Shri Shankaracharya Institute of Professional Management & Technology, Raipur

August -2022- Class Test-2

Date: 06 / 08 /2022

Student Name: V. Om Sai Nageshwar sharma

Roll No.:	3	0	3	3	0	2	2	2	0	0	2	0

Enrollment No.:	B	J	4	5	9	9

Course: B.Tech Semester: 4th

Branch: CSE

Subject Name: ~~DAA~~ ADA

Subject Code:	B	0	2	2	4	1	5	(	0	2	2	)

Mobile No.: 8602727389

Email id: nageshwar.sharma@ssipmt.com

Signature: V

Q B.(a)>

A)

Algorithm: Greedy-Fractional-Knapsack
(w[1..n], p[1..n], W)

```
for i = 1 to n
do x[i] = 0
weight = 0
 for i = 1 to n
 if weight + w[i] ≤ W then
  x[i] = 1
  weight = weight + w[i]
  else
  x[i] = (W - weight) / w[i]
  weight = W
 break
 return x.
```

eg: Let us consider that the capacity of the knapsack W = 60 and the list of provided in items are shown in the following table -

items	A	B	C	D
Profit	280	100	120	120
Weight	40	10	20	24
Ratio $\left(\frac{P_i}{W_i}\right)$	7	10	6	5

As the provided items are not sorted based on $\frac{P_i}{W_i}$. After sorting, the items are as shown in the following table.

Item	B	A	C	D
Profit	100	280	120	120
Weight	10	40	20	24
Ratio $\left(\frac{P_i}{W_i}\right)$	10	7	6	5

After sorting all the items according to $\frac{P_i}{W_i}$ First all of B is choosen as ~~the~~ weight of B is less than the capacity of the knapsack. Next, item A is chosen, as the available capacity of the knapsack is greater than weight of A. Now, C is chosen as the next item.

However, the whole item A is chosen, as the cannot be chosen as the remaining capacity of the knapsack is less than the weight of C.

Hence, fraction of C (i.e. (60-50)/20) is chosen.

Now, the capacity of the knapsack is equal to the selected items.

The total weight of the selected item is 10 + 40 + 20 * (10/20) = 60

the total profit is 100 + 280 + 120 * (10/20) = 380 + 60 = 440.

Q^{E} >

Ay

NP Problems are those sets of problems for which a typical user cannot find the solutions very easily. While finding solutions for an NP Problem is difficult - they are still very easy to verify. A Non-Deterministic Model can easily solve this type of problem in a given polynomial time. In this article, we will discuss the difference between NP-Hard and NP-complete Problem. But let us understand their individual ~~their~~ purposes in detail.

Any given problem X acts as NP-Hard only if there exists a problem Y that is NP-complete. Here, problem Y becomes reducible to problem X in a polynomial time.

The hardness of an NP-Hard problem is equivalent to that of the NP-Complete Problem. But here, the NP-Hard problems don't need to be in the NP class.

NP-Complete Problem:

Any given problem X acts as NP-Complete when there exists an NP problem Y-so that the problem X in a polynomial line. This means that a given problem can only becomes NP-Complete if it is a part of NP-Hard as well as NP Problems. A Turning machine of non-deterministic nature can easily solve this type of problem in a given polynomial time.

Q A)

Ans

→ **Boyer-Moore Algorithm :**

The Boyer- algorithm takes a backward approach the pattern string(P) is aligned with the start of the text string T and then compares the characters of a pattern from right to left, begining with right most character.

→ Given,

T : G C A A T G C C T A T G T G G A C C

P : T A T G T G

⇒ Bad match table

P	T	A	G	*
Value	1	4	2	6
index	0	1	3	

value = length (P) - index - 1

① Pass 1

Matching last two characters Text(T) with Pattern (P)

T: GC AATG CC TATGTG ACC

P: ~~TA~~ TATGTG

Shift = 1

② Pass 2

As last characters of P and T are not matching. so, considering value of C for pattern shifting.

