1. [To be honest, I still don't really understand this one. This is my best effort, albeit not matching what was shown on Lec. 20, Slide 25]

Since L_1 is a CFL, there must be some PDA that generates L_1. Since L_2 is an RL, there must be some NFA N that generates L_2. We can design a PDA P such that it nondeterministically guesses where y begins (with a $\epsilon, \epsilon \rightarrow \epsilon$ transition from $q_{\text{pre-}y}$ to q_{y1} where $q_{\text{pre-}y}$ is the last state before the PDA attempts to build y and q_{y1} is the effective "start state" of the y generating portion) and then generates the rest of the string.

We can edit P such that it changes the $\epsilon, \epsilon \rightarrow \epsilon$ transition to lead from $q_{\text{pre-}y}$ to a set of states that effectively replicate N, but lead to a state that uses up the symbols left in the stack before finally accepting. Since N also generates y, the PDA up to $q_{\text{pre-}y}$ must be capable of generating x, meaning there is a PDA that can generate x and thus that x is part of a CFL.

Alternatively, since L_1 is a CFL, there must be some CFG that generates L_1. Since we know that L_1 is composed of words in the xy format where y is part of L_2 is an RL (and thus a CFL that has its own CFG Y), L_1's CFG can easily be written in such a way that the starting rule is $S \rightarrow XY$ where X generates the remaining portion of the string. This would then mean that portion has a valid CFG and is thus a CFL; since Y generates y alone, this means that X must generate x and thus x is a member of some CFL.

2. No.

Let us assume for any CFL L, any way we can make $L = L_1 \cup L_2$, L_1 and L_2 are also CFLs, as this is what the question implies (that by knowing L and L_1 to be CFLs, L_2 must also be a CFL).

Let us use the language $L = \Sigma^*$, a regular language and therefore a CFL. For any CFL L_1, a possible way to make L via a union with it and another set L_2 is with its complement, since $\Sigma^* = L_1 \cup \overline{L_1}$. This would mean that $L_2 = \overline{L_1}$ should be both valid and a CFG. However, we know that the complement of a CFL is not always a CFL (from slides, copy of the proof below), so we cannot assume that L_2 is a CFL.

Therefore, L and L_1 being CFLs and knowing $L = L_1 \cup L_2$ still does not imply L_2 is a CFL.

Proof that complements of a CFL are not always a CFL from the slides:

CLOSURE PROPERTIES OF CFLs

Complement Let A be a CFL. Is $\overline{A}$ always a CFL?

No! Proof: Assume f.s.o.c. that for **any** CFL L, $\overline{L}$ is also a CFL.

Let A, B be CFLs. By the initial assumption, $\overline{A}, \overline{B}$ are CFLs.

Since CFLs are closed under union $\overline{A} \cup \overline{B}$ must be a CFL.

Finally due to the initial assumption $\overline{\overline{A} \cup \overline{B}}$ must be a CFL.

