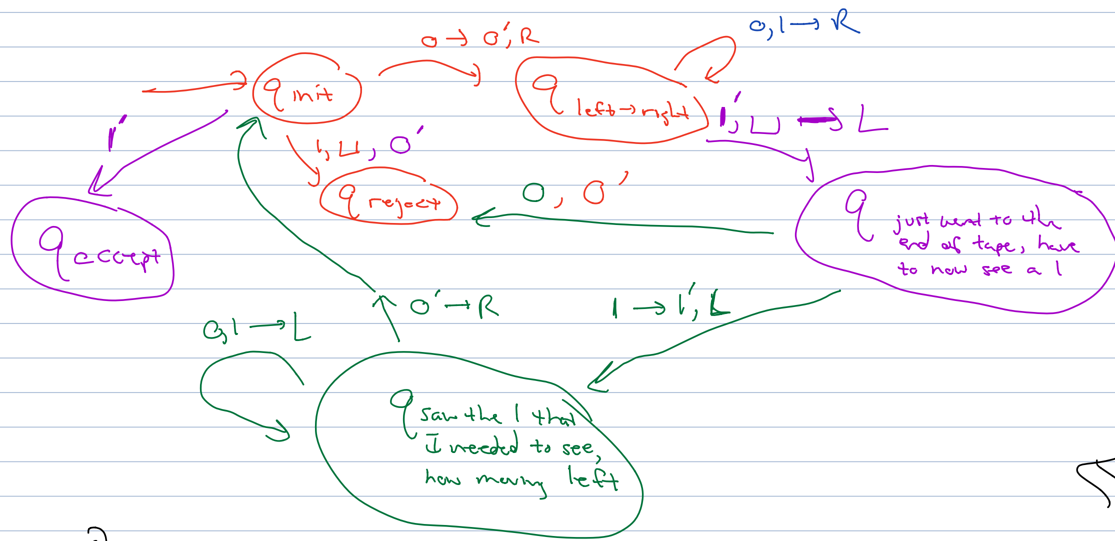
- No class on Wednesday, Oct 9



Turing Machine to decide $L = \{0^n 1^n \mid n \in \mathbb{N}\}$

$= \{01, 0011, 0^3 1^3, \ldots\}$

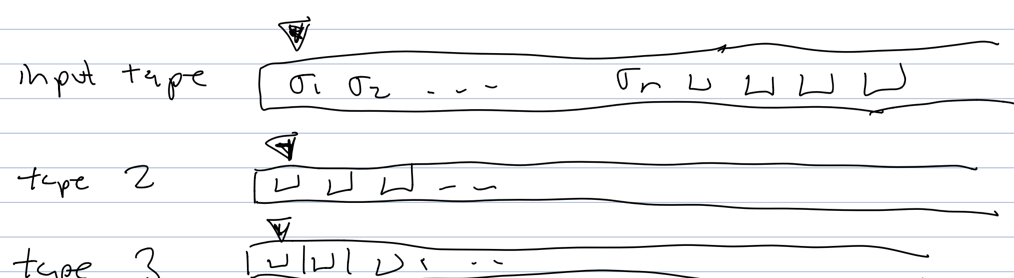
[Sip] "High level description" ☺

"Medium level"

☺ Implementation: Γ tape alphabet, Q,

Very simple language

Simplifying variant: Multi-tape Turing machine



Old 1-tape: $\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$

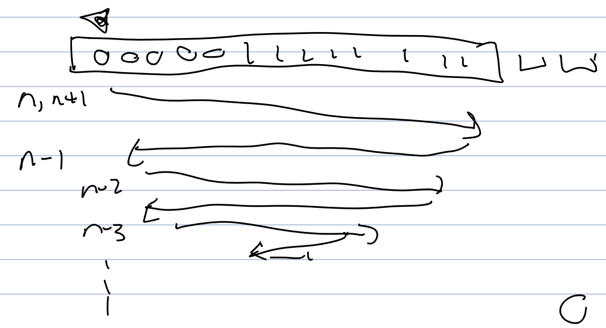
Multi-tape $\quad \delta : Q \times \Gamma^k \longrightarrow Q \times \Gamma^k \times \{L, R, S\}^k$

k tape:

$\uparrow$ stay

Infinite Tapes: Cheat $\to$ powerful

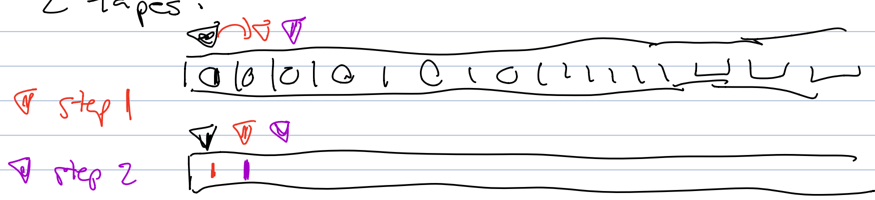
$L = \{0^n 1^m\}$ — "time" string length n, time $\approx n^2$

