UNIT-5

Algebric Computation :

GCD (greatest Common Divisor)

Euclid Algorithm is used to find out GCD.

gcd (a, b)

1. If b = 0
2. then return a
3. else return euclid (b, a mod b)

Q a = 35 , b = 4

Sol Euclid (4, 35 mod 4)

euclid (4, 3)

euclid (3, 1)

euclid (1, 0)

= 1

Q Find the GCD of 2740, 1760.

~~euclid~~ GCD (2740, 1760)

euclid (1760, 980)

euclid (980, 780)

euclid (780, 200)

euclid (200, 180)

Scanned by CamScanner

euclid(180, 20)
euclid(20, 0)

= 20

* RSA (Asymmetric Key encryption)

Step-1: Select two prime number P and Q such that $P \neq Q$.

Step-2: find out $n = P * Q$

Step-3: Find out relative prime P and Q.

$$\phi(n) = (P-1) * (Q-1)$$

Step-4: $d * e \bmod \phi(n) = 1$

where d = private key
e = public key

Step-5: cipher text $C = m^e \bmod n$

Step-6: $m = C^d \bmod n$

Q Consider the RSA with $p=11$ & $Q=29$, and $e=3$, $m=100$. Find out encrypted msg.

Sol:

① $P = 11$, $Q = 29$

② $n = 11 * 29$

$\Rightarrow n = 319$

③ $\phi(n) = 10 \times 28$

$= 280$

Scanned by CamScanner

④ $d * 3 \bmod 280 = 1$

$\Rightarrow d = 187$

⑤ $c = 100^3 \bmod 319$

$c = 254$

⑥ $m = 254^{187} \bmod 319$

$m = 100$

String Matching :

String matching problem occur during the text editing ~~problem~~ program.

In text editing program, pattern search by the user. Now, we can ~~con~~clude the problem

$T(1 \ldots\ldots n)$ where $n \geq m$

$P(1 \ldots m)$

and we can match the pattern ~~from~~ with the text $\boxed{0 \leq s \leq n-m}$

There are five types of algorithms -

1) Naive String
2) Finite Automata
3) Rabin Karp
4) Knuth Moris
5) Boyer Moore

Scanned by CamScanner

1) Naive String:

It works on with the condition –

$$P(1.....m) = T(1.....m)$$

$$= T(S+1.....S+m)$$

Naive string (T, P)

1. n ← length (T)
2. m ← length (P)
3. for { s ← 0 to n-m
4. if P(1.....m) = T(S+1.... S+m)
5. return 'S'

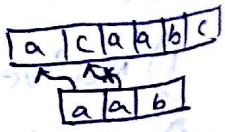
eg)

T = | a | c | a | a | b | c |

P = | a | a | b |

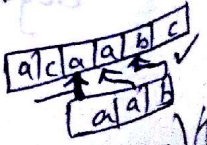
Sol)

1. n ← 6
2. m ← 3
3. for s ← 0 to 6-3=(3)
4. if P(1).....m) ≠ T(0+1.....0+m)



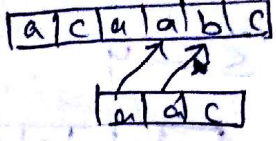
3. for s ← 1
4. if P(1.... m) = T(2.....1+m)

3. for s ← 2
4. if P(1.....m) = T(3..... 2+m)



Scanned by CamScanner

⑤ return 2



③ for s ← 3

④ if (P(1..... m) = T(4..... 3+m)) (false

③ for s ← 4 (false)

