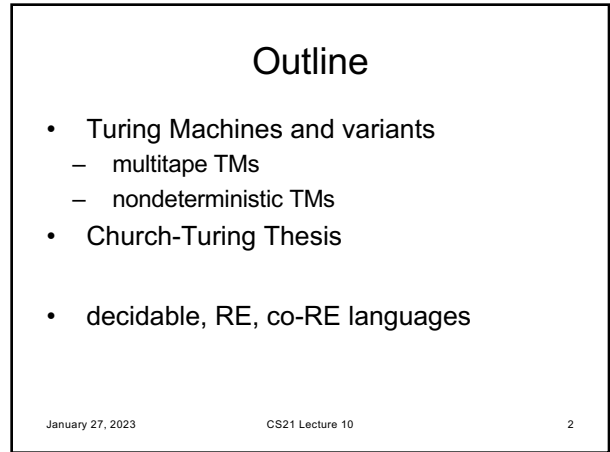
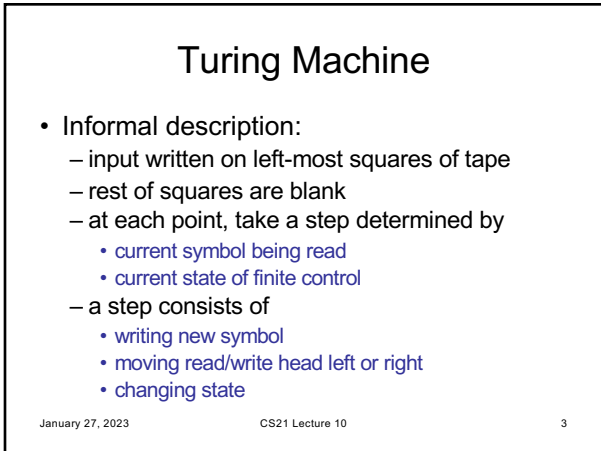


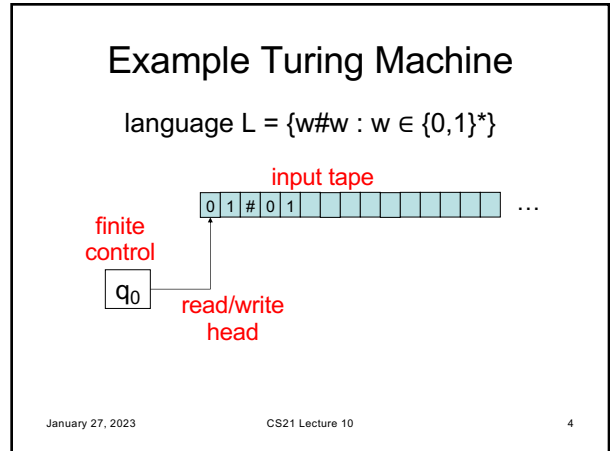
1



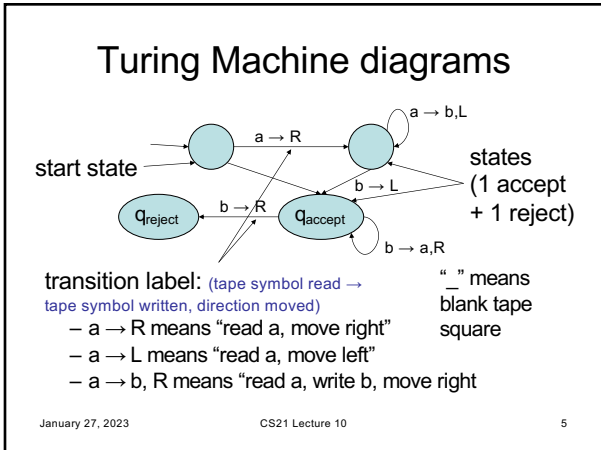
2



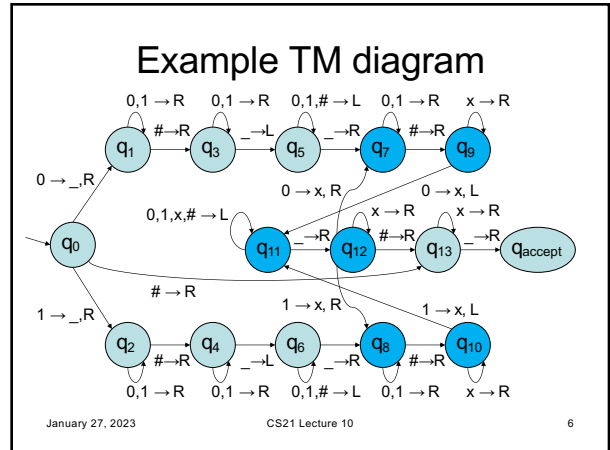
3



4



5



6

TM formal definition

- A TM is a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ where:
 - Q is a finite set called the **states**
 - Σ is a finite set called the **input alphabet**
 - Γ is a finite set called the **tape alphabet**
 - $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ is a function called the **transition function**
 - q_0 is an element of Q called the **start state**
 - $q_{\text{accept}}, q_{\text{reject}}$ are the **accept** and **reject states**

