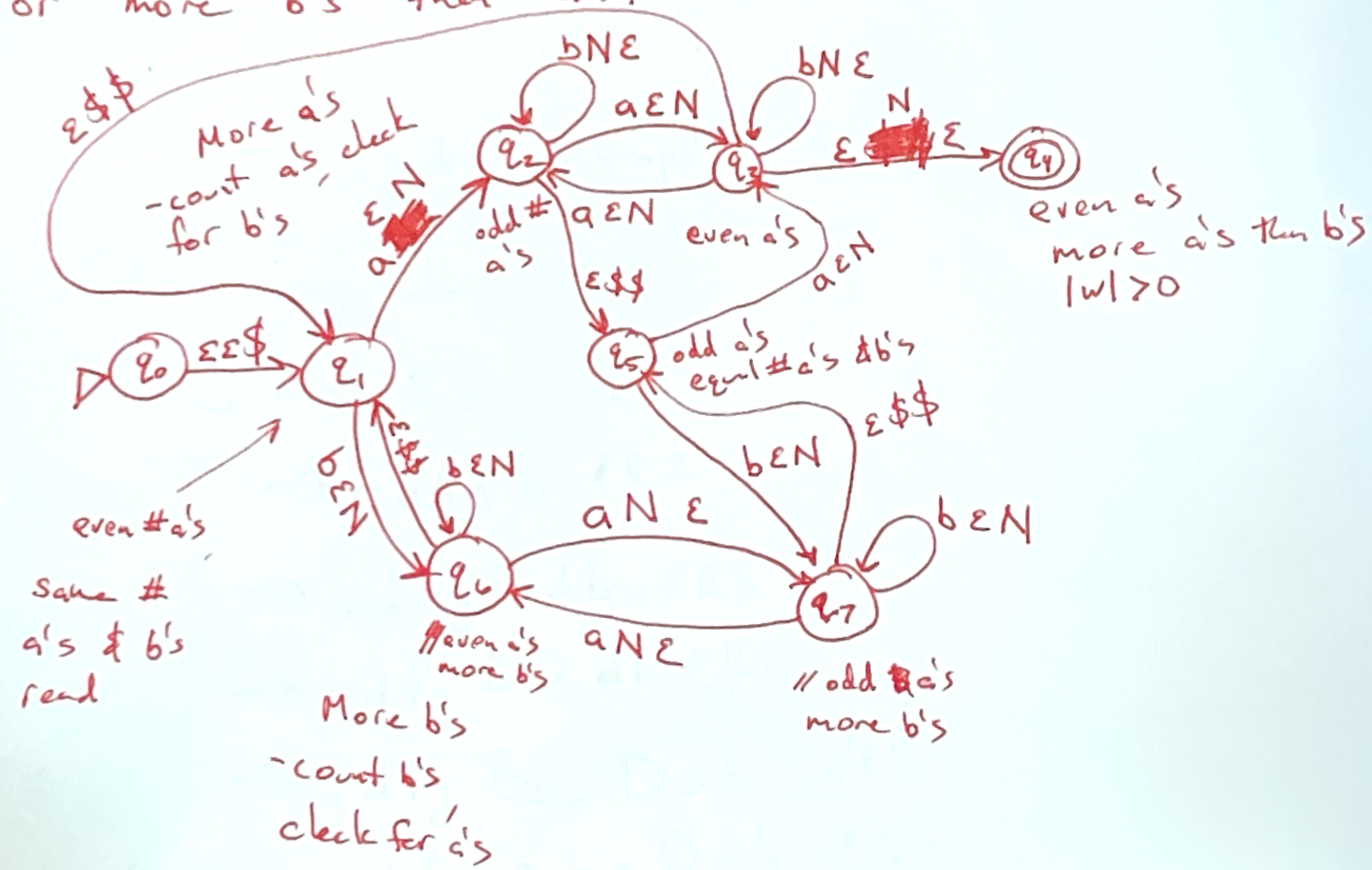
There is nothing else between the "b" and the start of the c's, so I can hook everything up!



FINAL ANSWER

4\) $|w| > 0$, w has more a's than b's, #a's is even

Notice: the a's & b's can be in any order. At any point in reading the string there can either be the same number of a's and b's consumed so far, more a's than b's, or more b's than a's. Consider each of these...