⇒ S = 2

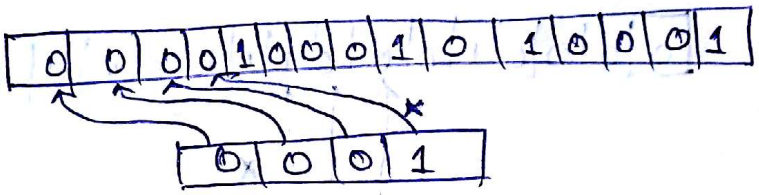
eg) T = 0 0 0 0 1 0 0 0 1 0 1 0 0 0 1

P = 0 0 0 1

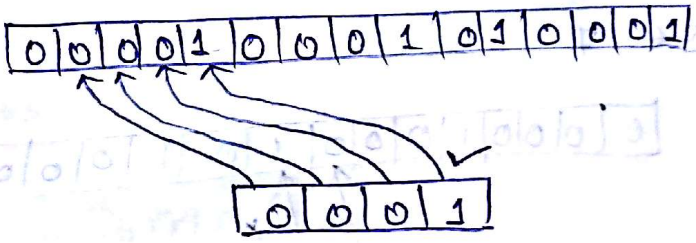
sol) ① n ← 15

② m ← 4

③ for s ← 0 to 11

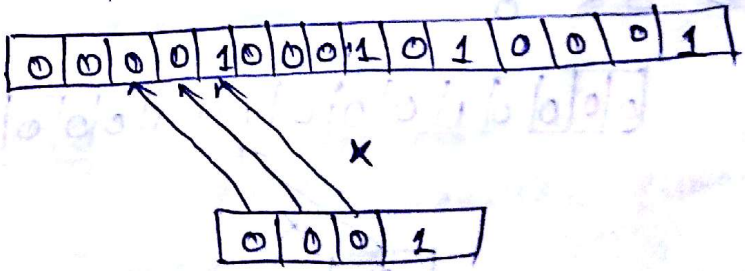


③ s ← 1

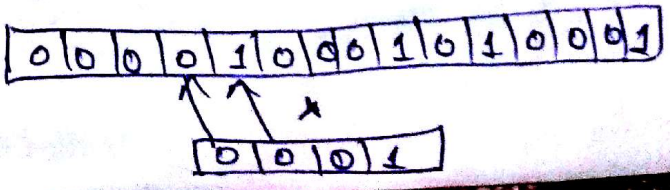


return 1

③ s ← 2

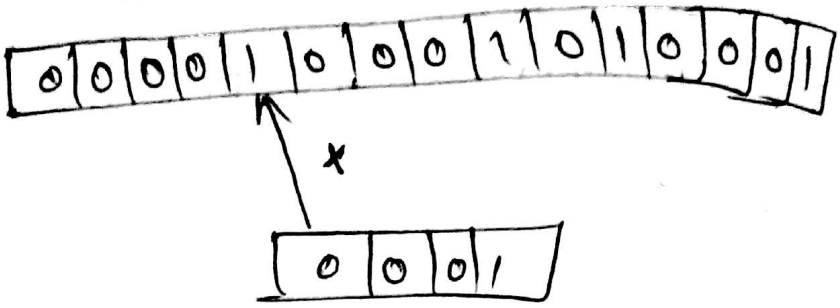


③ s ← 3

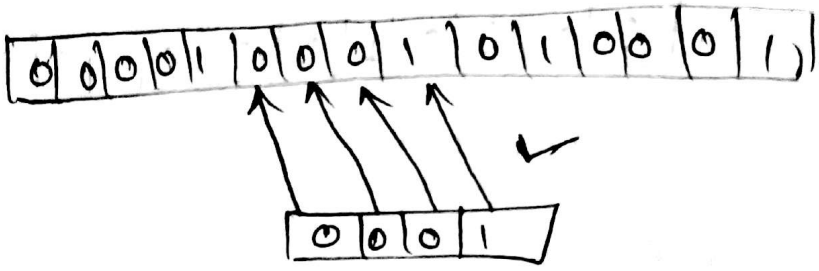


Scanned by CamScanner

(2) $S \leftarrow 4$

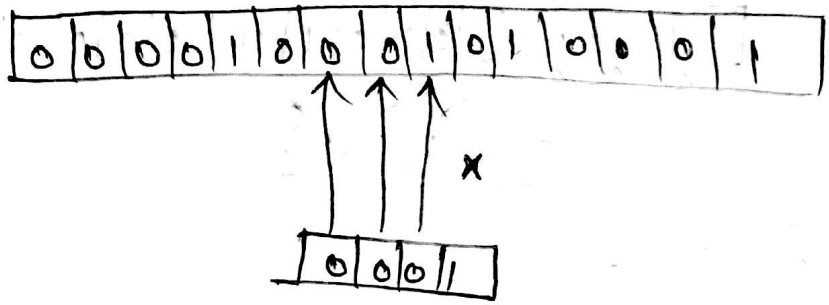


(3) $S \leftarrow 5$

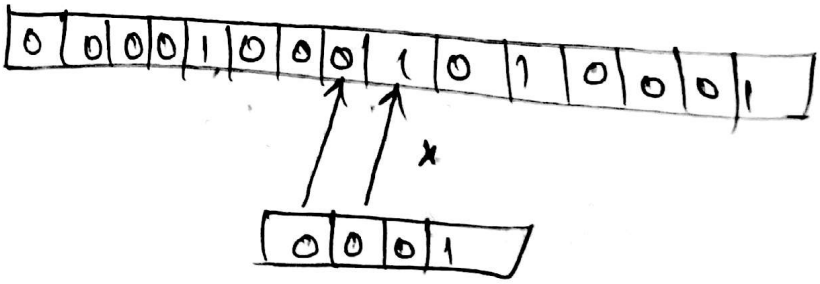


return 5

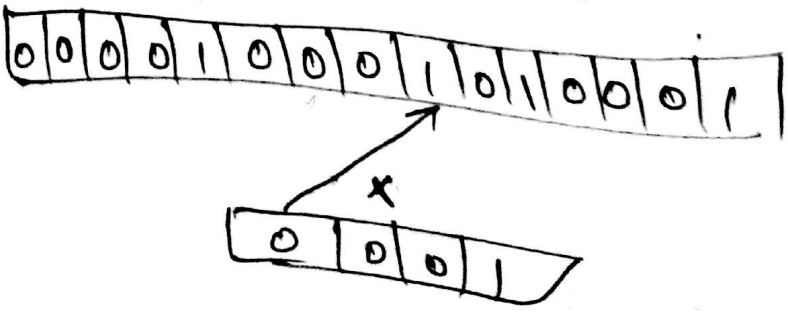
(3) $S \leftarrow 6$



(3) $S \leftarrow 7$

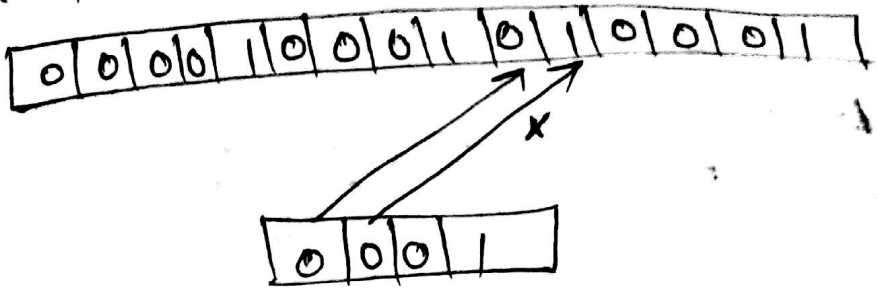


(3) $S \leftarrow 8$

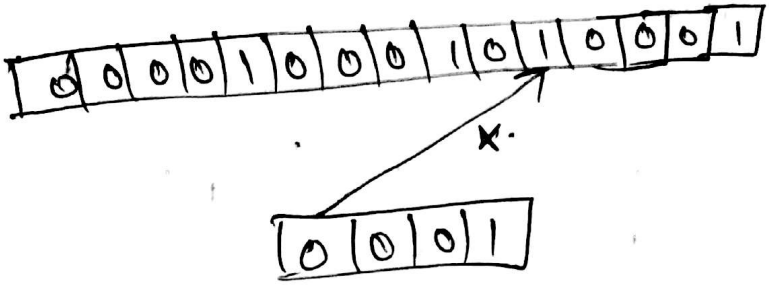


Scanned by CamScanner

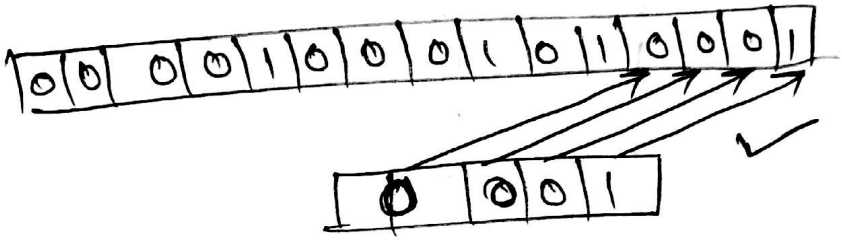
(3) $s \leftarrow 9$



(3) $s \leftarrow 10$



(9) $s \leftarrow 11$



return 11

$\Rightarrow$ s = 1, 5, 11

Finite Automata

There are 5 tuples for FA

q_0 – initial state

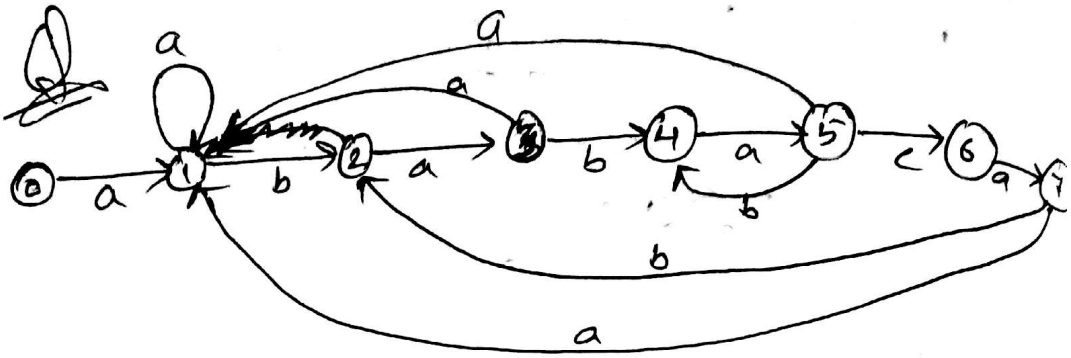
Σ – I/P symbol

Q – set of state

$\delta(Q, \Sigma) = Q$ – transition function

q_f – final state

Scanned by CamScanner



Transition table:

Q	Q' Input a	Input b	Input c
Q_0	1	0	0
Q_1	1	2	0
Q_2	3	0	0
Q_3	1	4	0
Q_4	5	0	0
Q_5	1	4	6
Q_6	7	0	0
Q_7	1	2	0

text - a b a b a b a c a b a

pattern - a b a b a c a

T(i) =	1 a	2 b	3 a	4 b	5 a	6 b	7 a	8 c	9 a	10 b	11 a
Q	0	1	2	3	4	5	4	5	6	7	2	3

text = 11 P_2 = 7

n = 11 m = 7

when q = m

return (i - m)

q = 7 for i = 9 ⇒ **S = 2**

Finite automata (T, S, M)

1) n ← length(T)
2) q ← 0
3) for i = 1 to n
4) q = δ(q, T(i))
5) if q = m
6) print shift(s) = i - m

Scanned by CamScanner

3) Boyer Moore :

In Boyer Moore, pattern match from right to left for that, we will use last function.

eg) text = a b a c a a b a c c

Pattern = a b a c a b
0 1 2 3 4 5

	a	b	c	d
last(c)	4	5	3	-1

Boyer Moore (T, P)

1. Compute last function
2. i ← m-1
3. j ← m-1
4. repeat
5. if P(j) = T(i) then
6. if j = 0 then
7. return i;
8. else i ← i-1
9. j ← j-1
10. else