January 27, 2023 CS21 Lecture 10 7

7

Example TM operation

#	0	1			start
#	0	1			start
#	0	1			start
#	0	1			start
#	0	1			↑
#	0	0			↑
#	1	0			accept

program
for "binary
successor"

q	σ	$\delta(q, \sigma)$
start	0	(start, 0, R)
start	1	(start, 1, R)
start	-	(t, -, L)
start	#	(start, #, R)
t	0	(accept, 1, -)
t	1	(t, 0, L)
t	#	(accept, #, R)

January 27, 2023 CS21 Lecture 10 8

8

TM configurations

- At every step in a computation, a **configuration** is determined by:
 - the contents of the tape
 - the state
 - the location of the read/write head
- next step completely determined by current configuration
- shorthand: string uqv with $u, v \in \Gamma^*$, $q \in Q$

meaning:

- tape contents: uv followed by blanks
- in state q
- reading first symbol of v

January 27, 2023 CS21 Lecture 10 9

9

TM configurations

- configuration C_1 **yields** configuration C_2 if TM can legally* move from C_1 to C_2 in 1 step
 - notation: $C_1 \Rightarrow C_2$
 - also: "yields in 1 step" notation: $C_1 \Rightarrow^1 C_2$
 - "yields in k steps" notation: $C_1 \Rightarrow^k C_2$ if there exist configurations D_1, D_2, \dots, D_{k-1} for which $C_1 \Rightarrow D_1 \Rightarrow D_2 \Rightarrow \dots \Rightarrow D_{k-1} \Rightarrow C_2$
 - also: "yields in some # of steps" ($C_1 \Rightarrow^* C_2$)

*Convention: TM halts upon entering $q_{\text{accept}}, q_{\text{reject}}$

January 27, 2023 CS21 Lecture 10 10

10

TM configurations

- Formal definition of "yields":

$u, v \in \Gamma^*$
 $a, b, c \in \Gamma$
 $q_i, q_j \in Q$

$$uaq_i b v \Rightarrow uq_j a c v$$
 if $\delta(q_i, b) = (q_j, c, L)$, and

$$uaq_i b v \Rightarrow uacq_j v$$
 if $\delta(q_i, b) = (q_j, c, R)$

$(q_i \neq q_{\text{accept}}, q_{\text{reject}})$
- two special cases:
 - left end: $q_i b v \Rightarrow q_j c v$ if $\delta(q_i, b) = (q_j, c, L)$
 - right end: uaq_i same as uaq_j

January 27, 2023 CS21 Lecture 10 11

11

TM acceptance

- start configuration: $q_0 w$ (w is input)
- accepting config.: any config. with state q_{accept}
- rejecting config.: any config. with state q_{reject}

TM M accepts input w if there exist configurations C_1, C_2, \dots, C_k

- C_1 is start configuration of M on input w
- $C_i \Rightarrow C_{i+1}$ for $i = 1, 2, 3, \dots, k-1$
- C_k is an accepting configuration

January 27, 2023 CS21 Lecture 10 12

12

Deciding and Recognizing

input

machine

- accept
- reject
- loop forever

- TM M:
 - L(M) is the language it **recognizes**
 - if M rejects every $x \notin L(M)$ it **decides** L
 - set of languages recognized by some TM is called **Turing-recognizable** or **recursively enumerable (RE)**
 - set of languages decided by some TM is called **Turing-decidable** or **decidable** or **recursive**

January 27, 2023
CS21 Lecture 10
13

13

Deciding and Recognizing

input

machine

- accept
- reject
- loop forever

- TM M:
 - L(M) is the language it **recognizes**
 - if M rejects every $x \notin L(M)$ it **decides** L
 - set of languages recognized by some TM is called **Turing-recognizable** or **recursively enumerable (RE)**
 - set of languages decided by some TM is called **Turing-decidable** or **decidable** or **recursive**

January 27, 2023
CS21 Lecture 10
14

14

Classes of languages

- We know: regular \subseteq CFL (proper containment)
- CFL \subseteq decidable
 - proof?
 - decidable \subseteq RE \subseteq all languages
 - proof?