A few notes

q_5: ~~why does q_5 only have an outgoing edge on b? If the next character read is an "a" that will be the case of an even # of a's and more a's. That is already accounted for by $q_2 \to q_3$ on an "a".~~

⑤ Shortest string derivation

$S \rightarrow dAd$ // R2

$\rightarrow dBDd$ // R4

$\rightarrow d\varepsilon Dd$ // R5

$\rightarrow d\varepsilon kd$ // R9

$\rightarrow dkd$ // simplify ε

⑥ Use any rule

$S \rightarrow aSb$ // R1

$\rightarrow adAdb$ // R2

$\rightarrow adfgAdb$ // R3

$\rightarrow adfgBDdb$ // R4

$\rightarrow adfgBbcDdb$ // R5

$\rightarrow adfg\varepsilon bcDdb$ // R6

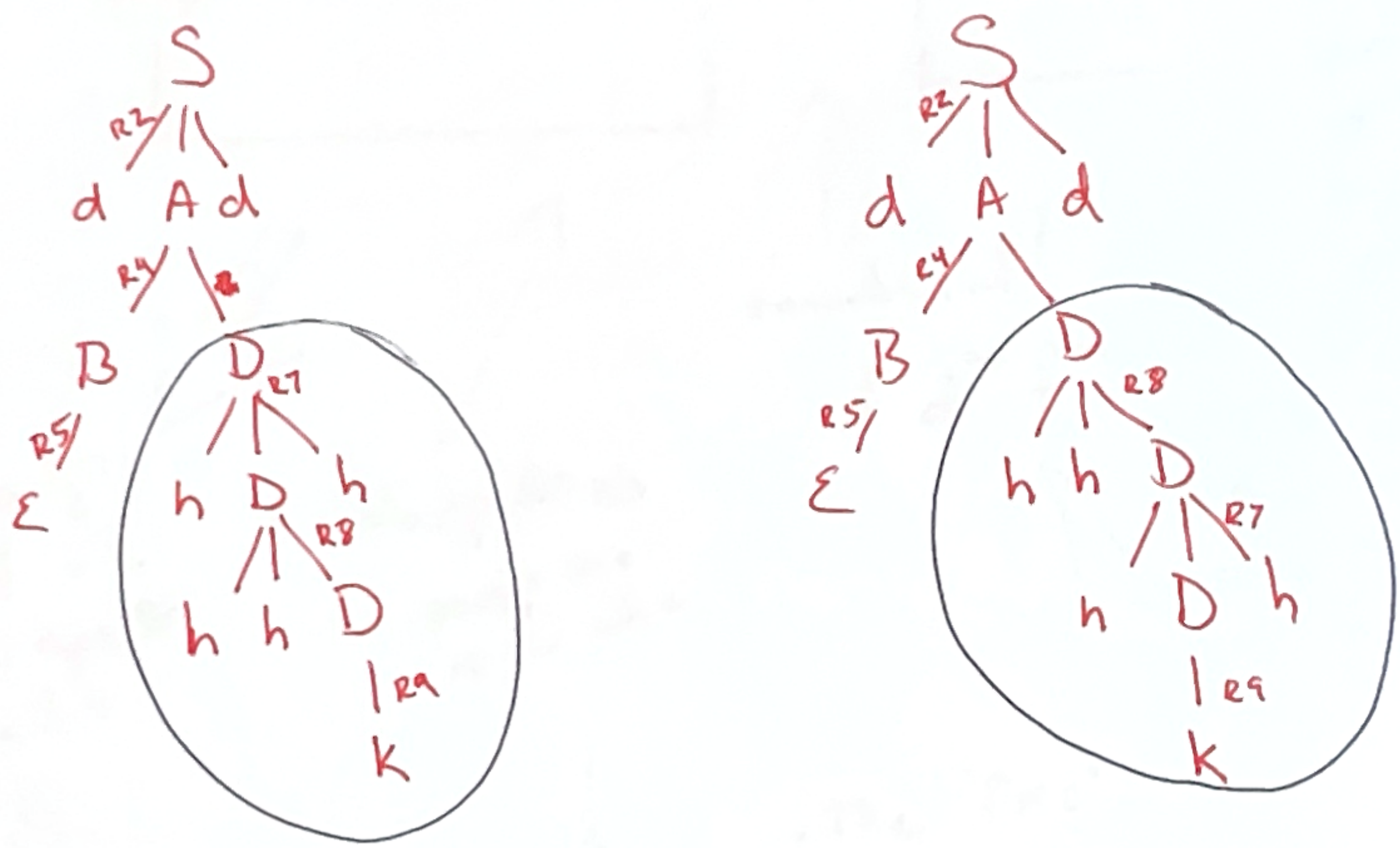
$\rightarrow adfg\varepsilon bchDhdb$ // R7

