

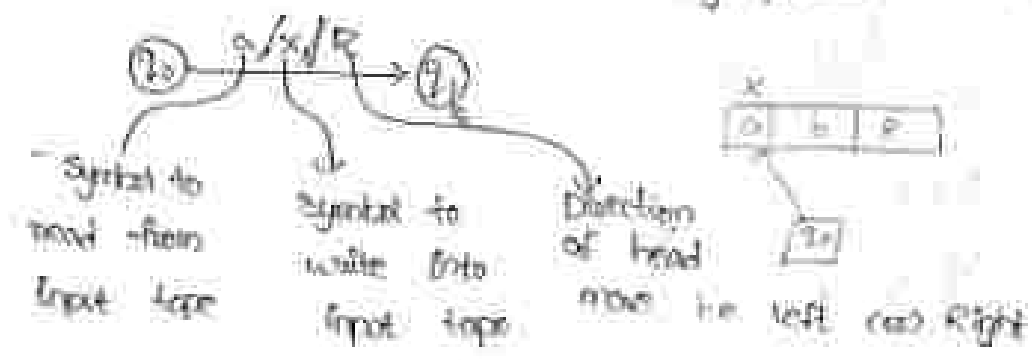
TURING MACHINE

Turing machine contains 7 tuples $(Q, E, \gamma, \delta, q_0, F, B)$

- where Q = Set of states
- E = Set of input symbols
- γ = Set of input tape symbols
- δ = Transition function
- q_0 = Initial state
- F = final state
- B = Blank symbol

where δ can be defined as $\delta: Q \times E \rightarrow Q \times \gamma \times \{L/R\}$

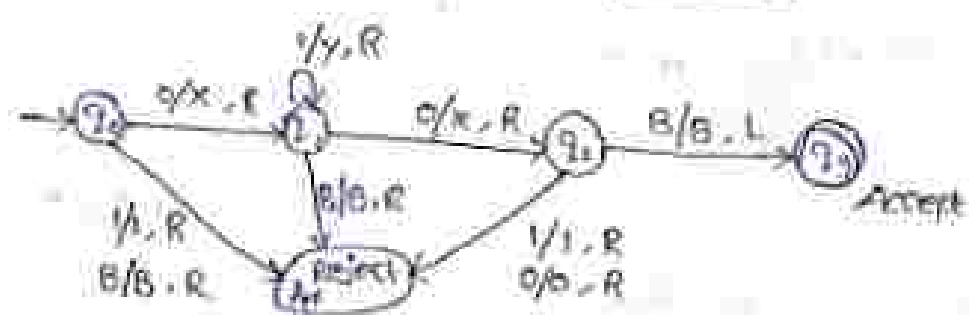
The Transition Diagram:- the transition function can be represented in the form of graphical notation.



Q Design a turing machine for $L = 01^*0$

$L = 01^*0$

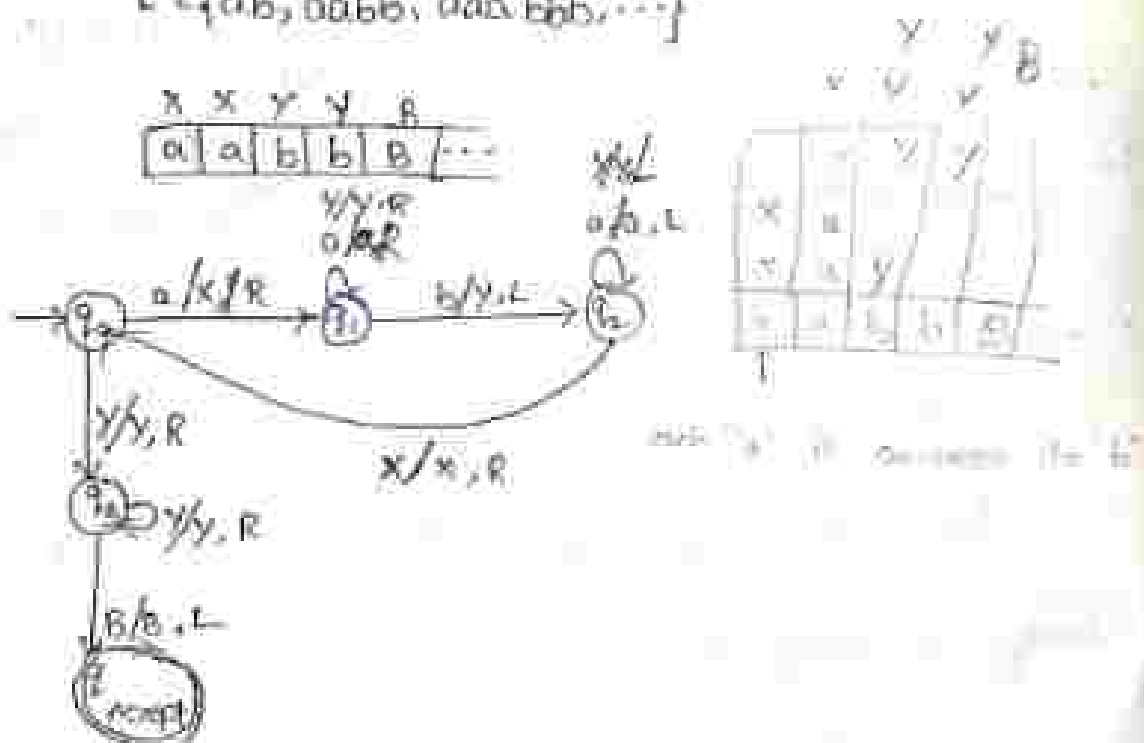
$L = \{00, 010, 0110, 01110, \dots\}$



State \ Input	a	b	x	y	B
q_0	$\langle q_1, a, R \rangle$	$\langle q_4, b, R \rangle$	-	-	$\langle q_2, B, R \rangle$
q_1	$\langle q_1, x, R \rangle$	$\langle q_1, y, R \rangle$	-	-	$\langle q_2, B, R \rangle$
q_2	$\langle q_4, a, R \rangle$	$\langle q_4, b, R \rangle$	-	-	$\langle q_3, B, L \rangle$
q_3	-	-	-	-	-
q_4	-	-	-	-	-

Q. Design a Turing machine for language $L = \{a^n b^n / n \geq 1\}$

$L = \{ab, aabb, aaabbb, \dots\}$



State \ Input	a	b	x	y	B
q_0	$\langle q_1, x, R \rangle$	-	-	$\langle q_0, y, R \rangle$	-
q_1	$\langle q_1, a, R \rangle$	$\langle q_2, y, L \rangle$	-	$\langle q_1, y, R \rangle$	-
q_2	$\langle q_2, a, L \rangle$	-	$\langle q_2, x, R \rangle$	$\langle q_2, y, L \rangle$	-
q_3	-	-	-	$\langle q_3, y, R \rangle$	$\langle q_1, B, L \rangle$
q_{Accept}	-	-	-	-	-