January 27, 2023
CS21 Lecture 10
15

15

Multitape TMs

- A useful variant: k-tape TM

January 27, 2023
CS21 Lecture 10
16

16

Multitape TMs

- Informal description of **k-tape** TM:
 - input written on left-most squares of tape #1
 - rest of squares are blank **on all tapes**
 - at each point, take a step determined by
 - current **k** symbols being read **on k tapes**
 - current state of finite control
 - a step consists of
 - writing **k** new symbols **on k tapes**
 - moving each of **k** read/write heads left or right
 - changing state

January 27, 2023
CS21 Lecture 10
17

17

Multitape TM formal definition

- A TM is a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ where:
 - everything is the same as a TM except the transition function:

$$\delta: Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R\}^k$$
 - $\delta(q_i, a_1, a_2, \dots, a_k) = (q_j, b_1, b_2, \dots, b_k, L, R, \dots, L) =$ “in state q_i , reading a_1, a_2, \dots, a_k on k tapes, move to state q_j , write b_1, b_2, \dots, b_k on k tapes, move L, R on k tapes as specified.”

January 27, 2023
CS21 Lecture 10
18

18

Multitape TMs

Theorem: every k -tape TM has an equivalent single-tape TM.

Proof:

- Idea: simulate k -tape TM on a 1-tape TM.

January 27, 2023

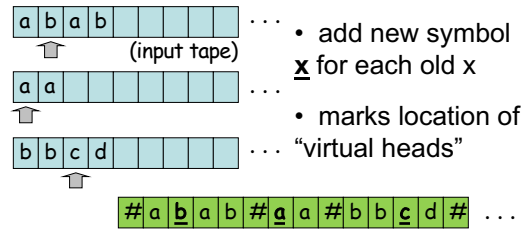
CS21 Lecture 10

19

19

Multitape TMs

simulation of k -tape TM by single-tape TM:



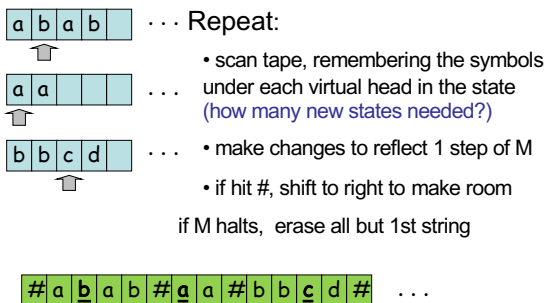
January 27, 2023

CS21 Lecture 10

20

20

Multitape TMs



January 27, 2023

CS21 Lecture 10

21

21

Nondeterministic TMs

- A important variant: **nondeterministic TM**
- informally, several possible next configurations at each step
- formally, a NTM is a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ where:
 - everything is the same as a TM except the transition function:

$$\delta: Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R\})$$

January 27, 2023

CS21 Lecture 10

22

22

NTM acceptance

- start configuration: $q_0 w$ (w is input)
- accepting config.: any config. with state q_{accept}
- rejecting config.: any config. with state q_{reject}

NTM M accepts input w if **there exist** configurations C_1, C_2, \dots, C_k

- C_1 is start configuration of M on input w
- $C_i \Rightarrow C_{i+1}$ for $i = 1, 2, 3, \dots, k-1$
- C_k is an accepting configuration

January 27, 2023

CS21 Lecture 10

23

23

Nondeterministic TMs

Theorem: every NTM has an equivalent (deterministic) TM.

Proof:

- Idea: simulate NTM with a deterministic TM

January 27, 2023

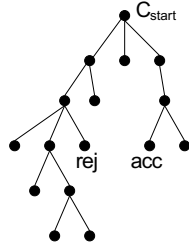
CS21 Lecture 10

24

24

Nondeterministic TMs

Simulating NTM M with a deterministic TM:



- computations of M are a tree
- nodes are configs
- fanout is b = maximum number of choices in transition function
- leaves are accept/reject configs.

January 27, 2023

CS21 Lecture 10

25

25

Nondeterministic TMs

Simulating NTM M with a deterministic TM:

- idea: breadth-first search of tree
- if M accepts: we will encounter accepting leaf and accept
- if M rejects: we will encounter all rejecting leaves, finish traversal of tree, and reject
- if M does not halt on some branch: we will not halt...

January 27, 2023

CS21 Lecture 10

26

26