Variants of Turing Machines

- what if a TM has more tapes? several heads?

This section: we’ll give detailed descriptions of our machines but not give detailed $\delta$-functions.
- have to redefine $\delta$-function:

$$\delta: Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L,S,R\}^k$$

Thm 3.13: Every multitape TM has an equiv. single-tape TM.

Pf: read in the book
Nondeterministic Turing Machines

- have to redefine $\delta$-function:

$$\delta: Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L,R\}} \quad \text{(or } \mathcal{P}(Q \times \Gamma \times \{L,R\})\text{)}$$

Thm 3.16: Every nondeterministic TM has an equivalent deterministic TM.

Pf:

- The det TM will simulate the computation tree in a BFS manner.

1) Create start config. on the tape

2) Simulate one step of the NTM, put the new config. at the end of the tape

3) Erase the first config., move onto the next and go to 2)
An alternative name for Turing-recognizable languages is **recursively enumerable** languages.

An enumerator is a TM-like “printer” with no input and an extra output tape that prints all strings in a given language.
Enumerators

**Thm 3.21:** A language is Turing-recognizable iff there is an enumerator for it.

\[ \text{Pf: } \Rightarrow \text{ have a TM } T \text{, going to construct an enumerator for } L(T) \]

**Idea 1:** start with string \( E \), if accepted, output continue with \( a, b, aa, ab, ba, bb, aaa \), etc. all strings in lexicographical order

\[ \Rightarrow \text{ PROBLEM: infinite string } \]

**Idea 2:** do the computations on all the strings in parallel (stop a computation if the corresponding string is accepted)

\[ \Rightarrow \text{ PROBLEM: infinite string } \]

**Idea 3:** start w. \( E \) and do 1 step of the TM computation, then add \( a \) and do 1 step on \( E \)

in general, when doing the \( i \)-th step on \( E \), add the \((i+1)\)-st lexicographical string, and 1 step on all \( i+1 \) strings

\[ \Leftrightarrow \text{ have enumerator } E \text{, want to construct TM } T \text{ for } L(E) \]

\[ \Rightarrow \text{ we'll construct a 3-tape TM} \]

- on first tape - input
- on the second tape - will have the output of the enumerator
- on the third tape - working tape for the enumerator

the TM behaves like the enumerator, except whenever it prints a new string, it compares it with the input - if same, accept, otherwise continue

\[ \square \]