11. i ← i + m - min(j, 1 + last[T[i]])
12. j ← m-1
13. until i > n-1

Scanned by CamScanner

Q T = a b a c a a b a d c a a b
0 1 2 3 4 5 6 7 8 9 10 11
a c a b a a b
12 13 14 15 16 17 18

P = a b a c a b
0 1 2 3 4 5

Sol ① Compute last function

	a	b	c	d
last(c)	4	5	3	-1

② i ← m-1 ⇒ 5

③ j ← m-1 ⇒ 5

④ repeat

⑤ if P(5) = T(5) false

⑩ else

⑪ i ← i + m - min(j, 1 + last[T[i]])

i ← 5 + 6 - min(5, 5) min(5, 1+4)

i ← 6

⑫ j ← m-1 ⇒ j ← 5

⑬ i > n-1 ⇒ 6 > 19 ~~(false)~~ do go to 4.

④ repeat

⑤ if P(5) = T(6) (true)

⑥ if j = 0 (false)

⑧ i ← 5

⑨ j ← 4

⑬ 5 > 19 do go to 4.

Scanned by CamScanner

④ repeat
⑤ if P(4)=T(5) (true)
⑤ if J=0 (false)
⑥ i ← 4
⑧ j ← 3
⑨ 4>19 so go to 4.
⑬
④ repeat
⑤ if P(3)=T(4) (false)
⑩ else
⑪ i ← 4+6-min(3,5)
i ← 7
⑫ j ← m-1 ⇒ j ← 5
⑬ 7>19 so go to 4.
④ repeat
⑤ if P(5)=T(7) (~~true~~ false)
⑩ ~~if J=0 (false)~~ else
⑪ i ← 7+6-min(5,5)
i ← 8
⑫ j ← 5
⑬ 8>19 so go to 4.
④ repeat
⑤ if P(5)=T(8) (false)
⑩ else
⑪ i ← 8+6-min(5,1-1)
i ← 14
⑫ j ← 5
⑬ 14>19 so go to 4

Scanned by CamScanner

④ repeat

⑤ if $P[5] = T[14]$ (false)

⑩ else

⑪ $i \leftarrow 14 + 6 - \min(5, 5)$

$i \leftarrow 15$

⑫ $j \leftarrow 5$

⑬ $15 > 19$ do go to 4.

④ repeat

⑤ if $P[5] = T[15]$ (true)

⑥ if $j = 0$ (false)

⑧ $i \leftarrow 14$

⑨ $j \leftarrow 4$

⑬ $14 > 19$ do go to 4.

④ repeat

⑤ if $P[4] = T[14]$ (true)

Scanned by CamScanner

⑥ if J=0 (false)
⑧ i ← 13
⑨ j ← 3

⑬ 13 > 19 so, go to 4.
④ repeat
⑤ if P[3] = T[13] (true)
⑥ if J=0 (false)
⑧ i ← 12
⑨ j ← 2.

⑬ 12 > 19 so go to 4.
④ repeat
⑤ if P[2] = T[12] (true)
⑥ if J=0 (false)
⑧ i ← 11
⑨ j ← 1

⑬ 11 > 19 so, go to 4.
④ repeat
⑤ if P[1] = T[11] (true)
⑥ if J=0 (false)
⑧ i ← 10
⑨ j ← 0

⑬ 10 > 19 so, go to 4.
④ repeat
⑤ if P[0] = T[10] (true)
⑥ if J=0 (true)
⑦ return i ⇒ 10

Scanned by CamScanner

Q) suffix → 10

Rabin Karp :

It is applicable for decimal numbers.
For a given pattern 1 to 'm' let 'p' denote its corresponding decimal value, and for a given text 'T' (1 to n) let 'ts' denotes its corresponding decimal value of the length 'm'.