8

design Turing machine for $L = \{a^n b^n c^n / n \geq 1\}$

$L = \{abc, aabbcc, aaabbbccc, \dots\}$

$TM = \{Q, \Sigma, \Gamma, \delta, q_0, B\}$

$x \ x \ y \ y \ z \ z \ B$
 $\boxed{a \ a \ b \ b \ c \ c \ B} \dots$

$x \ x \ y \ y \ z \ z \ B$

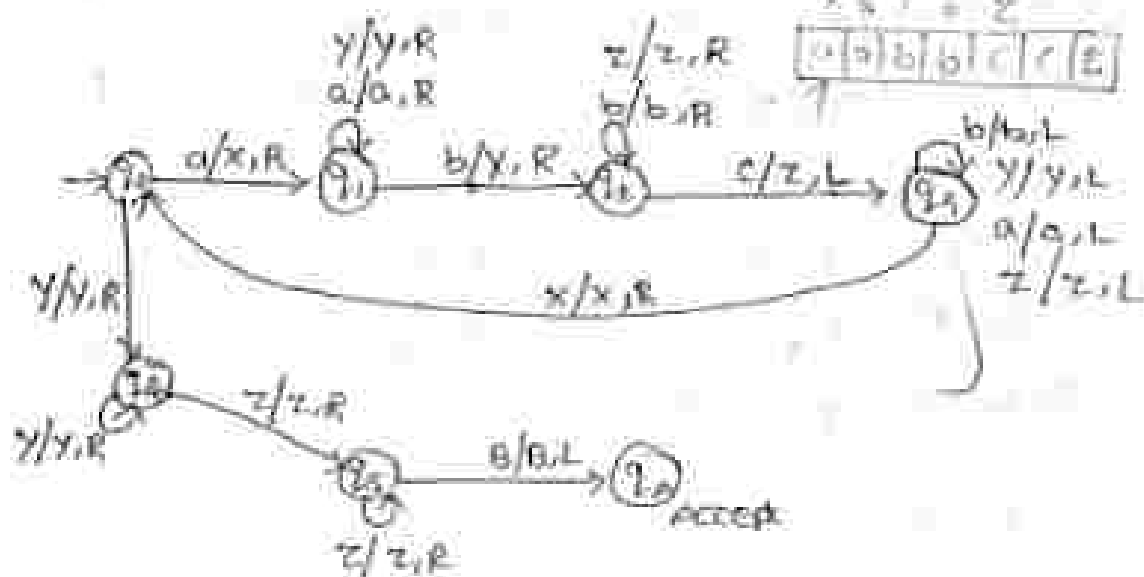
$x \ x \ y \ y \ z \ z \ B$

$x \ x \ y \ y \ z \ z \ B$

$x \ x \ y \ y \ z \ z \ B$

$x \ x \ y \ y \ z \ z \ B$

$x \ x \ y \ y \ z \ z \ B$



state \ tape symbol	a	b	c	x	y	z	B
$\rightarrow q_0$	$\langle q_1, x, R \rangle$	-	-	-	$\langle q_1, y, R \rangle$	-	-
q_1	$\langle q_1, a, R \rangle$	$\langle q_2, y, R \rangle$	-	-	$\langle q_1, y, R \rangle$	-	-
q_2	-	$\langle q_2, b, R \rangle$	$\langle q_3, z, L \rangle$	-	-	$\langle q_2, z, R \rangle$	-
q_3	$\langle q_3, a, L \rangle$	$\langle q_3, b, L \rangle$	-	$\langle q_0, x, R \rangle$	$\langle q_0, y, R \rangle$	$\langle q_0, z, L \rangle$	-
q_4	-	-	-	-	$\langle q_4, y, R \rangle$	$\langle q_5, z, R \rangle$	-
q_5	-	-	-	-	-	$\langle q_5, z, R \rangle$	$\langle q_4, B, L \rangle$
q_A	-	-	-	-	-	-	-

TM: $M = \{Q, \Sigma, \Gamma, \delta, q_0, B, F\}$

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_n\}$

$\Sigma = \{a, b\}$

$\Gamma = \{a, b, c, x, y, z, B\}$

$\delta(q_0, a) = (q_1, x, R)$

$\delta(q_5, b) = (q_3, b, L)$

$\delta(q_1, y) = (q_4, y, R)$

$\delta(q_5, x) = (q_0, x, R)$

$\delta(q_1, a) = (q_1, a, R)$

$\delta(q_3, y) = (q_3, y, L)$

$\delta(q_1, b) = (q_2, y, R)$

$\delta(q_3, z) = (q_5, z, L)$

$\delta(q_1, y) = (q_1, y, R)$

$\delta(q_4, y) = (q_4, y, R)$

$\delta(q_2, b) = (q_2, b, R)$

$\delta(q_4, z) = (q_5, z, R)$

$\delta(q_2, c) = (q_3, z, L)$

$\delta(q_5, z) = (q_5, z, R)$

$\delta(q_2, z) = (q_2, z, R)$

$\delta(q_5, B) = (q_n, B, L)$

$\delta(q_5, a) = (q_3, a, L)$

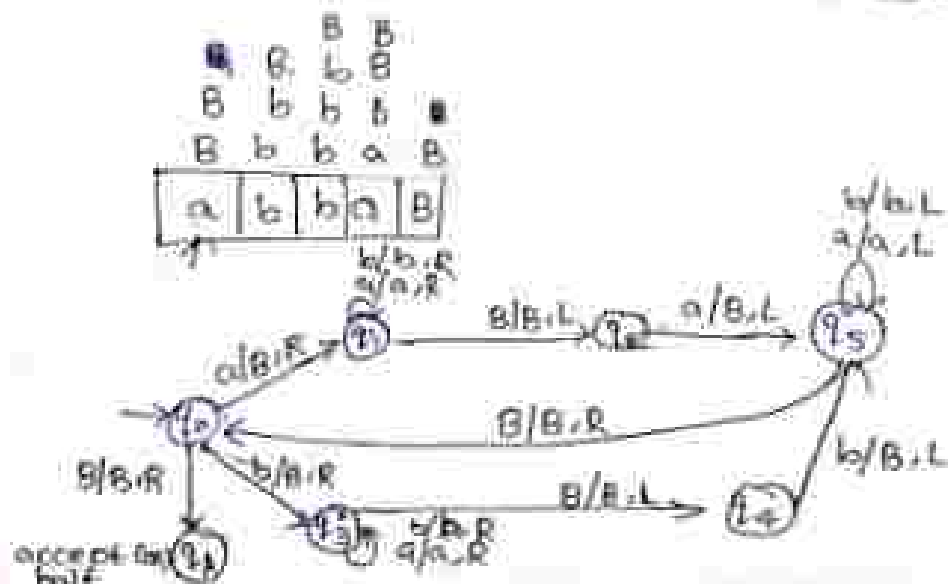
$F = \{q_n\}$

Design Turing machine for $L = \{w w^R \mid w \in (a, b)^*\}$

Let $L = \{w w^R \mid w \in (a, b)^*\}$

It is a even length palindrome

$L = \{aa, bb, abba, baab, abbbba, baabbb, \dots\}$



State	a	b	B
$\rightarrow q_0$	$\langle q_1, B, R \rangle$	$\langle q_3, B, R \rangle$	$\langle q_A, B, R \rangle$
q_1	$\langle q_1, a, R \rangle$	$\langle q_1, b, R \rangle$	$\langle q_2, B, L \rangle$
q_2	$\langle q_2, B, L \rangle$	-	-
q_3	$\langle q_3, a, R \rangle$	$\langle q_3, b, R \rangle$	$\langle q_4, B, L \rangle$
q_4	-	$\langle q_5, B, L \rangle$	-
q_5	$\langle q_5, a, L \rangle$	$\langle q_5, b, L \rangle$	$\langle q_6, B, R \rangle$
q_A	-	-	-

10 abba

q_0 a b b a \emptyset
 $\vdash \emptyset$ q_1 b b \emptyset \emptyset
 $\vdash \emptyset$ b q_1 b a \emptyset
 $\vdash \emptyset$ b b q_1 a \emptyset
 $\vdash \emptyset$ b b a q_1 \emptyset
 $\vdash \emptyset$ b b q_2 a \emptyset
 $\vdash \emptyset$ b q_5 b \emptyset \emptyset
 $\vdash \emptyset$ q_5 b b \emptyset \emptyset
 $\vdash \emptyset$ q_5 \emptyset b b \emptyset \emptyset
 $\vdash \emptyset$ q_6 b b \emptyset \emptyset
 $\vdash \emptyset$ \emptyset q_3 b \emptyset \emptyset
 $\vdash \emptyset$ \emptyset b q_9 \emptyset \emptyset
 $\vdash \emptyset$ \emptyset q_4 b \emptyset \emptyset
 $\vdash \emptyset$ q_5 \emptyset \emptyset \emptyset \emptyset
 $\vdash \emptyset$ \emptyset q_6 \emptyset \emptyset \emptyset
 $\vdash \emptyset$ \emptyset \emptyset q_4 \emptyset \emptyset
accept

7) Design Turing machine for 'parity checker' that outputs '0' if '1' depending on 'w'.