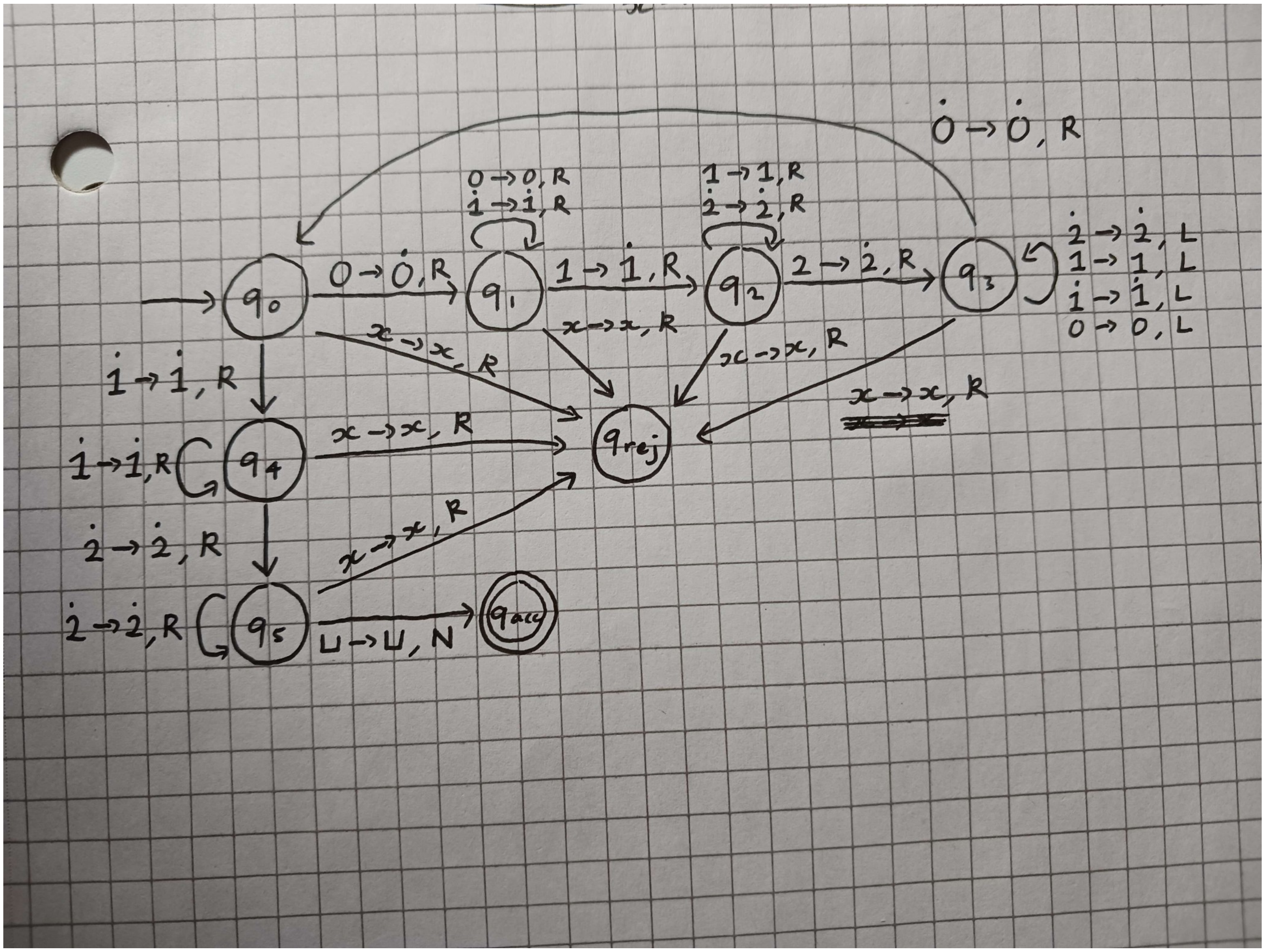
Since $A \cap B = (\overline{\overline{A} \cup \overline{B}})$, it must be that $A \cap B$ is a CFL.

We know this is not always True.

$\Rightarrow$ Contradiction!

