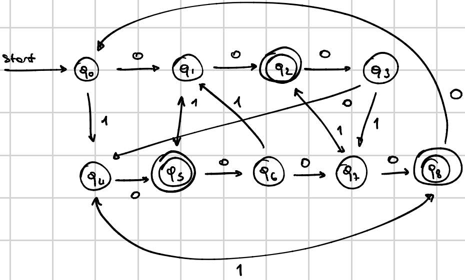
ES)



	0	1	2	3	4	5	6	7
1	X							
2	X	X						
3		X	X					
4	X		X	X				
5	X	X		X	X			
6		X	X		X	X		
7	X		X	X		X	X	
8	X	X		X	X		X	X

= 1st iteration

= 2nd iteration

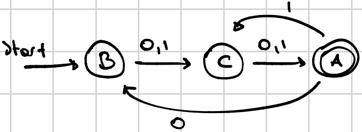
= 3rd iteration

↓

None ⇒ finished

$$Eq = \{(0,3),(0,6),(1,4),(1,7),(2,5),(2,8)\}$$

$$= \{\overset{B}{\{0,3,6\}}, \overset{C}{\{1,4,7\}}, \underset{A}{\{2,5,8\}}\}$$



Nondeterministic TM

In a **nondeterministic** Turing machine, NTM for short, the transition function δ is set-valued :

$$\delta(q, X) = \{(q_1, Y_1, D_1), (q_2, Y_2, D_2), \ldots, (q_k, Y_k, D_k)\}$$

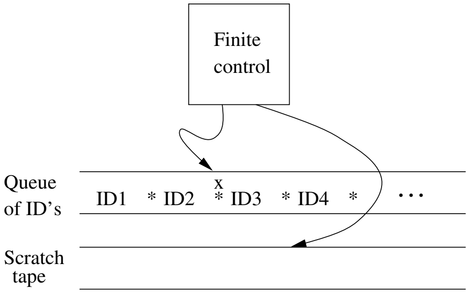
At each step, the NTM chooses one of the triples as the next move

The NTM accepts an input w if **there exists** a sequence of choices that leads from the initial ID for w to an ID with an accepting state

Nondeterministic TM

Theorem For each NTM M_N, there exists a (deterministic) TM M_D such that $L(M_N) = L(M_D)$

Proof (skecth) We specify M_D as a TM with two tapes



Nondeterministic TM

A single ID in the **queue** (first) tape is marked as being processed

M_D performs the following cycle

- copy the marked ID from the queue tape to the **scratch** (second) tape
- for each possible move of M_N, add a new ID at the end of queue tape
- move the marker in the queue tape to the next ID

Nondeterministic TM

Let m be the maximum number of choices for M_N. After n moves, M_N reaches a number of ID bounded by

$$1 + m + m^2 + \cdots + m^n \leqslant nm^n + 1$$

M_D explores all the IDs reached by M_N in n steps before each ID reached in $n + 1$ steps, as in a **breadth first** search

If there exists an accepting ID for M_N on w, M_D reaches this ID in a finite amount of time. Otherwise, M_D does not accept, and may not halt

We therefore conclude that $L(M_N) = L(M_D)$ $\square$

Nondeterministic TM

Observe that the TM M_D in the previous theorem can take an amount of time **exponentially larger** than M_N to accept an input string

$P \stackrel{?}{=} NP$

We **do not know** if this slowdown is necessary: this very important issue will be the subject of investigation in a next chapter

TM with restrictions

We impose some restrictions on the definition of TM / multi-tape TM:

- tape is unlimited only in one direction
- two tapes used in stack mode

We prove that these models are equivalent to TM

Think about the above definitions as normal forms

These models are especially useful in some proofs that we will present later on

TM with semi-infinite tape

In a TM with **semi-infinite tape**

- there are no cells to the left of the **initial tape position**
- a tape symbol can never be overwritten by the blank B

In a TM with semi-infinite tape each ID is a sequence of tape symbols other than B, i.e., there are no "holes"

TM with semi-infinite tape

We can **simulate** a TM by means of a TM with semi-infinite tape with two tracks

- the upper track represents the initial position X_0 and all tape cells to its right
- the lower track represents all tape cells to the left of X_0, in reverse order
- a special symbol $*$ is used to mark the initial position

X_0	X_1	X_2	$\dots$
$*$	X_{-1}	X_{-2}	$\dots$

TM with semi-infinite tape

Theorem Each language accepted by a TM M_2 is also accepted by a TM M_1 with semi-infinite tape