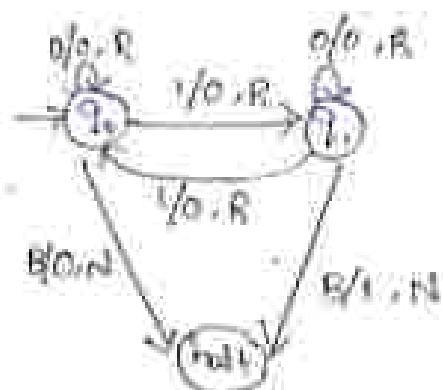
if odd = 1

if even = 0

0 0 0 1
0 1 0 B

transition table

0 0 0 0 0 0 0
0 1 1 0 1 1 B



odd 1's 10

even 1's 1010

+ 9, 0 1 0 B

+ 9, 1 0 1 0 B

+ 0 9, 1 0 B

+ 0 9, 0 1 0 B

+ 0 0 9, 0 B

+ 0 0 9, 1 0 B

+ 0 0 0 9, B

+ 0 0 0 9, 0 B

+ 0 0 0 1 halt

+ 0 0 0 0 9, B

+ 0 0 0 0 0 halt

Design TM for 2's complement

If 10010

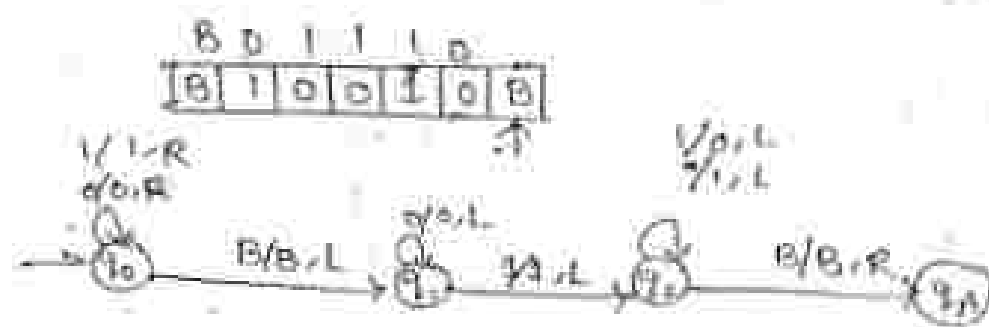
1's 01101

+ 1
2's 01110

MSB 1'sB
0 1 0 0 1 0 B

accept halt

Next of all we have to move LSB then upto



$q_0: 800108$

$q_1: 0100108$

$q_2: 0900108$

$q_3: 0090108$

$q_4: 0009108$

$q_5: 0001908$

$q_6: 0001098$

$q_7: 00019108$

$q_8: 0009108$

$q_9: 0090108$

$q_{10}: 0900108$

$q_{11}: 0901108$

$q_{12}: 0901108$

$q_{13}: 9001108$

$q_{14}: 9001108$

• Addition of two integers. Design TM's
design Turing Machine for to accept set of all the
palindromes.

34



6502

b6
b7C
b7D

666

100

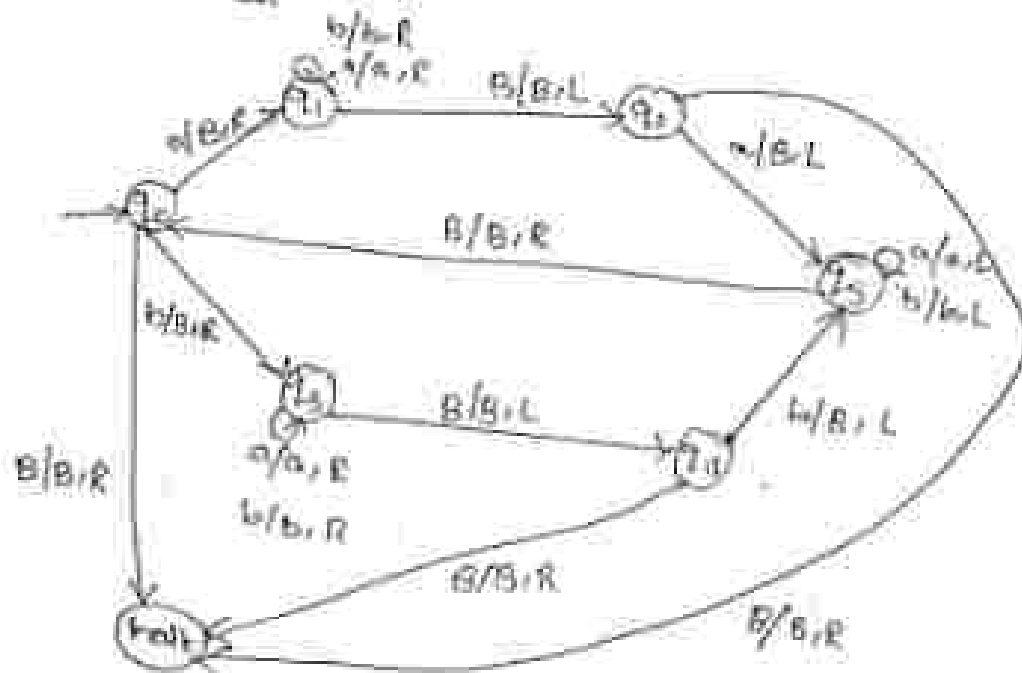


146

B a e e

68 69 70

13/6

**tools.**

4. 9. 1963

F 89.0bB

1. 80, 9, 68

1-62 69B

1 Ba B₂ b B

189088

1953 B a B B

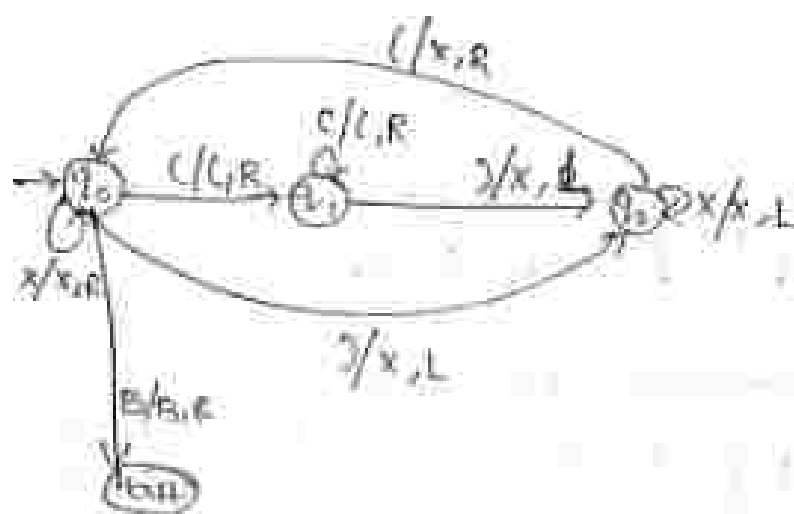
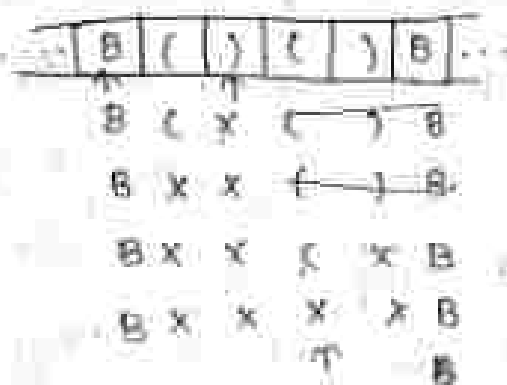
1 8 9 0 6 8