Robin Karp (T, P, d, q)

1) n ← length (T)
2) m ← length (P)
3) h ← d^{m-1} mod q
4) p ← 0
5) ts ← 0
6) for i ← 1 to m
7) p ← (dp + P(i)) Mod q
8) to = (dto + T(i)) Mod q
9) for s ← 0 to (n-m)
10) if p = ts
11) then P(1.....m) = T(s+1, ..., s+m)
12) Pattern occurs with shift s.
13) if s < n-m
14) then t_{s+1} = (d(ts - T(s+1)h) + T(s+m+1)) mod q

Scanned by CamScanner

Q1

T = | 2 | 3 | 5 | 9 | 0 | 2 | 3 | 1 | 4 | 1 | 5 | 2 | 6 | 7 | 3 | 9 | 9 | 2 | 1 |

P = | 3 | 1 | 4 | 1 | 5 |
1 2 3 4 5

d = 10 (base)

and working with modulo, q = 13. find out how many valid match & spurious hit.

Sol

① $n \leftarrow 19$

② $m \leftarrow 5$

③ $h \leftarrow 10^4 \bmod 13$

$h \leftarrow 3$

④ $p \leftarrow 0$

⑤ $t_s \leftarrow 0$

⑥ for $i \leftarrow 1$ to 5

⑦ $p_0 \leftarrow (10 \times 0 + 3) \bmod 13$

$p_0 \leftarrow 3$

⑧ $t_0 \leftarrow (10 \times 0 + 2) \bmod 13$

$t_0 \leftarrow 2$

⑥ for $i \leftarrow 2$

⑦ $p_0 \leftarrow 5$

⑧ $t_0 \leftarrow 10$

⑥ for $i \leftarrow 3$

⑦ $p_0 \leftarrow 2$

⑧ $t_0 \leftarrow 1$

⑥ for $i \leftarrow 4$

⑦ $p_0 \leftarrow 8$

⑧ $t_0 \leftarrow 6$

⑥ for $i \leftarrow 5$

⑦ $p_0 \leftarrow 7$

⑧ $t_0 \leftarrow 8$

Scanned by CamScanner

(9) for $s \leftarrow 0$

$p = 31415 \bmod 13 = 7$

$t_s = 23590 \bmod 13 = 8$

(10) $p \neq t_s$

(9) for $s \leftarrow 1$

$p = 7$

$t_s = 35902 \bmod 13 = 9$

(10) $p \neq t_s$ (#

(9) for $s \leftarrow 2$

$p = 7$

$t_s = 59023 \bmod 13 = 3$

(10) $7 \neq 3$ (false)

(9) for $s \leftarrow 3$

$p = 7$

$t_s = 90231 \bmod 13 = 11$

(10) $7 \neq 11$ (false)

(9) for $s \leftarrow 4$

$p = 7$

$t_s = 02314 \bmod 13 = 0$

(10) $7 \neq 11$ (false)

(9) for $s \leftarrow 5$

$p = 7$

$t_s = 23141 \bmod 13 = 1$

(10) $7 \neq 1$ (false)

(9) for $s \leftarrow 6$ ——— **VALID MATCH**

$p = 7$

$t_s = 31415 \bmod 13 = 7$

(10) $7 = 7$ (true)

Scanned by CamScanner

(9) for $s \leftarrow 7$

$p = 7$

$t_s = 14152 \bmod 13 = 8$

(10) $7 \neq 8$ (false)

(9) for $s \leftarrow 8$

$p = 7$

$t_s = 41526 \bmod 13 = 4$

(10) $7 \neq 4$ (false)

(9) for $s \leftarrow 9$

$p = 7$

$t_s = 15267 \bmod 13 = 5$

(10) $7 \neq 5$ (false)

(9) for $s \leftarrow 10$

$p = 7$

$t_s = 52673 \bmod 13 = 10$

(10) $7 \neq 10$ (false)

(9) for $s \leftarrow 11$

$p = 7$

$t_s = 26739 \bmod 13 = 11$

$7 \neq 11$ (false)

——— SPURIOUS HIT

(9) for $s \leftarrow 12$

$p = 7$

$t_s = 67399 \bmod 13 = 7$

(10) $7 = 7$ (true)

(9) for $s \leftarrow 13$

$p = 7$

$t_s = 73992 \bmod 13 = 9$

(10) $7 \neq 9$ (false)

(9) for $s \leftarrow 14$

$p = 7$

$t_s = 39921 \bmod 13 = 11$

Scanned by CamScanner

(10) 7≠11 (false)

Here is only one valid hit and one spurious hit at s=6 & s=12 respectively

(9)

q=11

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
T =	3	1	4	1	5	9	2	6	5	3	5	8	9	7	9	3

	1	2
P =	2	6

Find no. of spurious hit.

Sol

(1) n ← 16

(2) m ← 2

(3) h ← $10^{1} \bmod 11$

h ← 10

(4) p ← 0

(5) t_s ← 0

(6) for i ← 1 to 2.

(7) p ← $(10 \times p(1)) \bmod 11$ ⇒ p ← 2

(8) t_0 ← $(10 \times 0 + T(1)) \bmod 11$ ⇒ t_0 ← 3

(6) for i ← 2

(7) p ← $(10 \times 2 + p(2)) \bmod 11$ ⇒ p ← 4

(8) t_0 ← $(10 \times 3 + T(2)) \bmod 11$ ⇒ t_0 ← 9

(9) for s ← 0 to (n−m) i.e., 14

p ← 25 mod 11 = 4

t_s ← 31 mod 11 = 9

(10) p ≠ t_s

(9) for s ← 1

p ← 4

t_s ← 14 mod 11 = 3

Scanned by CamScanner

(10) p ≠ ts

(9) for S ← 2

p ← 4

ts ← 41 mod 11 = 8

(10) p ≠ ts

(9) for S ← 3 ——— SPURIOUS HIT

p ← 4

ts ← 15 mod 11 = 4

(10) 4 = 4 (true)

(9) for S ← 4 ——— Spurious Hit

p ← 4

ts ← 59 mod 11 = 4

(10) 4 = 4 (true)

(9) for S ← 5 ——— Spurious hit.

p ← 4

ts ← 92 mod 11 = 4

(10) 4 = 4 (true)