$\rightarrow adfg\varepsilon bchhhDhdb$ // R8

$\rightarrow adfg\varepsilon bchhhkhdb$ // R9

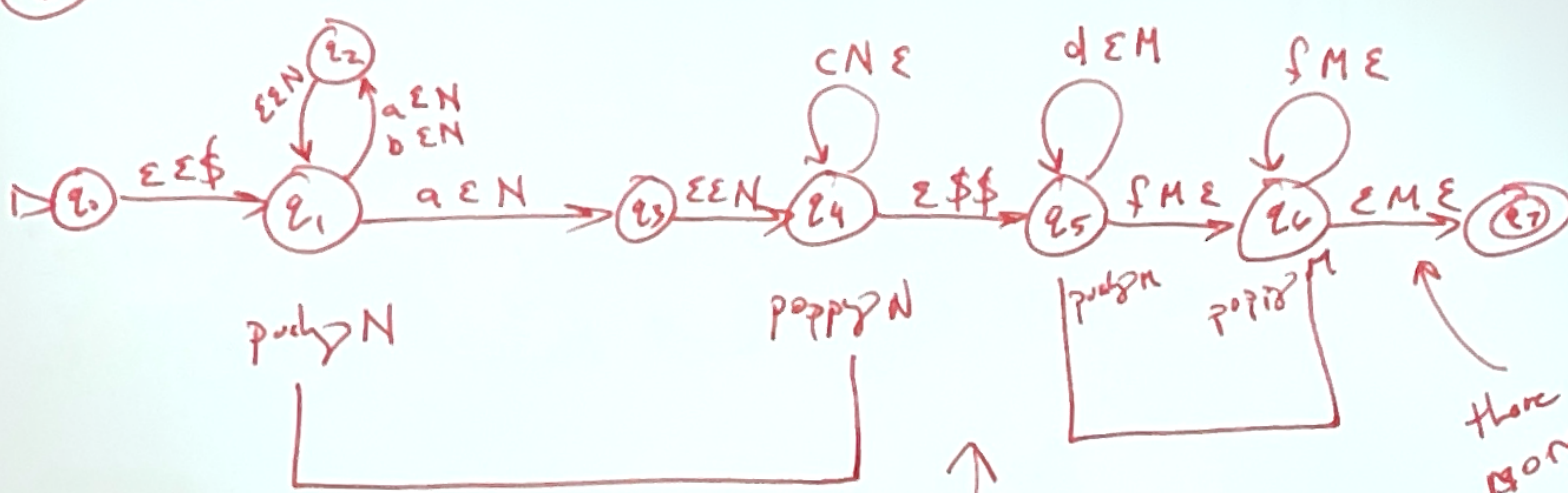
$\rightarrow adfgbchhhkhdb$ // simplify ε

7. Yes. This grammer is ambiguous. Consider the string "dhhhkhd". It has two parse trees that are diffent in the circled regions



Showing one such string ~~ad~~ with two diffent parse trees is sufficient to prove the grammer is ambiguous.

8 Grammar for PDA A



there are more d's than f's

⇓

this means there will be some number of d's equal to the number of f's plus at least one more.

concatenated!

$d^r d^m f^m$

Notice: counting a's and b's. Counting 2 things for each a & b

⇓

$(a+b)^n$

checking for c's popping one N for each c

c^{2n}

But! what about the a ε N and ε ε N between q_1 & q_4?

That means the last character in that section must be an a

⇓

$(a+b)^n a \, c^{2(n+1)}, \; n \geq 0$

• The "f M ε" transition between q_5 & q_6 means $m > 0$.