1 8 8 9. 8 8

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040 1

1389

Design Turing Machine for parenthesis checking



1. $q_0((,)B$

2. $q_1(,)B$

3. $q_2(,)B$

4. $q_0(x, x)B$

5. $q_1(x, x)B$

6. $q_2(x, x)B$

7. $q_0(x, x)B$

8. $q_1(x, x)B$

9. $q_2(x, x)B$

10. $q_0(x, x)B$

11. $q_1(x, x)B$

12. $q_2(x, x)B$

13. $q_0(x, x)B$

$x, x, x, B \rightarrow H$

Types of Grammars - Chomsky Hierarchy:

Linguist Noam Chomsky defined a hierarchy of languages in terms of complexity. This four level hierarchy, called the Chomsky hierarchy, corresponds to four classes of machines.

The Chomsky hierarchy classifies grammars according to form of their productions into the following four levels.

- (1) Type 0 grammars - unrestricted grammar
- (2) Type 1 grammars - context sensitive grammar
- (3) Type 2 grammars - context free grammar
- (4) Type 3 grammars - regular grammar

(1) Type - 0 grammars - Unrestricted Grammars (URG)

These grammars include all formal grammars. In URGs, all the productions are of the form $\alpha \rightarrow \beta$, where α and β may have any number of terminals and non-terminals, i.e., no restrictions on either side of production. Every grammar is included in it if it has at least one non-terminal on the left hand side.

Ex: $aA \rightarrow abCB$
 $aA \rightarrow bAA$
 $bA \rightarrow a$
 $S \rightarrow aAb|C$

or $\{a^i b^j c^k\}$

They generate exactly all languages that can be recognized by a Turing machine. The language that is recognized by a Turing machine is defined as set of all the strings on which it halts. These languages

Scanned by CamScanner

are also known as the recursively enumerable languages.

(2) Type 1 grammar - Context sensitive Grammar: (CSG)

These grammars define the context-sensitive languages. In Context-sensitive grammar, all the productions are of the form $\alpha \rightarrow \beta$, where length of α is less than or equal to length of β , i.e. $|\alpha| \leq |\beta|$. α and β may have any number of terminals and non-terminals. These languages are exactly all the languages that can be recognized by linear bound automata.

Ex: $|\alpha| \leq |\beta| \quad \alpha \rightarrow \beta$
 $aAbcD \rightarrow abcDbeD$

(3) Type 2 Grammar - Context-free Grammar (CFG)

These grammars define the context-free languages. These are defined by rules of the form $\alpha \rightarrow \beta$ with $|\alpha| \leq |\beta|$ where $|\alpha| = 1$ and α is non-terminal and β is a string of terminals and non-terminals. We can replace α by β regardless of where it appears. Hence the name context-free grammar.

These languages are exactly those languages that can be recognized by a pushdown automaton. CFG Context-free languages defines the syntax of all programming languages.

Ex: $\begin{aligned} &S \rightarrow aS \mid Sa \mid a \\ &S \rightarrow aAA \mid bBB \mid \epsilon \end{aligned}$

(4) Type 3 grammars - regular grammars:

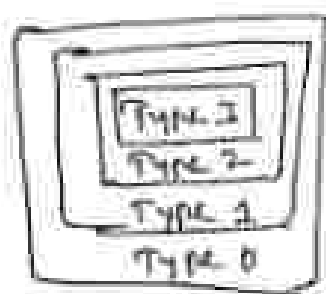
These grammars generate the regular languages. Such a grammar restricts its rules to a single non-terminal on the LHS. The RHS consists of either a single terminal or string of terminals with single non-terminal on left or right end.

Ex: $A \rightarrow SA|a$ — right linear grammar $\alpha \in \{V\}^*$
 $A \rightarrow Aa|a$ — left linear grammar $\beta = V^+T^+$

* Every regular language is context free, every context free language is context sensitive and every context sensitive language is recursively enumerable.

Table: Chomsky's hierarchy

<u>Grammar</u>	<u>Language</u>	<u>Automaton</u>	<u>production rules</u>
Type 0	Recursively enumerable	Turing machine	$\alpha \rightarrow \beta$ no restriction on α, β α should have at least one non-terminal
Type 1	Context sensitive	Linear bounded automata	$\alpha \rightarrow \beta$ $ \alpha \leq \beta $
Type 2	Context free	Push down automata	$\alpha \rightarrow \beta$ $ \alpha = 1$
Type 3	Regular	Finite state automata	$\alpha \rightarrow \beta$ $\alpha \in \{V\}$ $\beta \in \{V\}^+T^+$ $= T^+ \{V\}$ $= T^+$



Chomsky hierarchy of grammars

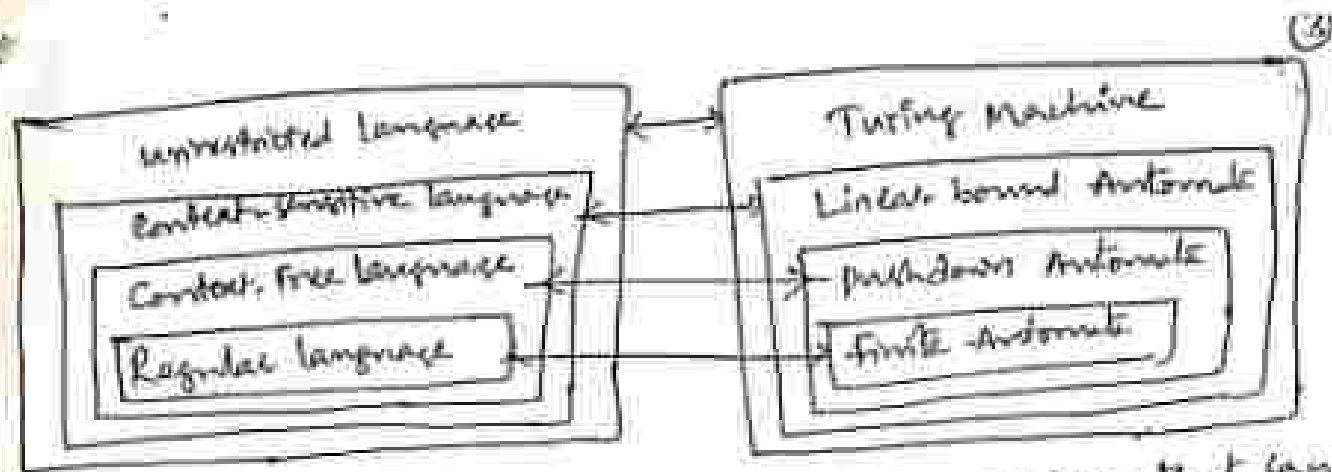


Fig: The hierarchy of languages and the machine that can recognize the same is shown above fig.

