- Multi-tape TM's and

1. \( \{ (i, j, k) \in \mathbb{N}^3 \mid k = i + j \} \)
2. \( \{ (i, j, k) \in \mathbb{N}^3 \mid k = i - j \} \)
3. Universal TM

Recall, a universal TM recognizes

\[ \text{ACCEPTANCE}_{TM} \]

\[ \text{def} = \{ \langle M, i \rangle \mid M \text{ is a T.M. that accepts the input } i \} \]
Why not give some explicit 2-tape TM algorithms?

They get technical:

\[(a, q) \rightarrow (b, w), (R, 5)\]

Instead...
Say: given 

\[(i, j, k)\] say base 10

37126 # 48512 # 20159831

Want to know

① is \(i + j = k\) ?
② is \(i \cdot j = k\)
(2) $i \cdot j = k$

\[
\begin{array}{c}
37126 \\
\times \quad 90002 \\
\hline
74252
\end{array}
\]

\[\text{tape 2} \rightarrow \text{tape 3} \rightarrow \text{tape 4} \rightarrow \text{keep a running total}\]
$i \ast j = k$

\[
\begin{array}{c}
37126 \\
+ 48512 \\
85638
\end{array}
\]

- tape 2
- tape 3
- tape 4
\[ i \cdot j = k \]

\[
\begin{array}{c}
37126 \\
\times \\
48512 \\
\hline
74252 \\
37126 \\
185630
\end{array}
\]

\( \text{running total} \leftarrow \text{tapes} \)

\( \text{sum} \)
Multi-tape $TM$:

Makes it easier (more convincing) to describe algorithms, closer to actual implementations.

Rules:

- finite number of tapes (independent of the input, say $k$ tapes, $k = 1, 2, 3, \ldots$)

Then $k$-tape machine

$$(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$$

Only difference
Thm: Any \( k \)-tape TM, \( M \), has an equivalent 1-tape TM, \( M' \).
Equivalent means

1. literally, for all inputs \( i \),

\[
M \text{ accepts } i \iff M' \text{ accepts } i
\]

\[
M \text{ rejects } i \iff M' \text{ rejects } i
\]

\[
M \text{ loops on } i \iff M' \text{ loops on } i
\]

\[
\text{ never halts}
\]

Q: What is the difference in speed?

Ans: PALINDROME requires \( n^2 \) time

( for an input of length \( n \) ) on a 1-tape TM, but \( O(n) \) time on a 2-tape.
Claim: Any algorithm on a k-tape machine, that runs in time
\leq f(n), where \( f(n) = n^\alpha \)
some \( \alpha > 1 \), can be run in time
\( O \left( (f(n))^2 \right) = O \left( n^{2\alpha} \right) \) on
a 1-tape machine.

Proof idea:
Given 2-tape algorithm, how to simulate on 1-tape machine
Each cell has some element of $\Sigma$.

So let's combine the tapes:

<table>
<thead>
<tr>
<th>cell 1</th>
<th>cell 2</th>
<th>tape 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>cell 1</td>
<td>cell 2</td>
<td>tape 2</td>
</tr>
</tbody>
</table>

by intertwining

<table>
<thead>
<tr>
<th>$c_1+1$</th>
<th>$c_1+2$</th>
<th>$\delta_1+1$</th>
<th>$\delta_1+2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>type</td>
<td>head</td>
<td>head</td>
</tr>
<tr>
<td>over</td>
<td>over</td>
<td>cell 1</td>
<td>cell 1</td>
</tr>
<tr>
<td>(tape 1)</td>
<td>(tape 2)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1-tape equivalent

Rem: After time $T$ of a 2-tape algorithm, the 1-tape algorithm could have

or cell $T$
This accounts for possibly time on 1-tape machine, for \( T \) steps on 2-tape machine, to have total time

\[
\text{order } \left( 1 + 2 + 3 + \ldots + T \right)
\]

\[
= \text{order } \left( \frac{(T+1)}{2} \right)
\]

\[
= \text{order } \left( T^2 \right)
\]

(giving from time \( T \) to order \( T^2 \))

is essentially optimal, see CPSC 58
Alternate way to go from 2-tapes to 1-tape algorithm:

\[ S 
\]

1-tape

write all type 1 info

\[ T \]

tape 2 info

You may need \( T \) tape cells at time \( T \)
Alternate, variant of method 1

1-tape equivalent

stuff about cell 1

tapes 1, 2 of

2-tape algorithm

could go from $\Gamma$ to

to $\big|_{\text{big}} = \Gamma x \Gamma x \{ \text{is tape head} \}^2$ here?
Also; for input

\[ \Gamma \]

add

\[ \Sigma \text{ to } \Gamma_{\text{big}} \]

so that

\[ \Sigma \subset \Gamma_{\text{big}} \]

Fri-day:

Universal TM's...