(9) for S ← 6 ——— valid match

p ← 4

ts ← 26 mod 11 = 4

(10) 4 = 4 (true)

(9) for S ← 7

p ← 4

ts ← 65 mod 11 = 10

(10) 4 ≠ 10 (false)

(9) for S ← 8

p ← 4

ts ← 53 mod 11 = 9

(10) 4 ≠ 9 (false)

Scanned by CamScanner

(9) for $S \leftarrow 9$

$p \leftarrow 4$

$ts \leftarrow 35 \bmod 11 = 3$

(10) $4 \neq 3$ (false)

(9) for $S \leftarrow 10$

$p \leftarrow 4$

$ts \leftarrow 58 \bmod 11 = 3$

(10) $4 \neq 3$ (false)

(9) for $S \leftarrow 11$

$p \leftarrow 4$

$ts \leftarrow 89 \bmod 11 = 1$

(10) $4 \neq 1$ (false)

(9) for $S \leftarrow 12$

$p \leftarrow 4$

$ts \leftarrow 97 \bmod 11 = 9$

(10) $4 \neq 9$ (false)

(9) for $S \leftarrow 13$

$p \leftarrow 4$

$ts \leftarrow 79 \bmod 11 = 2$

(10) $4 \neq 2$ (false)

(9) for $S \leftarrow 14$

$p \leftarrow 4$

$ts \leftarrow 93 \bmod 11 = 5$

(10) $4 \neq 5$ (false)

Here are 3 spurious hit on $S = 3, 4, 5$ and 1 valid hit on $S = 6$.

Scanned by CamScanner

Knuth Morris Patt : [KMP]

we perform many comparisons while text is a placement of the pattern against, if we discover a pattern that does not match in the text then we away all the information gain by these comparisons and start again from next implementation placement of the pattern. It means in KMP, we can discard the invalid shifts/match with the help of prefix function. prefix function is denoted by $\pi(1 \text{ to } m)$

compute prefix function (P)

```
(1) m ← length(P)
(2) π(1) ← 0
(3) K ← 0
(4) for q ← 2 to m
(5) while K > 0 and P(K+1) ≠ P(q)
(6)   then K ← π(K)
(7) if P(K+1) = P(q)
(8)   K ← K+1
(9)   π(q) ← K
(10) return π
```

Scanned by CamScanner

Q. Compute the prefix fn ϕ for the ~~text~~ Pattern

P = | A | B | A | B | A | B | A | B | C | A |

Sol: ① $m \leftarrow 10$

	1	2	3	4	5	6	7	8	9	10
	a	b	a	b	a	b	a	b	c	a
$\pi(p)$	0	0	1	2	3	4	5	6	0	1

② $\pi(1) \leftarrow 0$

③ $k \leftarrow 0$

④ for $q \leftarrow 2$ to 10

⑤ while $k > 0$ (false)

⑦ if ($p(1) = p(2)$) $(a \neq b)$ false

⑨ $\pi(2) \leftarrow 0$

④ for $q \leftarrow 3$

⑤ while $k > 0$ (false)

⑦ if $(p(1) = p(3))$ $a = a$ true

⑧ $k \leftarrow k+1 \Rightarrow k \leftarrow 1$

⑨ $\pi(3) \leftarrow 1$

④ for $q \leftarrow 4$

⑤ while ~~0~~ While $1 > 0$ and $p(2) \neq p(4)$ (false)

⑦ if $(p(2) = p(4))$ $b = b$ true

⑧ $k \leftarrow k+1 \Rightarrow k \leftarrow 2$.

⑨ $\pi(4) \leftarrow 2$.

Scanned by CamScanner

④ for $q \leftarrow 5$

⑤ while $(2 > 0)$ and $P(3) \neq P(5)$ (false)

⑦ if $P(3) = P(5)$ $(a=a)$ (true)

⑧ $K \leftarrow K+1 \Rightarrow K \leftarrow 3$

⑨ $\pi(5) \leftarrow 3$

④ for $q \leftarrow 6$

⑤ while $(3 > 0)$ and $P(4) \neq P(6)$ false)

⑦ if $P(4) = P(6)$ $(b=b)$ (true)

⑧ $K \leftarrow K+1 \Rightarrow K \leftarrow 4$

⑨ $\pi(6) \leftarrow 4$

④ for $q \leftarrow 7$

⑤ while $(4 > 0)$ and $P(5) \neq P(7)$ (false)

⑦ if $P(5) = P(7)$ $(a=a)$ (true)

⑧ $K \leftarrow K+1 \Rightarrow K \leftarrow 5$

⑨ $\pi(7) \leftarrow 5$

④ for $q \leftarrow 8$

⑤ while $(5 > 0)$ and $P(6) \neq P(8)$ (false)

⑦ if $P(6) = P(8)$ $(b=b)$ true)

⑧ $K \leftarrow K+1 \Rightarrow K \leftarrow 6$

⑨ $\pi(8) \leftarrow 6$

④ for $q \leftarrow 9$

⑤ while $(6 > 0)$ and $P(7) \neq P(9)$ (true)

⑥ $K \leftarrow \pi(K) \Rightarrow K \leftarrow 4$

⑤ ~~while $(5 > 0)$ and $P(5) \neq P(9)$ (false)~~ while $(K > 0)$ and $P(5) \neq P(9)$ (true)

~~$\pi(6) \leftarrow 4$~~

⑥ $K \leftarrow \pi(K) \Rightarrow K \leftarrow 2$

Scanned by CamScanner

(5) while (k>0) and P(3) ≠ P(q) (true)

(6) $k \leftarrow \pi(k) \Rightarrow k \leftarrow 0$

(5) while (0>0) false)

(7) if P(1) = P(9) (false)

(9) $\pi(q) \leftarrow 0$

(4) for $2 \leftarrow 10$

(5) while (0>0) false)

(7) if P(2) = P(10)

(8) $k \leftarrow k+1$ & $k \leftarrow 1$

(9) $\pi(10) \leftarrow 1$

(4) for $2 \leftarrow 11$ (false)

Q = P = a b a b b a b a a

Sol ① $m \leftarrow 9$

	1	2	3	4	5	6	7	8	9
	a	b	a	b	b	a	b	a	a
π(p)	0	0	1	2	0	1	2	3	1

② $\pi(1) \leftarrow 0$

③ $k \leftarrow 0$

Scanned by CamScanner

④ for $q \leftarrow 2$ to 9.

⑤ while $k > 0$ (false)

⑦ if $(P(1) = P(2))$ false.