- Every RL is context free, every CFL is context-sensitive and every CSL is unrestricted. So the family of regular language can be recognized by any machine.
- CFLs are recognized by pushdown automata, linear bounded automata and Turing machines.
- CSLs are recognized by linear bounded automata and Turing machines.
- Unrestricted languages are recognized by only Turing machines.

(6)

Push Down Automata (PDA)

①

PDA - FA + Stack
non-deterministic

PDA
↓
DPDA NPDA

PDA: $(Q, \Sigma, \Gamma, q_0, Z_0, F, \delta)$

DPDA: $\delta: Q \times \{\Sigma \cup \epsilon\} \times \Gamma \rightarrow Q \times \Gamma$

NPDA: $\delta: Q \times \{\Sigma \cup \epsilon\} \times \Gamma \rightarrow 2^{(Q \times \Gamma)^*}$



Q = finite set of states

Σ = input symbols

δ = Transition function

q_0 = initial state

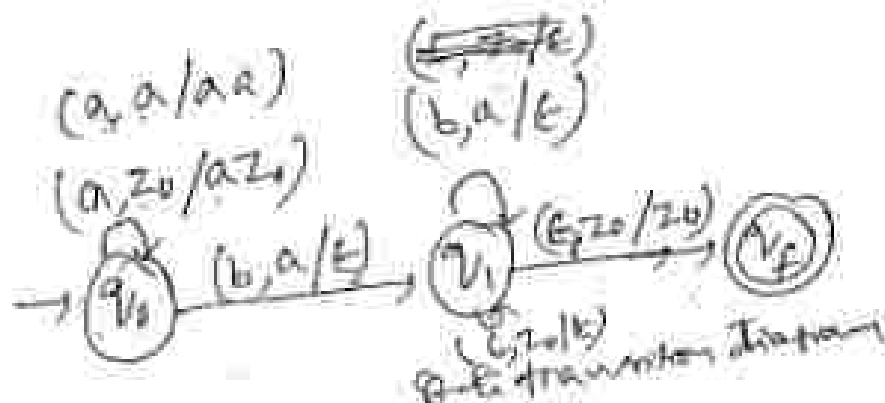
Z_0 = bottom of the stack

F = set of final states

Γ = stack alphabet

Ex: $a^n b^n | n \geq 1$

$a a b b \epsilon$
↑ ↑ ↑ ↑ ↑



transitions
function

PDA → final state
Empty stack

$\delta(q_0, a, z_0) = \delta(q_0, a, a)$

$\delta(q_0, a, a) = (q_0, aa)$

$\delta(q_0, b, a) = (q_1, \epsilon)$

$\delta(q_1, b, a) = (q_1, \epsilon)$

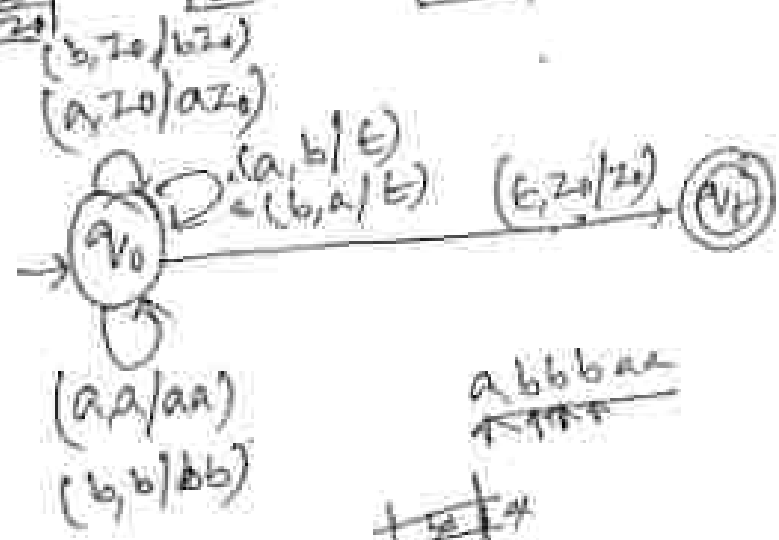
$\delta(q_1, \epsilon, z_0) = (q_f, z_0) \text{ or } (q_f, \epsilon)$

accepting final state or empty stack

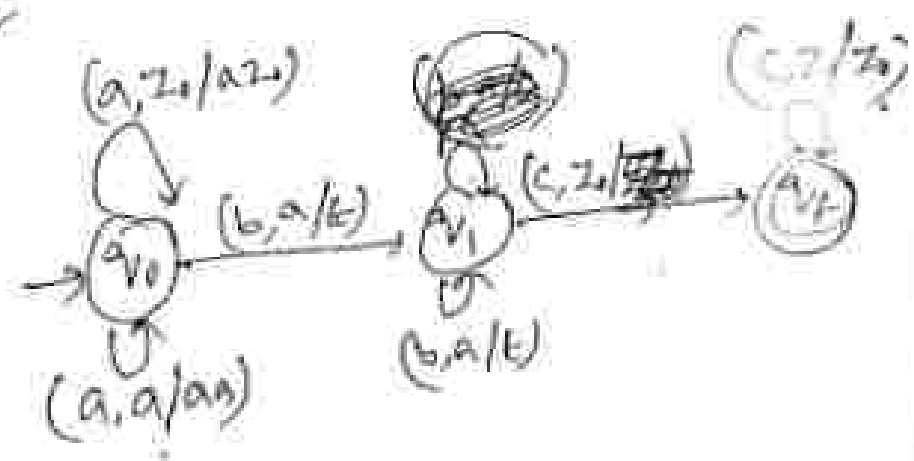
$|w|/n_a(w) = n_b(w)$ [number of a's must be even]
DPDA

Ex: a b bbaa
 aabb baba
 a bab

baba a bab a bba

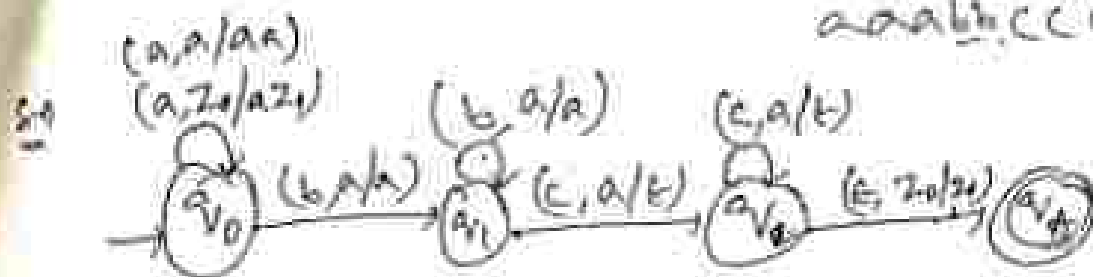


Ex: $a^n b^n c^m$ | $n, m \geq 1$.
 $a^n \rightarrow paa$
 $b^n \rightarrow pbb$
 $c^m \rightarrow m \text{ times}$