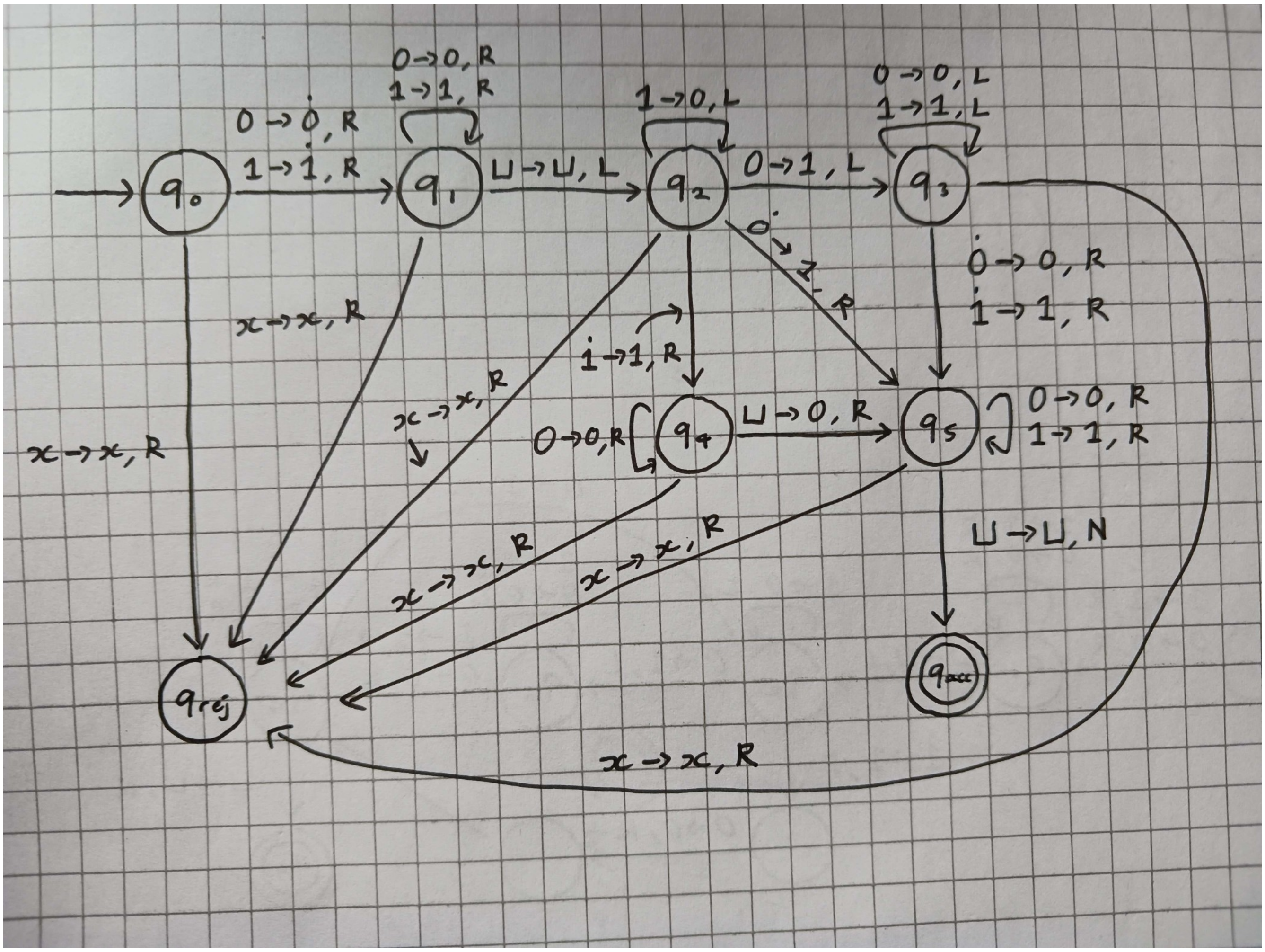
3. [NOTE: $x \rightarrow x$, R means move right on any symbol that isn't one with a transition listed already]

[NOTE: $\sqcup \rightarrow \sqcup$, N means we don't need to move again]



4. [NOTE: $x \rightarrow x$, R means move right on any symbol that isn't one with a transition listed already]

[NOTE: $\sqcup \rightarrow \sqcup$, N means we don't need to move again]



5. Given that the Turing Machine will always start on the leftmost symbol on the tape:

Step 1: If the symbol is blank, accept. Otherwise, place a mark on top of the current symbol.

Step 2: Scan the tape until a blank symbol is found.

Step 3: Move one space left. If the symbol there is not the unmarked version of the marked symbol, reject (such that if we marked 1 and see a marked 1, or see a 0 of any kind, we reject). Otherwise, set the symbol to the blank symbol.

Step 4: Scan left until a marked symbol is found. Set it to a blank symbol.

Step 5: Move right one space and return to Step 1.