⑨ $\pi(2) \leftarrow 0$

④ for $q \leftarrow 3$

⑤ while $k > 0$ (false)

⑦ if $P(1) = P(3)$. (true)

⑧ $k \leftarrow k+1$ ⇒ $k \leftarrow 1$

⑨ $\pi(3) \leftarrow 1$

④ for $q \leftarrow 4$

⑤ while $1 > 0$ and $P(2) \neq P(4)$ (false)

⑦ if $(P(2) = P(4))$ (true)

⑧ $k \leftarrow k+1$ ⇒ $k \leftarrow 2$

⑨ $\pi(4) \leftarrow 2$

④ for $q \leftarrow 5$

⑤ while $2 > 0$ and $P(3) \neq P(5)$ (true)

⑥ $k \leftarrow \pi(k)$ ⇒ $k \leftarrow 0$

~~⑦ if~~

⑤ while $0 > 0$ (false)

⑦ if $(P(1) = P(5))$. (false)

⑨ $\pi(5) \leftarrow 0$

④ for $q \leftarrow 6$

⑤ while $0 > 0$ (false)

⑦ if $(P(1) = P(6))$ (true)

⑧ $k \leftarrow k+1$ ⇒ $k \leftarrow 1$

⑨ $\pi(6) \leftarrow 1$

Scanned by CamScanner

④ for $q \leftarrow 7$

⑤ while $1 > 0$ and $P(2) \neq P(7)$ (false)

⑦ if $(P(2) = P(7))$

⑧ $K \leftarrow K+1 \Rightarrow K \leftarrow 2$

⑨ $\pi(7) \leftarrow 2$

④ for $q \leftarrow 8$

⑤ while $2 > 0$ and $P(3) \neq P(8)$ (false)

⑦ if $(P(3) = P(8))$

⑧ $K \leftarrow K+1 \Rightarrow K \leftarrow 3$

⑨ $\pi(8) \leftarrow 3$

④ for $q \leftarrow 9$

⑤ while $3 > 0$ and $P(4) \neq P(9)$ (true)

⑥ $K \leftarrow \pi(K) \Rightarrow K \leftarrow 1$

⑤ while $1 > 0$ and $P(2) \neq P(9)$ (false)

⑦ if $(P(2) = P(9))$ (false)

⑨ $\pi(9) \leftarrow 1$

⑤ for $q \leftarrow 10$ (false)

	1	2	3	4	5	6	7	8	9
Pattern	a	b	a	b	b	a	b	a	a
$\pi(P)$	0	0	1	2	0	1	2	3	1

Scanned by CamScanner

NP Completeness :

Till now we have studied polynomial time algorithm where input in 'n' & complexity is $O(n^k)$ for some constant 'k'.

Now, I want to ask a question, that Every algorithm can be solved in polynomial time? answer is no, such as TSP, hamiltonian cycle, and halting problem can't be solved in polynomial time.

Now the other interesting class of problem is based on complexity.

these are NP problems which cannot be solved in polynomial time. Till now no polynomial time algorithm has been discovered for an NP problem.

Now, We can divide the problem in three category -

1) P
2) NP
3) NP completeness.

1) P problem : the class P contains those problems that are solved in Polynomial time.

eg) Sorting, Searching, MST.

Scanned by CamScanner

2) NP problem: NP contains of those problems that are verified in polynomial time. It means they can give a certificate of a solutions.
NP problems depends on the size of the input
eg TSP, ~~Circuit~~ Hamiltonian problem.

Now we can say that 'P' is the subset of NP

$$P \subseteq NP$$

3) NP Completeness problem: A problem is in the class of ~~Com~~ NP Completeness if it satisfies —
i) NP: It verifies in polynomial time
ii) NP-hard: If any NP-Complete problem can be solved in polynomial time then it is called NP-Hard
It means for NP-hard problem, polynomial time algorithm exists.

Difference b/w decision and optimization problem

Optimization problem

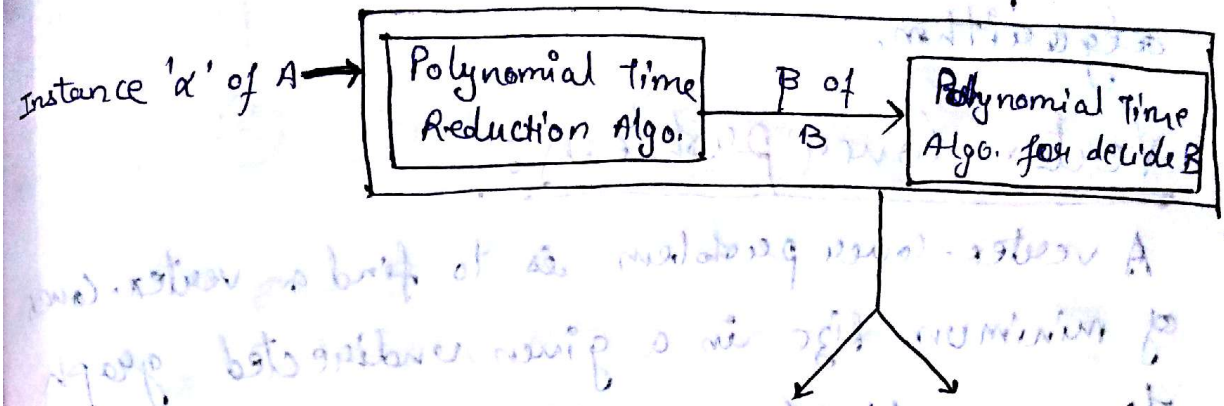
These are those problems that gives the optimal sol?. It means more than one sol? can exist.

Decision problem

Those problems that give '0' or '1' sol?
It means, it gives only two solutions

Scanned by CamScanner

Reductions :



NP completeness :

A language $L \subseteq (0,1)^*$ is NP complete if

i) $L \in NP$

ii) $L' \leq_p L$

If a language 'L' satisfies property (ii) then it is called NP-Hard not necessary property (i)

Approximation Algorithms

In NP complete problem, till now we don't know how to find an optimal Sol^n in polynomial time.

Now, we have atleast three ways to get around NP Completeness —

i) If the actual inputs are reduced.

ii) we may be able to isolate important special case that we can solve in polynomial time.

iii) Find out the near about optimal Sol^n in Polynomial time.

Scanned by CamScanner

we call an algorithm that returns near optimal sol^n, an approximation algorithm.