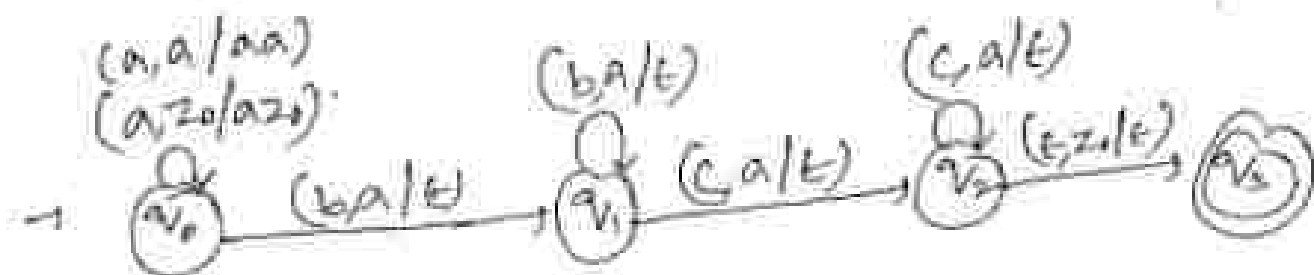


$a^n b^m c^n \mid n, m \geq 1$ ②

distinct last $\frac{1}{2}$

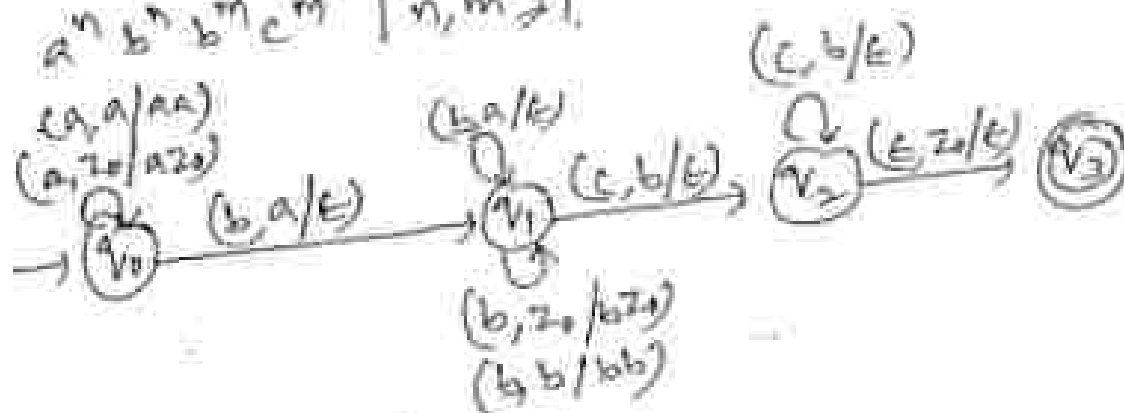


$a^{m+n} b^m c^n \mid m, n \geq 1$

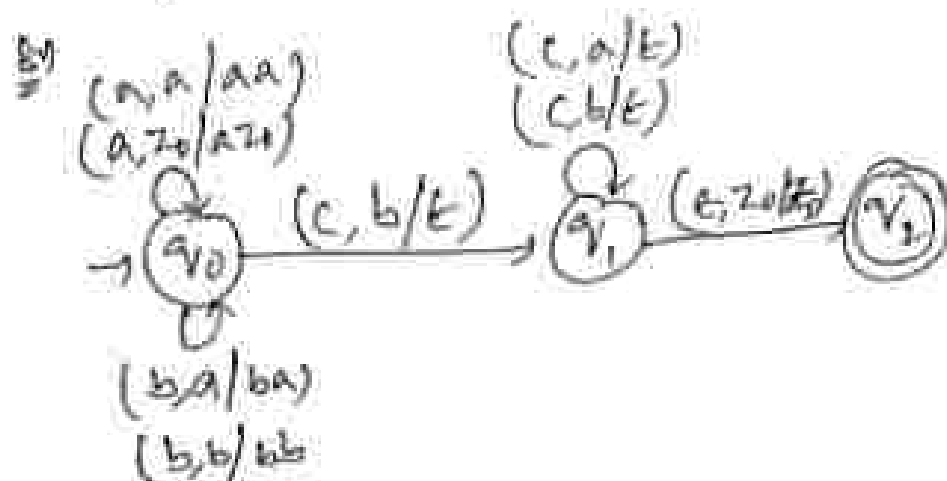


$a^n b^{m+n} c^m \mid n, m \geq 1$

by $a^n b^n b^m c^m \mid n, m \geq 1$



$a^n b^m c^{n+m} \mid n, m \geq 1$



* $a^n b^m c^m a^n \mid n, m \geq 1$

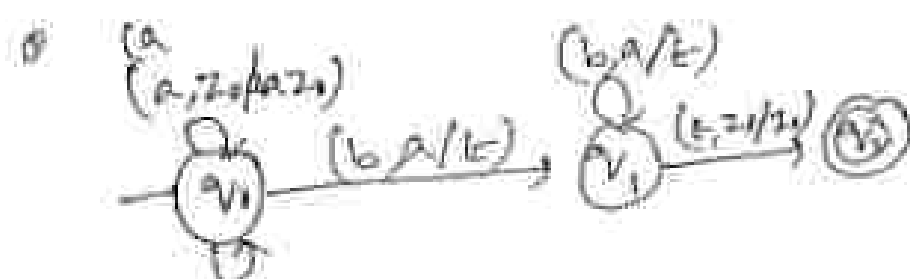
~~1.3~~ $a^n b^m c^m a^n \mid n, m \geq 1$

* $a^n b^m c^m a^n \mid n, m \geq 1$ X Not CF
Not PD

* $a^n 2^n \mid n \geq 1$

$a^n b^{2n} \mid n \geq 1$

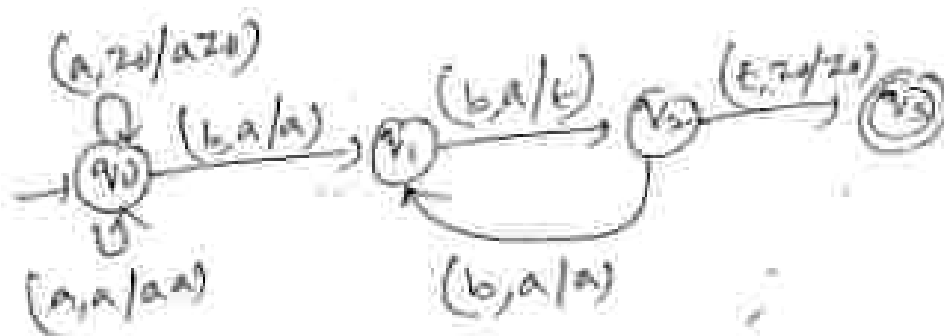
1.3 $abb, aabbbb, aaabbbbbb \dots$
for every a - push two a's.
for every b - pop two b's