T: G C A A T G C C T A T G T G A C C

T A T G T G

Shift = 1 + 6 = 7

③ Pass 3

As last characters of P and T are not matching. So, considering value of T from match table for pattern shifting.

T: G C A A T G C C T A T G T G A C C.

P: TA T G T G

Shift = 1 + 6 + 1 = 8.

④. Pass 4

As the pattern matches from text. so, no more shifting is required and the processing is stopped and pattern occur with shift 8.

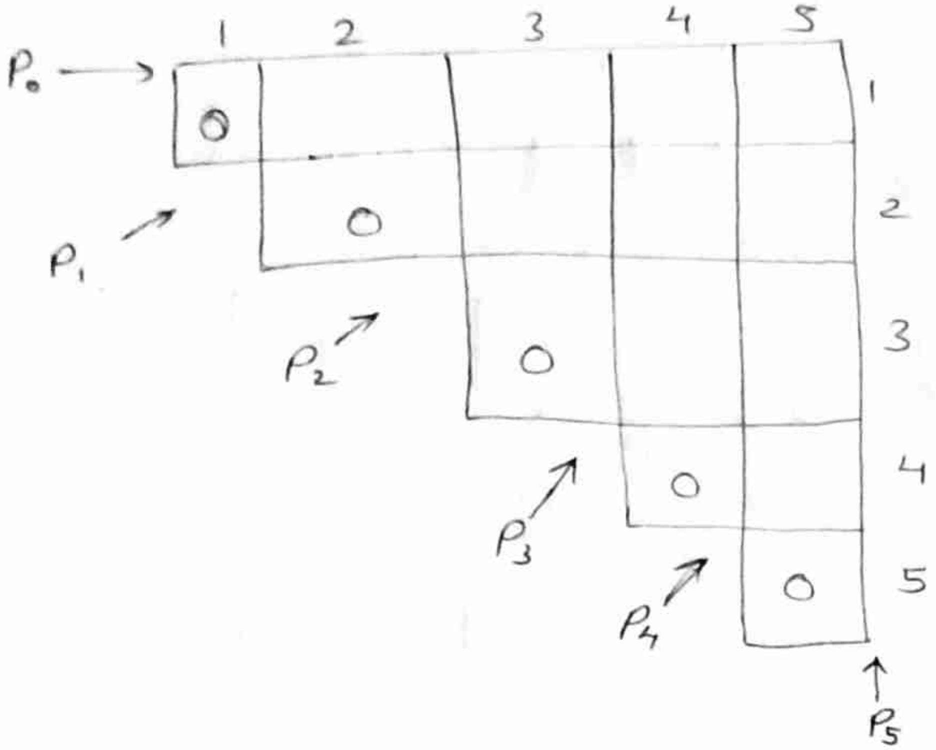
Q6 D)

Ans

Given Sequence {4, 10, 3, 12, 20, 7}

Matrix have Size : M_1 M_2 M_3 M_4 M_5

4×10 10×3 3×12 12×20 20×7



Calculation of Product of 2 Matrices:

1.) $M[1, 2] = m_1 \times m_2$

$= 4 \times 10 \times 10 \times 3$

$= 120$

2.) $M[2, 3] = M_2 \times M_3$

$= 10 \times 3 \times 3 \times 12$

$= 360$

3.) $M[3,4] = M_3 \times M_4$

$= 3 \times 12 \times 12 \times 20$

$= 720$

4) $M[4,5] = M_4 \times M_5$

$= 12 \times 20 \times 20 \times 7$

$= 1680$

Now,

<u>Calculate product of 3 Matrices</u>

	1	2	3	4	5
1	0	120			
2		0	360		
3			0	720	
4				0	1680
5					0

① $M[1,3] = M_1 \times M_2 \times M_3$

$M[1,3] = \min \begin{cases} M[1,2] + M[3,3] + P_0 P_2 P_3 \\ = 120 + 0 + 4 \times 3 \times 12 \\ = 264 \\ M[1,1] + M[2,3] + P_0 P_1 P_3 \\ = 0 + 360 + 4 \times 10 \times 12 \\ = 840 \end{cases}$

so, we will consider $M[1,3] = 264$.