Vertex-Cover problem:

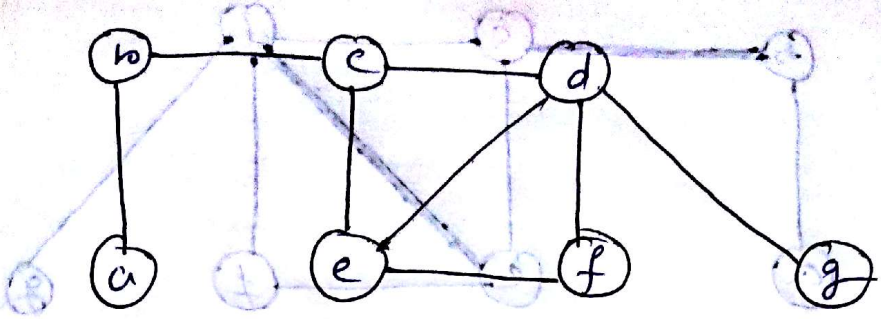
A vertex-cover problem is to find a vertex-cover of minimum size in a given undirected graph. It means all edge should be cover with some set of vertex, the size of vertex should be minimum. But, vertex cover problem is the category of NP-complete problem so, we can not find out optimal sol^n so, by approximation algorithm we will find out near about optimal vertex.

Approx vertex Cover (G)

1) $C \leftarrow \phi$
2) $E' \leftarrow G(E)$
3) while $E' \neq \phi$
   4) let (G,V) be an arbitrary edge of E'.
   5) $C = C \cup (u,v)$
   6) remove from E' every edge incidence on u or v.
7) return C.

Scanned by CamScanner

eg)



Sol:

1) $C \leftarrow \phi$

2) $E' \leftarrow \{ab, bc, ce, cd, de, df, ef, dg\}$

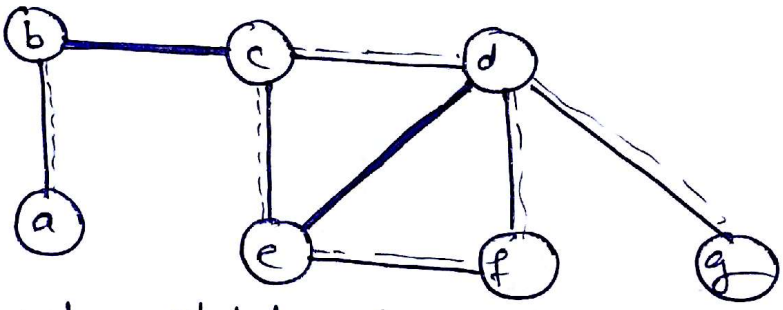
3) while $E' \neq \phi$ (true)

4) select (b, c)

5) $C = \phi \cup (b, c) \Rightarrow C = (b, c)$

6) remove (ab, ce, cd) from E'

$\Rightarrow E' \leftarrow \{de, df, ef, dg\}$



3) while $E' \neq \phi$ (true)

4) select (d, e)

5) $C = (b, c) \cup (d, e)$

$\Rightarrow C = \{b, c, d, e\}$

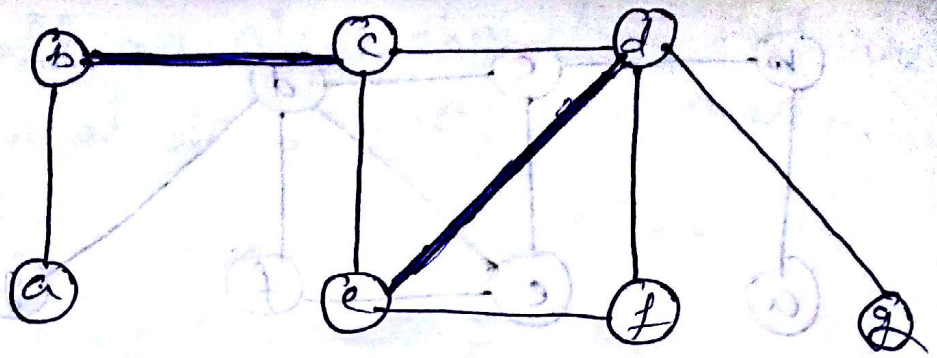
6) remove (df, ef, dg) from E'

$\Rightarrow E' \leftarrow \{\phi\}$

3) while $E' \neq \phi$ (false)

7) return C $\Rightarrow$ $\boxed{C = \{b, c, d, e\}}$

Scanned by CamScanner

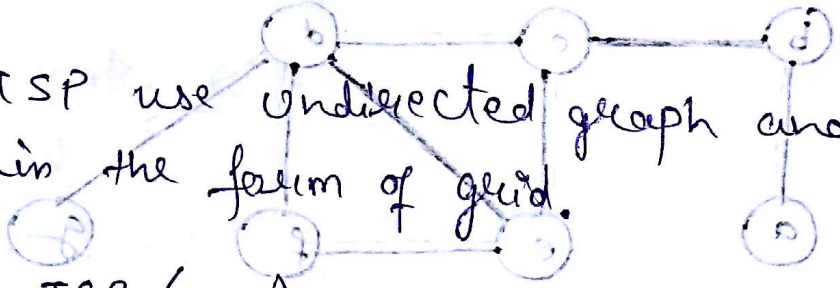


TSP (Travelling Salesman Problem)

Here, TSP problem satisfy the triangle in-equility means $u, v, w \in V$ then

$$c(u,w) \leq c(u,v) + c(v,w)$~~It is called triangle inequility~~

It means triangle inequility satisfies the property of equilidian distance.