$bb, bbb,$

$(a, a/aaa)$

for every b



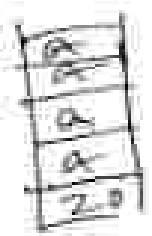
$aaabbbbbb$



Ex: $a^n b^n c^n \mid n \geq 1$ X not PDA

one possibility

X
 $a^n b^n c^n$
 $a^n a^n a^n$



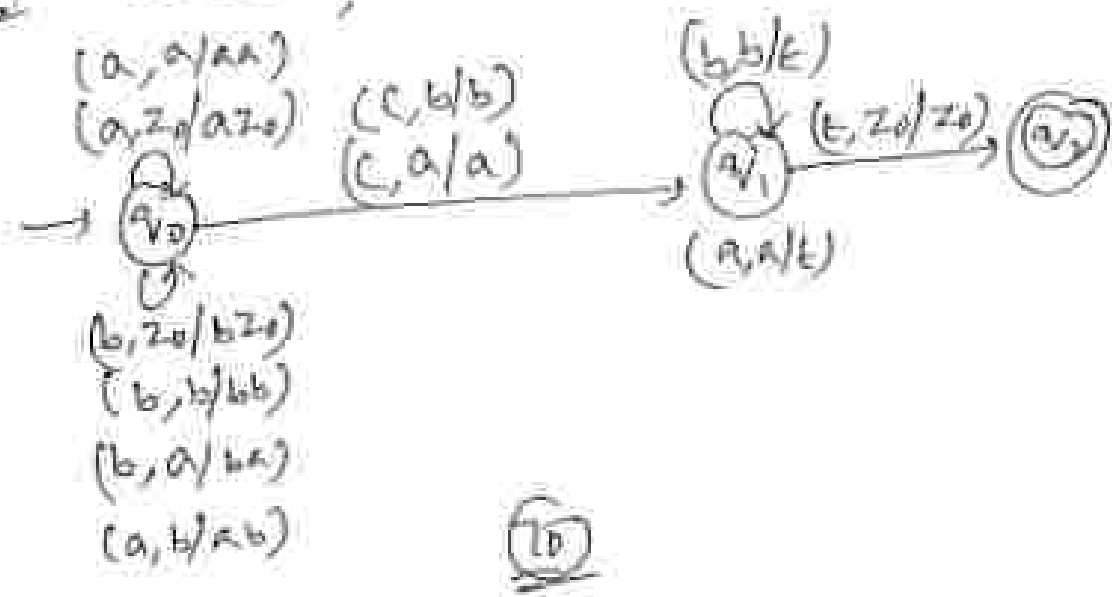
abba

for every a - two a's per

Ex: $ww^R \mid w \in (a,b)^+$



Ex: $ababa, abbcbba$



Ex: $ww^R \mid w \in (a+b)^+$ → whenever

Ex: $\frac{ababab}{w} \frac{bababa}{w^R}$



NDPDA $ww^R \mid w \in (a+b)^+$

 $(b, z_1/b^2)$
$$(a, z_0 / a z_1)$$

Diagram illustrating a transition in a finite automaton:

State q_0 (start state) transitions to state q_1 on input a with output b . The transition is labeled $(b, a / \epsilon)$.

State q_1 transitions to state q_2 (final state) on input a with output ϵ . The transition is labeled $(\epsilon, a / a)$.

$$(a, b/a^2)$$

(2/2/24)

(a, a/aa)

(b) (5) DPP

29. *Myrica*

$$(b, b/k)$$
$$(a, a/E)$$
$$(L_2/Z) \rightarrow (V_2)$$
Abba

Interlocking style

2222
Bertilsson & Son
10. 2222, 201

5/1, 9, 20, 21

$$\{v_0, a_0, a_1, a_2, \dots\}$$

no data! cat

$$(a_1, a_2, a_3, \dots) \quad (r_1, r_2, r_3, \dots)$$


no. of atoms | centre

$$(a_1, a_2, a_3) \quad (a_1, a_2, a_3)$$

$\frac{1}{2} \times 10 \times 10 = 50$

$$(v_0, t, a_0, z_0) \quad (v_1, t, a_1, z_1) \quad (v_2, t, a_2, z_2)$$
$$\frac{(1, 2)}{2!}$$

total

had a

aa baa
w w



$$\frac{100}{100} = \frac{100}{100}$$

NPAA

Center not same

12

Change in getting in center

2/2/2

Language decidability

* Introduction

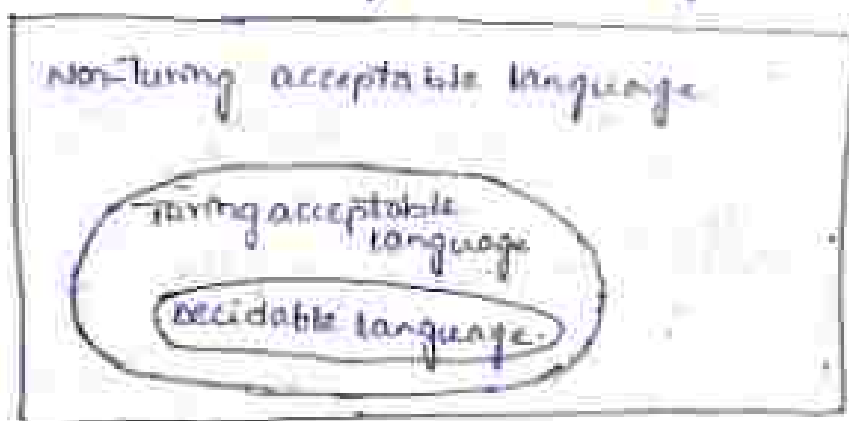
* Examples

Introduction:-

* Decidable problem:-

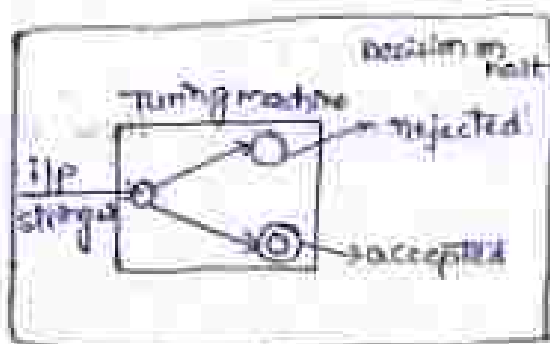
A language is called Decidable (or recursive) if there is a Turing machine which accepts and halts on every IP string 'w'.

* Every decidable language is a Turing acceptable



* A decision problem 'p' is decidable if the language 'L' of all 'YES' instances to 'p' is decidable.

* For a decidable language, for each IP string, the Turing machine halts either at the accept (or) the reject state.



Example:-

1) find out whether the following problem is decidable (or) not:

Is a number 'n' prime?

eg:- Prime numbers = {2, 3, 5, 7, 11, 13, 17, 19, ...}

divide the number 'n' by all the numbers b/w

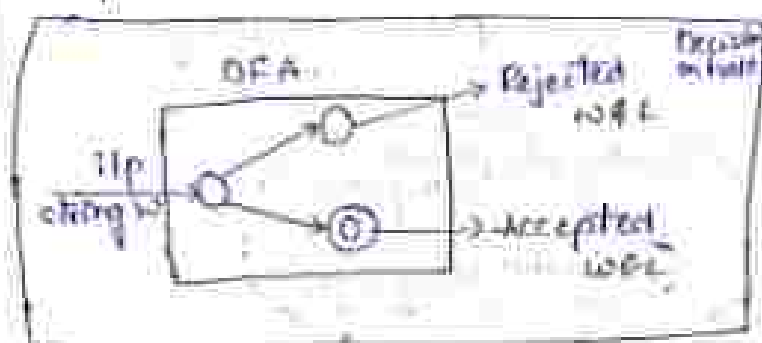
1. and into starting from 2.

If any of these numbers produce a remainder 0 then it goes to the rejected state. otherwise it goes to the accepted state. so, here the answer could be either yes or no.

-hence, it is a decidable problem.

2) Given a Regular language 'L' and String 'w', how can we check if $w \in L$.

sol: Take the DFA that accepts 'L' and check if 'w' is accepted.



Some more decision problems are

i) Does DFA accept the empty language

ii) Is $L_1 \cap L_2 = \emptyset$ for regular sets

iii) If a language 'L' is decidable then its complement is also decidable.

iv) If a language is decidable then there is a Turing machine for it.

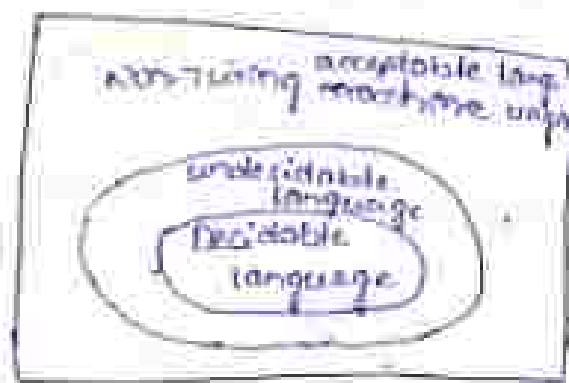
Undecidable problems:-

Introduction:-

* For an undecidable language there is no TM which accepts the language and makes a decision for every input string 'w'.