Proof (sketch) First, we modify M_2 in such a way that it uses a **new** tape symbol B' each time B is used to overwrite a tape symbol

Let $M_2 = (Q_2, \Sigma, \Gamma_2, \delta_2, q_2, B, F_2)$ be the modified TM. We define

$$M_1 = (Q_1, \Sigma \times \{B\}, \Gamma_1, \delta_1, q_0, [B, B], F_1)$$

TM with semi-infinite tape

The states of M_1 are $Q_1 = \{q_0, q_1\} \cup (Q_2 \times \{U, L\})$. Symbols U, L indicate whether M_1 is visiting the upper or lower track

The input symbols of M_1 are pairs $[a, B]$ with a an input symbol of M_2

The tape symbols Γ_1 of M_1 are pairs in $\Gamma_2 \times \Gamma_2$ with the addition of pairs $[X, *]$ for each $X \in \Gamma_2$, where $*$ is used to mark the initial position of M_1 tape

The accepting symbols of M_1 are $F_1 = F_2 \times \{U, L\}$

TM with semi-infinite tape

Transitions in δ_1 implement the following moves

- place $*$ on the initial position, in the lower track, and restore the initial conditions of M_2
- when M_1 is not in the initial cell, the moves of M_2 are simulated with
  - the same direction if U appears in the state
  - the reverse direction if L appears in the state
- upon reading $*$
  - if M_2 moves to the right, M_1 simulates the same move
  - if M_2 moves to the left, M_1 simulates the same move but it reverses the direction

TM with semi-infinite tape

It can be shown by induction on the number of steps of a computation that the IDs of M_1 and M_2 match, modulo

- the reversal of the L track of M_1
- its concatenation on the left with the U track of M_1
- the elimination of the $*$ marker

It follows that $L(M_1) = L(M_2)$ $\square$

Multi-Stack machine

We apply to a multi-tape TM the restriction to use each tape in stack mode

- can only overwrite at the top
- can only insert at the top
- can only delete at the top

The resulting model accepts only recursively enumerable language, since it is a restriction of a multi tape TM

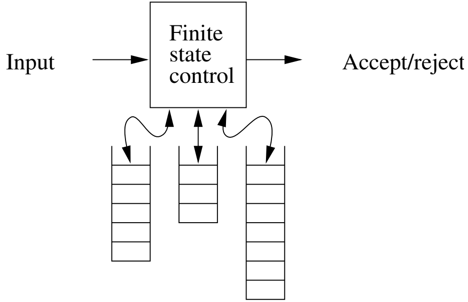
Multi-Stack machine

Let M be a multi-tape TM with tapes used in stack mode. We also assume that

- the input is provided in an **external**, read-only tape and with end-marker \$, and it can only be read from left to right
- M can perform ϵ-moves, but these moves must not be in conflict with each other or with other reading moves (determinism)

Multi-Stack machine

M is called a **multi-stack machine**, and can be viewed as a generalization of the deterministic PDA



Multi-Stack machine

In a multi-stack machine with k stacks, a **transition rule** has the form

$$\delta(q, a, X_1, X_2, \ldots, X_k) = (p, \gamma_1, \gamma_2, \ldots, \gamma_k)$$

In words, when the machine is in state q and reads input symbol $a \in \Sigma \cup \{\epsilon\}$, and with X_i on top of the i-th stack, $1 \leqslant i \leqslant k$, it moves to state p and replaces each X_i with γ_i

Multi-Stack machine

Theorem If a language L is accepted by a TM, then L is accepted by a multi-stack machine with two stacks

Proof (sketch) Let $L = L(M)$ for a TM M. We construct a machine S with two stacks, having special symbols used as **markers** at the bottom of the stack

The basic idea is to

- simulate the tape to the left of the current position with the first stack
- simulate the tape starting from the current position and extending to the right with the second stack

Multi-Stack machine

The transition rules of S implement the following strategy

- copy the input $w\$$ into the first stack
- move the contents of the first stack into the second stack
- if M overwrites X with Y and moves to the right, S pushes Y on the first stack and pops X from the second stack
- if M overwrites X with Y and moves to the left, S pops the symbol Z from the first stack and replaces X with ZY in the second stack
- in addition, S employs some special moves to handle the case where M is located at the end points of the tape (one of the two stacks contains the bottom marker)
- S accepts whenever M accepts □

TM and computer

Theorem If a language L is accepted by a modern computer, then L is accepted by a TM

Proof Omitted