② $M[2,4] = M_2 . M_3 . M_4$

$M[2,4] = \min \begin{cases} M[2,3] + M[4,4] + P_1 P_3 P_4 \\ 360 + 0 + 10 . 12 . 20 = 2760 \\ M[2,2] + M[3,4] + P_1 P_2 P_4 \\ = 0 + 720 + 10 \times 3 \times 20 = 1320 \end{cases}$

so, $M[2,4] = 1320$

$M[3,5] = M_3 \cdot M_4 \cdot M_5$

$$M[3,5] = \min \begin{cases} M[3,4] + M[5,5] + P_2 \times P_4 \times P_5 \\ = 720 + 0 + 3 \times 20 \times 7 = 1140 \\ M[3,3] + M[4,5] + P_2 P_3 P_5 \\ = 0 + 1680 + 3 \times 12 \times 7 \\ = 1932 \end{cases}$$

Product of 4 Matrices:

① $M[1,4] = M_1 \, M_2 \, M_3 \, M_4$

$M[1,4] = \min$

$M[1,3] + M[4,4] + P_0 P_3 P_4$
$= 264 + 0 + 4 \times 12 \times 120 = 1224$

$M[1,2] + M[3,4] + P_0 P_2 P_4$
$= 120 + 720 + 4 \times 3 \times 20 = 1080$

$M[1,1] + M[2,4] + P_0 P_1 P_4$
$= 0 + 1320 + 4 \times 10 \times 20$
$= 2120$

1	2	3	4	5	
0	120	264			1
	0	360	120		2
		0	720	1140	3
			0	1680	4
				0	5

So, $M[1,4] = 1080$

$M[2,5] = \min$

$M[2,4] + m[5,5] + P_1 P_4 P_5$
$= 1320 + 0 + 10 \times 20 \times 7 = 2720$

$M[2,3] + M[4,5] + P_1 P_3 P_5$
$= 360 + 1680 + 10 \times 12 \times 7$
$= 2880$

$M[2,2] + M[3,5] + P_1 P_2 P_5$
$= 0 + 1140 + 10 \times 3 \times 7 = 1350$

So, $M[1,5] = 1344$

Now Product of 5 Matrices :

	1	2	3	4	5	
	0	120	264	1080		1
		0	360	1320	1350	2
			0	720	1140	3
				0	1680	4
					0	5

$M[1,5] = M_1 M_2 M_3 M_4 M_5$

$$M[1,5] = \min \begin{cases} M[1,4] + M[5,5] + P_0 P_4 P_5 \\ = 1080 + 0 + 4 \times 20 \times 7 = 1544 \\ M[1,3] + M[4,5] + P_0 P_3 P_5 \\ = 264 + 1680 + 4 \times 12 \times 7 = 2016 \\ M[1,2] + M[3,5] + P_0 P_2 P_5 \\ = 120 + 1140 + 4 \times 3 \times 7 = 1344 \\ M[1,1] + M[2,5] + P_0 P_1 P_5 \\ = 0 + 1350 + 4 \times 10 + 7 = 1630 \end{cases}$$

so, $M[1,5] = \cancel{13} \; 1344$

	1.	2	3	4	5	
	0	120	264	1080	1344	1
		0	360	1320	1350	2
			0	720	1140	3
				0	1680	4
					0	5

Q8)
C

Ans

The 8-queen problem is a case of more general set of problems namely n queen problem. The basic idea: how to place n queen on n by n board, so that they don't attack each other. The complexity increase as with increasing value of n. There are 92 solutions are rotations of some of the other, while looking for the purists solution there are on 12 distinct solutions.