* A decision problem 'P' is called undecidable if the langth 'L' of all instances to 'P' is not decidable.

undecidable languages are not recursive languages but, sometimes they may be recursive enumerable languages.



Examples:-

- i) the halting problem of turing machine
- ii) the mortality problem.
- iii) the mortal matrix problem.
- iv) the post correspondence problem [pcp]

i) the halting problem:

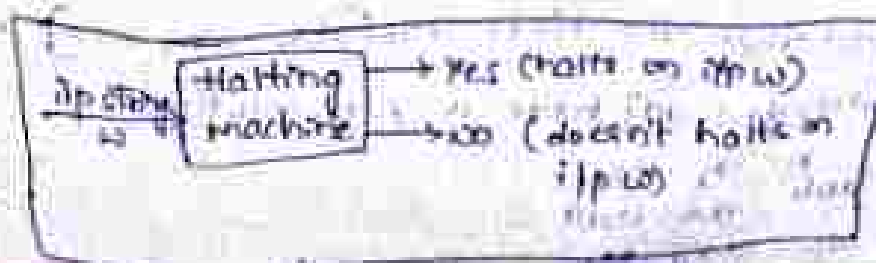
The halting problem inp: a turing machine and the inp string is.

problem: Does the turing machine finish computing of the string is in a finite no. of steps? The answer must be either yes (or) no.

Proof:- At first, we will assume that a turing machine exists to solve the problem we will show and then it is contradicting itself.

We will call this turing machine as a halting machine that produces a yes (or) no in a finite amount of time.

If the halting machine finishes in a finite amount of time then the inp comes as YES. otherwise, as NO.

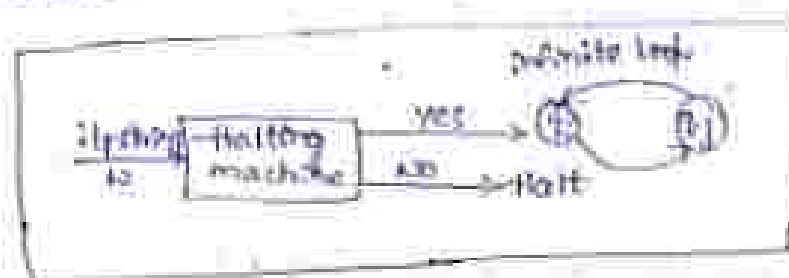


Now, we will design an inverted halting machine as

i) If HM returns yes then loop forever.

ii) If HM returns no then halt.

The following block diagram shows the inverted halting machine.



Further, a machine HM which ip itself is constructed as follows:

i) If HM halts on ip loop forever.

ii) Else halt.

∴ Here, we have got a contradiction. Hence, the halting problem is undecidable.

Post Correspondence problem (PCP):

• It was introduced by "Emil post" in 1946 is an undecidable decision problem.

The PCP problem over an alphabet " Σ " is stated as follows.

Given the following two lists, M and N , of non-empty strings over Σ :

$M = (x_1, x_2, x_3, \dots, x_n)$

$N = (y_1, y_2, y_3, \dots, y_n)$

We can say that there is a PCP solution if for some $i_1, i_2, i_3, \dots, i_k$ where $1 \leq i_k \leq n$, the condition

$x_{i_1} x_{i_2} x_{i_3} \dots x_{i_k} = y_{i_1} y_{i_2} y_{i_3} \dots y_{i_k}$ satisfies

Ex: - Find whether the lists $M = (abb, aa, aab)$ and

$N = (bba, aaa, aa)$ have a PCP solution.

Sol: -

	x_1	x_2	x_3
M	abb	aa	aab
N	bba	aaa	aa

Here $x_1, x_2, x_3 = aababaaa$

$y_1, y_2, y_3 = aababaaa$

we can see that $x_1, x_2, x_3 = y_1, y_2, y_3$

Hence, the solution is $i=2, j=1, k=3$

2) find whether the list $M = [ab, bab, bbaaa]$ and $N = [a, ba, bab]$ have a pep solution

sol:-

	x_1	x_2	x_3
M	ab	bab	bbaaa
N	a	ba	bab

In this case, there is no solution because

$|x_1, x_2| \neq |y_1, y_2|$ lengths are not same

Hence, it can be said that this pep is a undecidable problem.

Modified post correspondence problem:-

Given two lists $M = x_1, x_2, x_3, \dots, x_n$ and $N = y_1, y_2, y_3, \dots, y_n$

Given a set of pairs of strings $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$

then the solution is an instance such that,

$$x_1, x_2, x_3, \dots, x_n = y_1, y_2, y_3, \dots, y_n$$

that means the pair (x_i, y_i) is forced to be at the beginning of the strings

Ex:-

	x_1	x_2	x_3
M	11	100	111

	y_1	y_2	y_3
N	111	001	11

Then the solution is $x_1, x_2, x_3 = y_1, y_2, y_3$

$$x_1, x_2, x_3 = 11100111$$

$$y_1, y_2, y_3 = 11100111$$

That means it is essential to have x_i, y_i at the beginning

Part NP classes :-

- * P-problems
- * NP-problems
- * P vs NP

P-problems:-

* P is the class of problems that can be solved by deterministic algorithm in a polynomial time $f(n)$ where n is the size of input string.

* P-problem consist of a language accepted by deterministic Turing machine that runs in polynomial amount of time.

Ex:- 1) shortest path problem

2) equivalence of DFA and NFA

3) shortest cycle in a graph.

4) sorting algorithms

NP-problems:-

* NP-problem is a class of problems that can be solved by non-deterministic algorithms in a polynomial time $f(n)$ where n is the size of input string.

* NP-problems consists of a language accepted by non-deterministic Turing machine that runs in a polynomial amount of time.

Ex:- 1) Travelling salesman problem.

2) subgraph isomorphism

NP-problem classified into two types.

1) NP-hard problem.

2) NP-complete problem.

NP-hard problem:-

If there is a language x such that every language y in NP can be polynomially reducible to x , and we cannot prove that x is in NP, then x is said to be NP-hard problem.

Ex:- Turing machine halting problem.

NP-complete problem:-

If there is a language x such that every language

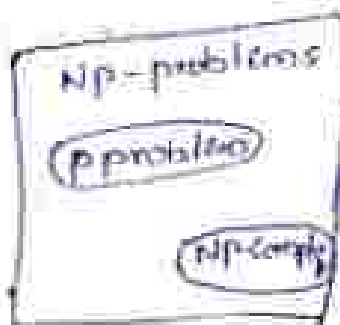
if a problem x can be polynomially reducible to y and we can prove that y is in NP then x is said to be NP-complete problems.

- Ex:- 1) Travelling salesman problem
2) Subgraph isomorphism

P vs NP :-

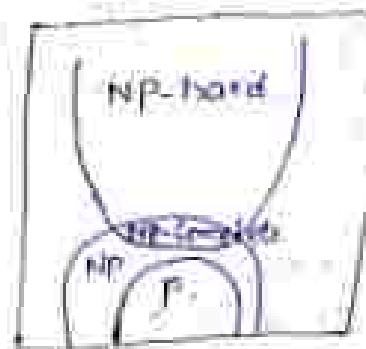
1. Kadner's theorem:

(a) $P \neq NP$



2. Euler's theorem:

(a) $P \neq NP$



(b) $P = NP$