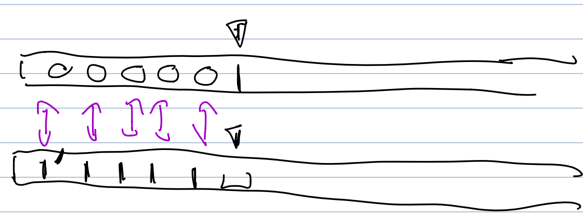


time could be as big as

$\underbrace{0\ 0 \ldots 0}_{n/2}\ \underbrace{1\ 1\ 1\ 1 \ldots 1}_{n/2}$ even

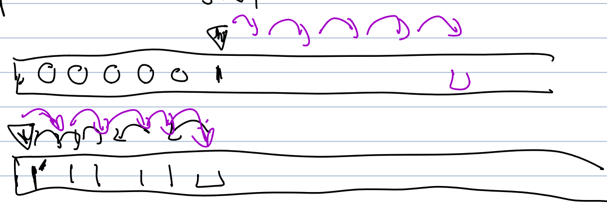
2 tapes:

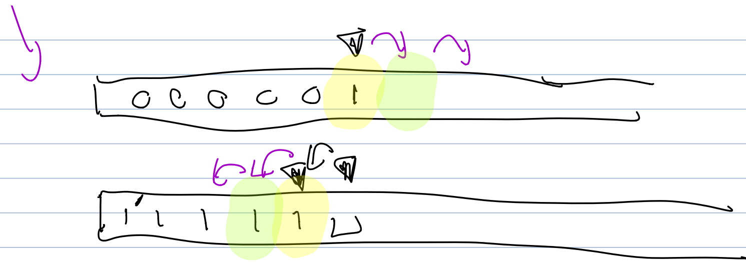




1' start of tape

stay





2 tape machine decide $L = \{0^m 1^m \mid m \in \mathbb{N}\}$

in linear time   input size n

$0\,0\,0 \dots 0\,1$ n steps

$1'\,1\,1\,1\,1\,1\,1$

=

3 or more tapes, then can simulate n steps in time $O(n \log n)$ on 2-tape machine

=

PALINDROME

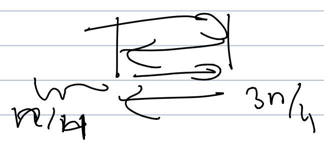
$s = \sigma_1 \dots \sigma_n$, $\quad s^{rev} = \sigma_n \sigma_{n-1} \dots \sigma_2 \sigma_1$, $\quad \sigma_i \in \Sigma$

To decide PALINDROME on 1-tape T.M takes $\geq cn^2$ time

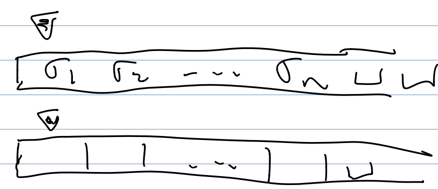
input $\sigma_1 \dots \sigma_n$, is $\sigma_1 = \sigma_n$?

$\sigma_2 = \sigma_{n-1}$?

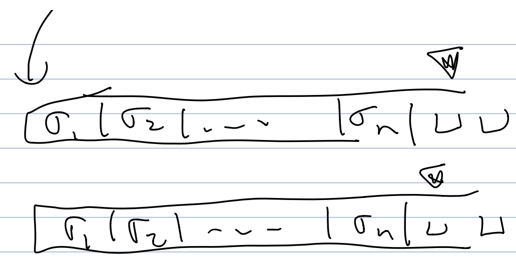
;



On 2 tapes:



n steps



Phase 1

i.e.

