CSCC63 TUT 0002

Tutorial 2

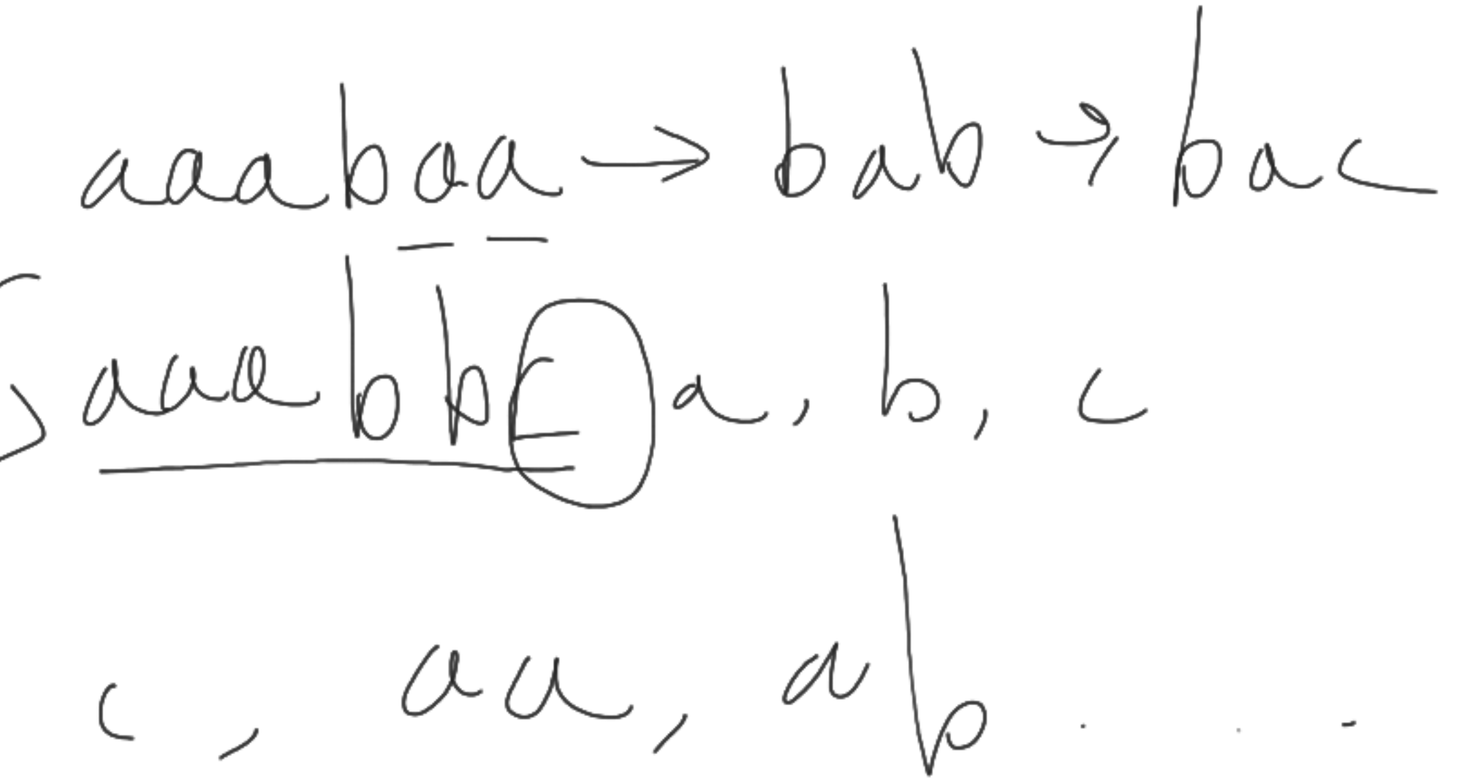
how many iterations to get the string "aaabbc"

aaaaa_: 3^0 + 3 ^1 + 3^2 + 3^3 + 3^4 + 3^5

aaaba_: 3^2

aaabb_: 3

aaabbc: 3



M = (Q, Sigma, Gamma, d, s, qA, qR)

L = { <M, w> | M doesnt move its head beyond the end of the input }

Decidable: Recognizable and Co-Recognizable

Recognizable: Halts on yes instances and accepts

Co-Recognizable: Halts on no instances and rejects

L is decidable if A TM M accepts its yes instances, and rejects its no instances

L = { <M, w> | M doesnt move its head beyond the end of the input }

$$(|\Gamma|^{|w|}) * |Q| * |w|$$

M_l on input <M, w>:
run M on w for |Gamma|^w * |Q| *|w|
reject if any configurations goes beyond the end of the input
otherwise accept

We say A reduces to B of there exists a function P such that for all x, x belongs to A iff P(x) belongs to B

$Halt \leq_p L$

If A reduces to B:

- and A is undecidable, B is undecidable

HALT = { <M, w> | M halts on w }

L = { <M, w> | M tries to move its head off the left side of the input }

P on input <M, w>:

defined M' on input <x>:

1. []
2. run M on w
3. [move left on x if x = w]

return <M', w' = w>

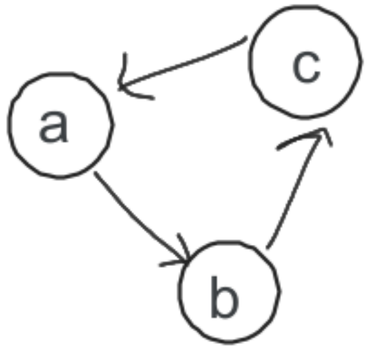
show that if <M, w> in HALT <M', w'> in L

If <M, w> in HALT, M halts on w, M' will move left on w, therefore <M', w'> in L

show that if <M', w'> not in L, then <M, w> not in HALT

If <M', w'> not in L, line 3 was never ran, line 2 must have looped. If M loops on w, then <M, w> is not in halt.

Graph: G = (V, E)



For every node v, build the tiles

#v (s = v)	#
#	v#

the idea is to enforce that there is atleast one tile in between the augmented start and end tiles.

#a(s = a)	b(s = a)	b(s = a)	a	#
#	a(s = a)	c(s = a)	c(s = a)	a#