Familiarize ourselves with multi-tape Turing machines...

+ Convenient
- Can be simulated by one-tape machine
- As powerful as {C, Java, Python} program
- Universal TM

=

Decide: $\{n_1 \# n_2 \mid n_1^2 = n_2\}$

What does this mean?

Say $\Sigma = \{0,1,2,\ldots,9,\#\}$

SQUARES

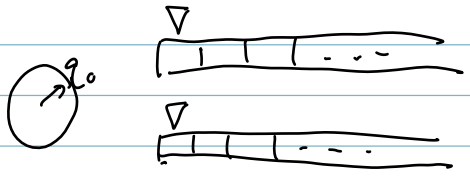
Ch 3 (also done 7.1 - time)
(moving into Ch. 4)

=

2-tape Turing machine

$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej}, \sqcup)$

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



PALINDROME:

(1) 2-tape machine, decide PAL in $O(n)$

(2) 1-tape (claimed) PAL requires time n^2/c, $c = c_M$

Could be convenient to have 5 tapes:

TM: $M = (Q, \Sigma, \ldots)$

$\delta: Q \times \Gamma^5 \to Q \times \Gamma^5 \times \{L,R,S\}^5$

=

Claim: Any k-tape TM deciding a language (either just deciding, or running in polynomial (n) time) has an equivalent 1-tape Turing machine doing the same.

SQUARES = $\{1\#1, 2\#4, 3\#9, 0\#0, 25\#625, \ldots\}$

Not in SQUARES:

$3\#10$, $\#\#$, $025\#00625$ (subject to debate)

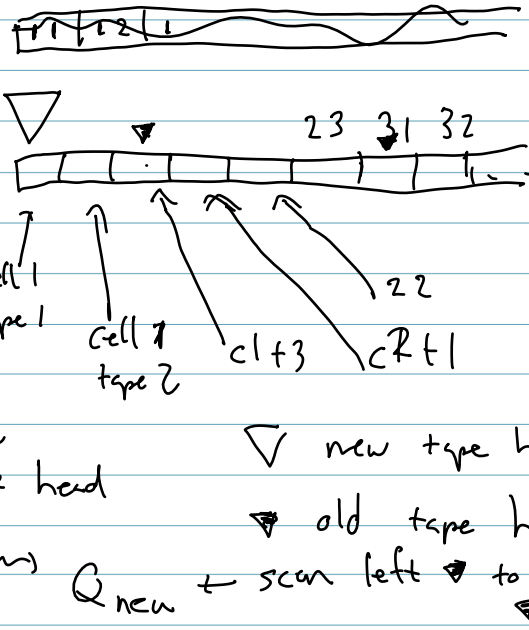
=

$\underbrace{25\#625}_{\text{on tape 1}}$?

25 ← tape 2
25 ← tape 3
125 ← tape 4
500 ←
625 ← tape 5

How long does it take to simulate one step of 3-tape machine with 1-tape?

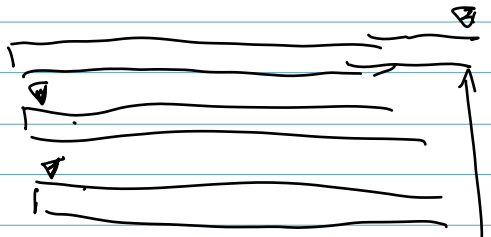
=



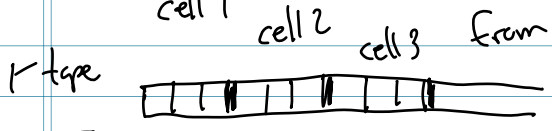
$Q_{old} \rightsquigarrow Q_{new}$ ← scan left to rightmost

⇒ Order (n^4) steps

=



distance $5n^2$ cells from cell #1

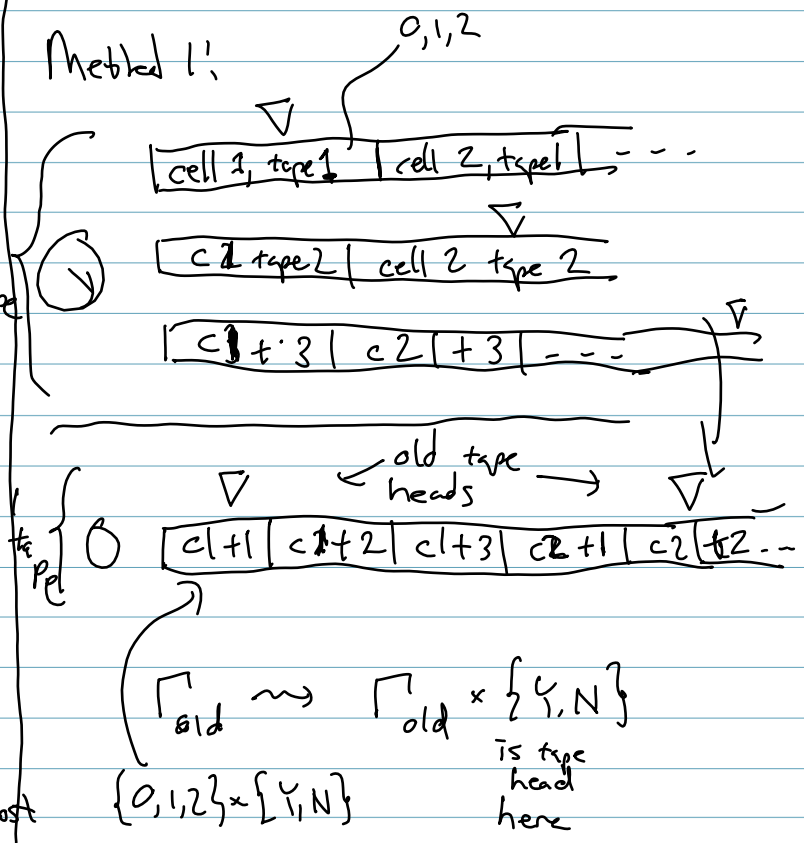


=

- This given $O(n^k)$ on 3-tape $\rightsquigarrow$ $O(n^{2k})$ one tape

Why?

Method 1:

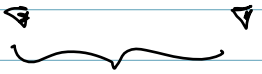


Idea: If tape heads are a distance D apart, and ▽ is at leftmost ▽, we should be able to update in order(D) steps

=

Polynomial time:

In 3-tape machine runs in time $\leq 5n^2$, then



Order(n^2) steps

Order(n^2) steps on 1-tape per 3-tape step