- start with one queen at the first column first row.
- Continue with second queen from the second column first row
- Go up until find a permissible situation
- Continue with next queen

8-Queens problem:
Algorithm:

The backtracking algorithm, in general checks all possible configurations and test whether the required result is obtained or not for the given problem. we will explore all posible positions the queens can be relatively placed at. The solution will be correct when the number of placed queens = 8..

input format - The number 8, which does not need to be read, but we will take an input number for the sake of generalisation of the algorithm to an N×N chessboard.

Output format: To - all matrices that constitute the possible solutions will contain the numbers 0 and 1.

Visualisation from a 4×4 chessboard solution:

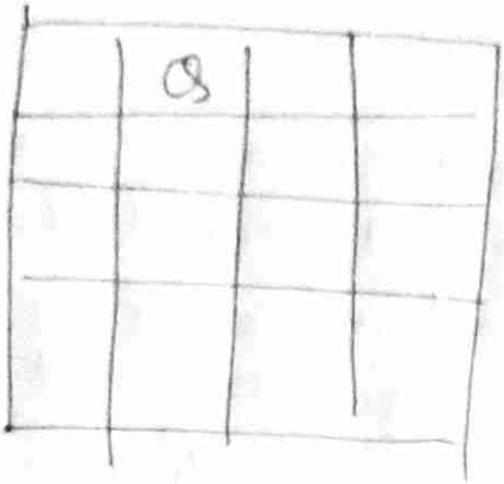
In this configuration, we place 2 queens in the first iteration and see that checking by placing further queens is not required as we will not get a solution in this path.

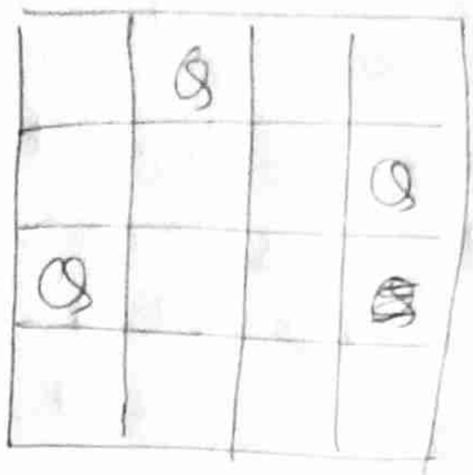
Q			

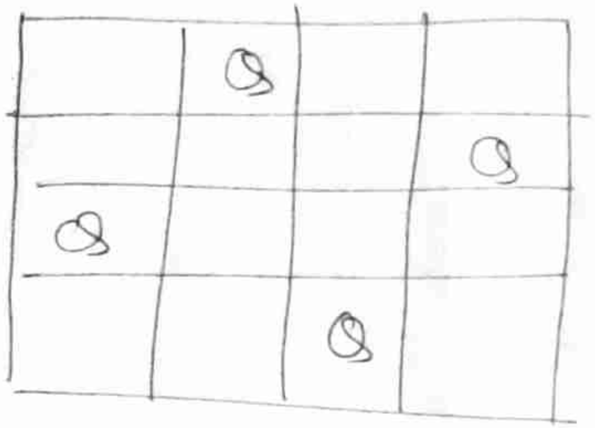
Q			
		Q	

Q			
		Q	
×	×	×	×

As the above combination was not possible, we will go back and go for the next iteration. This means we will change the position of the second queen.







The excepted output is a binary matrix that has 1s for the blocks where queens are placed. for example, the following is the output matrix.

{0, 1, 0, 0}
{0, 0, 0, 1}
{1, 0, 0, 0}

Naive Algorithm:

while there are untried configuration
{
generates the next configuration
if queen don't attack in this configuration then
{
print this configuration;
}
}

Backtracking Algorithm:

1.) ~~stan~~ start in the leftmost column.
2.) if all queens are placed return true.
3.) Try all rows in the current column.
Do following for every tried row
(a) if the queen can be placed safely in this row than mark this as part of the solution and recursively check if placing queen here leads to a solution.

(b) If placing the queen in leads to a solution then return true

(c) If placing queen doesn't lead to a solution then unmark this and go to step (a) try other rows

4.) If all rows have been tried and nothing worked, return false to trigger backtracking.