• The "ε M ε" transition from q_6 to q_7 means $r > 0$.

⇓

$d^r d^m f^m, \; m, r > 0$

FINAL: $(a+b)^n a \, c^{2(n+1)} d^r d^m f^m \quad n \geq 0, \; m, r > 0$

I got distracted and wrote ~~it~~ out how to get the set notation for the language of PDA A instead of giving the grammar. However, it is still useful. Here's how to get the CFG

CFG

Since the overarching relationship is concatenated, I can make the first rule $S \rightarrow AB$ and let A generate the language associated with states q_0 to q_4, and B generate strings in the language of $q_5 \rightarrow q_7$.

For A, the outside relationship is where the a's & b's are being "counted" and c's are being "checked".

$$A \rightarrow aAcc \mid bAcc$$

So, each time either an "a" or "b" is added ~~to~~ in ~~the~~ front of A, "cc" will be added behind it.

The exit condition is the one "a" on the transition from q_1 to q_3:

$$A \rightarrow acc$$

On to B...

The same thought process as detailed on the first page of this problem's solution holds. We can think of the strings accepted by $q_5 \to q_7$ as $d^r d^m f^m$ with $r, m > 0$. I will treat this as a concatenated relationship, so

$$B \to DF$$

D will generate one or more d's

$$D \to dD \mid d$$

F will generate the same # of d's and f's, at least one of each, and all d's before all f's:

$$F \to dFf \mid df$$

All together:

$$S \to AB$$
$$A \to aAcc \mid bAcc \mid acc$$
$$B \to DF$$
$$D \to dD \mid d$$
$$F \to dFf \mid df$$

9 CFG for Language B

$$B = \{ w \text{ over } \{a, b, c\} \mid a^n b c^{n+2}, n \geq 0 \}$$

This can be rewritten as

$$a^n b c c c^n$$

relationship here means every time an "a" is added, a "c" will also be added

$$S \rightarrow aSc$$

The constraint $n \geq 0$ means ~~this~~ there can be zero a/c pairs added.

"bcc" must be added in the middle of any a's & c's.

$$S \rightarrow bcc$$

All together:

$$\boxed{S \rightarrow aSc \mid bcc}$$

(10) Give set notation for grammar C

$$S \to aSb \mid dAd$$
$$A \to fgA \mid BD$$
$$B \to Bbc \mid \varepsilon$$
$$D \to hDh \mid hhD \mid k$$

This is long, but just take it one relationship at a time.

$S \to aSb$ means the outside relationship will have the same number of a's & b's

so far: a^n ? b^n

$S \to dAd$ means there can be zero a's/b's so $n \geq 0$. It also means there will be a "d" inside the a's/b's

so far: $a^n d$? $d b^n$ $n \geq 0$

Now we're into the A piece

$A \to fgA$ means there can be

~~p~~ "fg"'s repeated

so far: $a^n d (fg)^m$ db^n $n \geq 0$

The "exit condition" is $A \to BD$, so $m \geq 0$

$B \to Bbc$ means $(bc)^r$, and the $B \to \varepsilon$

means $r \geq 0$

so far: $a^n d (fg)^m (bc)^r$ db^n $r, n, m \geq 0$

Now we have the rules related to D

$D \to hDh \mid hhD \mid k$

these only involve h's which gives

us some leeway in describing the language.

$hDh \Rightarrow h^s \ldots h^s$

$hhD \Rightarrow (hh)^t$

However, since they can be applied in any order, we could derive substrings like:

h D h	/ h D h
h h D h h	/ h D h
h h h D h h h	// h D h
h h h h h D h h h	// h h D
h h h h h h D h h h h	// h D h

However, since both rules are only adding h's, we can "reorder" which h's were associated with which rule application.

The h's after the D can only come from application of the h D h rule.

All applications of the h h D rule lead to more h's in front of the D.

In other words, we can write this part as

$$(hh)^t h^s \_\_ h^s$$

Finally, D is replaced by k. $\Rightarrow (hh)^t h^s k h^s$

All together: $\boxed{a^n d (fg)^m (bc)^r (hh)^t h^s k h^s \quad r, n, m, s, t \geq 0}$