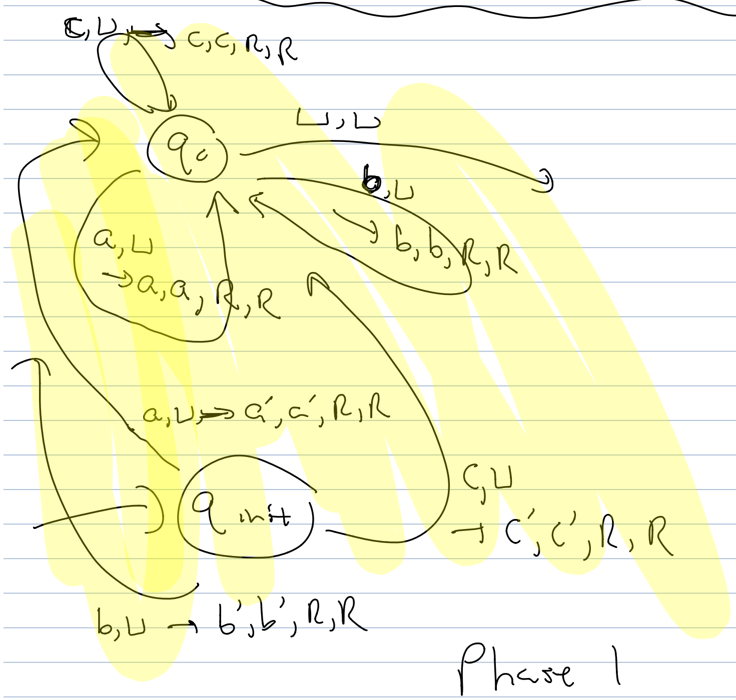
end of tape?

end of Phase 1

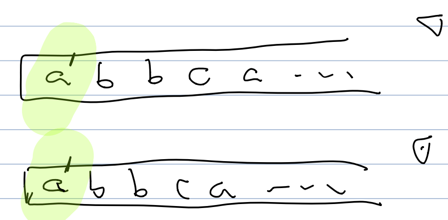
$\Sigma = \{a, b, c\}$

$\delta : Q \times \Gamma^2 \to Q \times \Gamma^2 \times \{L, R, S\}^2$

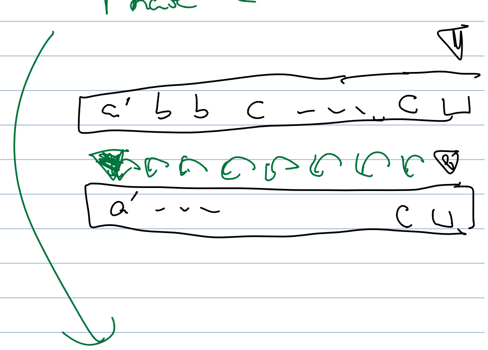
→ q_0 work, copying symbol for symbol until we see a blank



Phase 1



Phase 2



… → … S, L

stay left

(S — is never really needed on a 1-tape machine …)

Phase 3: compare

Time to decide PALINDROME on 2-tape machine

in linear (n)

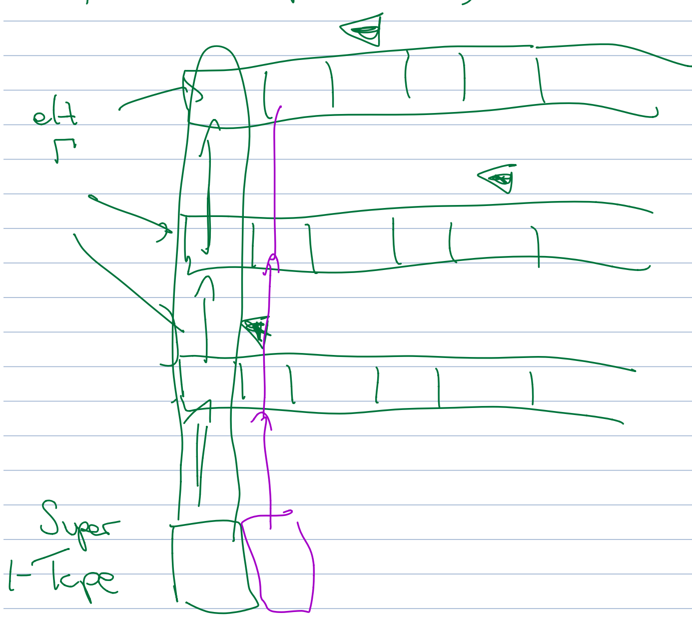
n = size input

Goal: A TM (1-tape, multi-tape) a "reasonable" notion of an algorithm

---

Goal 2: Any "polynomial time" algorithm can be implemented on a Turing machine and also take poly time.

Goal 3: Any k-tape T.m. algorithm can be implemented by a 1-tape Turing machine algorithm



$\underbrace{\Gamma \times \Gamma \times \Gamma}_{\Gamma^k} \times \underbrace{\left\{ \begin{array}{l} \text{t.h. present,} \\ \text{t.h. not present} \end{array} \right\} \times \cdots \times \{ \quad \}}_{\{0,1\}^k}$

$\Gamma^k \times \{0,1\}^k$