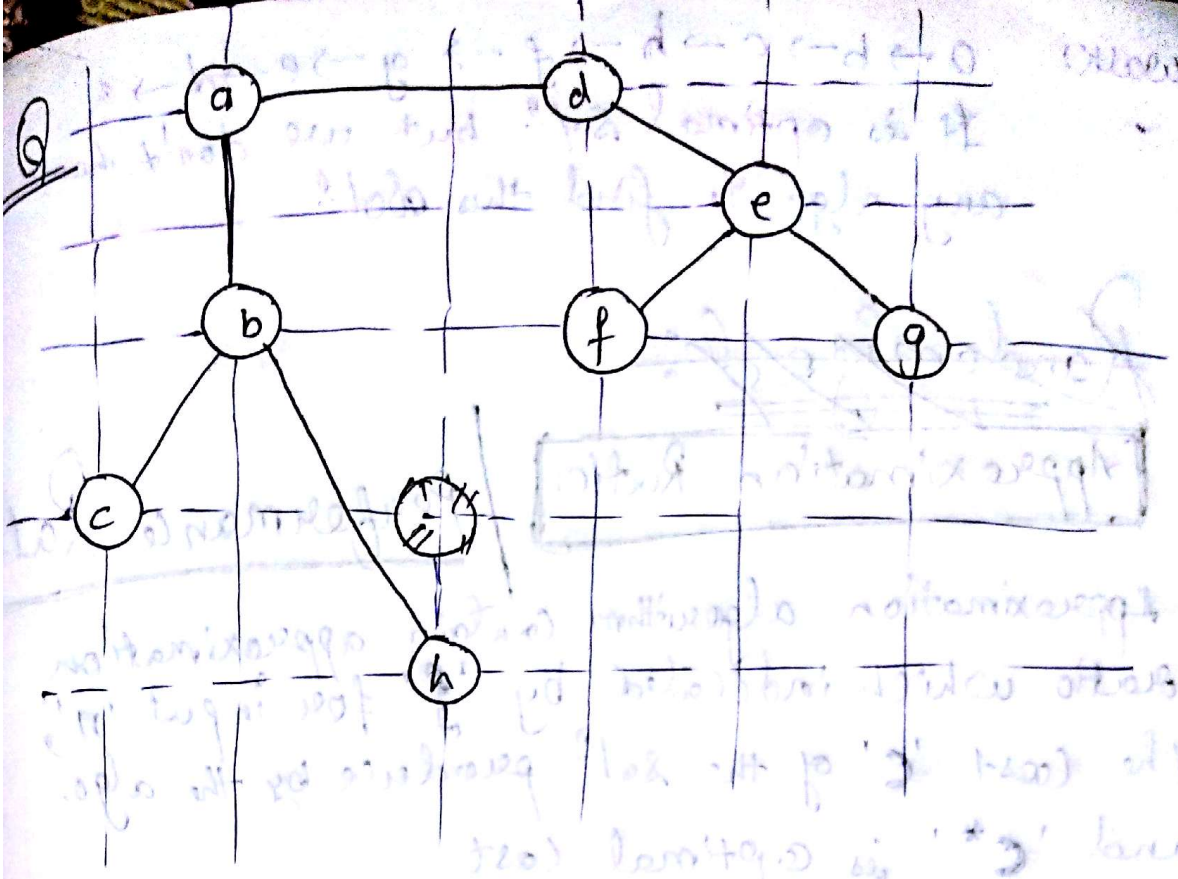
Here TSP use undirected graph and distance given in the form of grid.

Approx. TSP (G, C)

1) Select a vertex 'R' $\in$ G which is called root.
2) Compute a minimum spanning tree for G from root 'R' using Prim's algo.
3) Let H be a list of vertex order according to when they are first visit in a preorder tree walk.
4) return the hamiltonian cycle.

Scanned by CamScanner

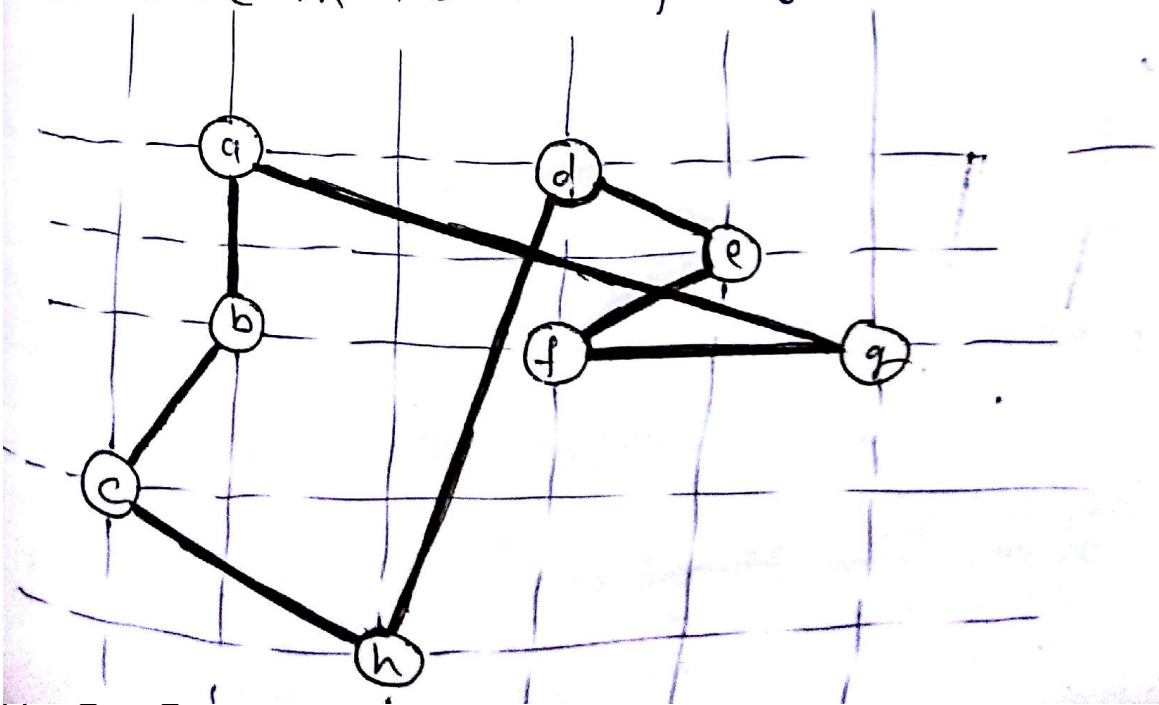
Q



Walk :

$a \to b \to c \to b \to h \to b \to a \to d \to e \to f$
$\to e \to g \to e \to d \to a$

$a \to b \to c \to h \to d \to e \to f \to g \to a$



Scanned by CamScanner

Walk: $a \to b \to c \to h \to f \to g \to e \to d \to a$

It is optimal sol^n but we don't have any algo to find this sol^n

~~Randomized~~

Approximation Ratio / **Performance Ratio**

Approximation algorithm contain approximation ratio which indicated by 'ρ' for input 'n', the cost 'C' of the sol^n produce by the algo. and 'C^*' is optimal cost

$$\max\left(\frac{C}{C^*}, \frac{C^*}{C}\right) \leq \rho(n)$$

Scanned by CamScanner