4.3.4 b) 2-stack PDAs = TMs

figure out how to encode TM tape in 2 stacks
(can use bottom of stack markers)

4.4.4) how does mply mess up poly-time simulation of RATM
– pay attention!

4.6.2a) ww 2 approaches: 1) generate ww^R
reverse w^R

2) for ex for aabaaaba
generate aaaabbaa
rearrange

b) a^{2^n} think of as repeated doublings

c) idea: take advantage of $1 = 1 \quad 4 = 1+3 \quad 9 = 1+3+5 \quad n^2 = \sum_{i=0}^{n-1}(2n+1)$

produce $\cdot a$, $\cdot a \cdot aaa$, $\cdot a \cdot aaa \cdot aaaaa$

$S \to e$	special cases	• marks groups of a's
$S \to a$		$ marks end of string
$S \to \cdot aC\$$	initialization	C makes A for each a (copier)
$aC \to CaA$	make A for a	D moves to A's after they've been rearranged
$Aa \to aA$	move A's to right of a's	T changes A's to a's
$aA \to a \cdot A$	put marker between groups	E erases •'s
$\cdot C \to \cdot D$	change mode	
$Da \to aD$	find marker...	
$D \cdot \to \cdot T$	...found marker, start transforming A's	
$TA \to aT$	transform A to a	
$T\$ \to aaC\$$	all transformed; make 2 more	
$T\$ \to aaE$	and quit or start again	
$aE \to Ea$	skip over a's	
$\cdot E \to E$	erase •	
$\cdot E \to e$	erase E	

$S \Rightarrow \cdot aC\$$
$\Rightarrow \cdot CaA\$$
$\Rightarrow \cdot DaA\$$
$\Rightarrow \cdot Da \cdot A\$$
$\Rightarrow \cdot aD \cdot A\$$
$\Rightarrow \cdot a \cdot TA\$$
$\Rightarrow \cdot a \cdot aT\$$
$\Rightarrow \cdot a \cdot aaC\$$
$\Rightarrow \cdot a \cdot aaCaA\$$

$\Rightarrow^2 \cdot a \cdot CaAaAaA\$$
$\Rightarrow^4 \cdot a \cdot Caaa \cdot AAA\$$
$\Rightarrow^4 \cdot a \cdot Daaa \cdot AAA\$$
$\Rightarrow \cdot a \cdot aaaD \cdot AAA\$$
$\Rightarrow \cdot a \cdot aaa \cdot TAAA\$$
$\Rightarrow^3 \cdot a \cdot aaa \cdot aaaT\$$
$\Rightarrow \cdot a \cdot aaa \cdot aaaaaE$
$\Rightarrow^4 a\,aaa\,aaaaa$

if do @ wrong time, can't erase all •'s

if do @ wrong time, can't transform all A's

Random Access TM

- tape for each reg
- tape for input/output
- tape with $(a, T[a])$ pairs for tape squares w/ non-zero contents
- maybe a temp tape

If RATM computes f in poly time, there is a standard TM that computes f in poly time.

a)

read | $R_0 = T[R_1]$

R_0 | 14

R_1 | 3

tape | $(1,100); (2,10); \cancel{(4,3)}; (3,1); (4,100)$

simulating one step scans tape; how long to scan tape depends on length of tape (m)

$O(m)$

how fast can tape grow?

```
for i = 1 to ∞
    T[i] = 100000
```

step i adds $\log_{10} i + 10$ chars to tape

($\log_{10} i$: decimal rep of i; $+10$: ;(,100000))

```
pow = 2
for i = 1 to ∞
    T[i] = 2^i pow
    pow = pow + pow
```

step i adds $O(i)$ chars to tape

at each step, tape grows at most by O(size of largest value in reg or on tape)

largest value on tape or in reg can increase at most by a constant at each step

∴ size of largest value is $O(t)$

∴ tape increases by $O(t)$ at each step

∴ size of tape is $O(t^2)$

∴ Each of t steps takes $O(t^2)$ to simulate

∴ Simulating t steps takes $O(t^3)$ time.

Nondeterministic TMs

transition relation might have $((q, a), (p, b))$ and $((q, a), (s, \rightarrow))$

NTM M accepts a string w if at least one computation path halts on w (semidecides L iff M accepts $w \Leftrightarrow w \in L$)

NTM M decides a language (or computes f) iff

a) no computation path takes longer N steps (N depends on input)

b) every computation path ends in y if $w \in L$
or n if $w \notin L$

(every path ends w/ $f(w)$ on tape)