Date:
P. No:

# Assignment - 03

(Q1) How reliability design can be obtained using Dynamic programming?

Ans In reliability design the problem is to design a system that is composed of several devices connected in series.

$\to \boxed{D_1} \to \boxed{D_2} \to \boxed{D_3} \dashrightarrow \boxed{D_n}$

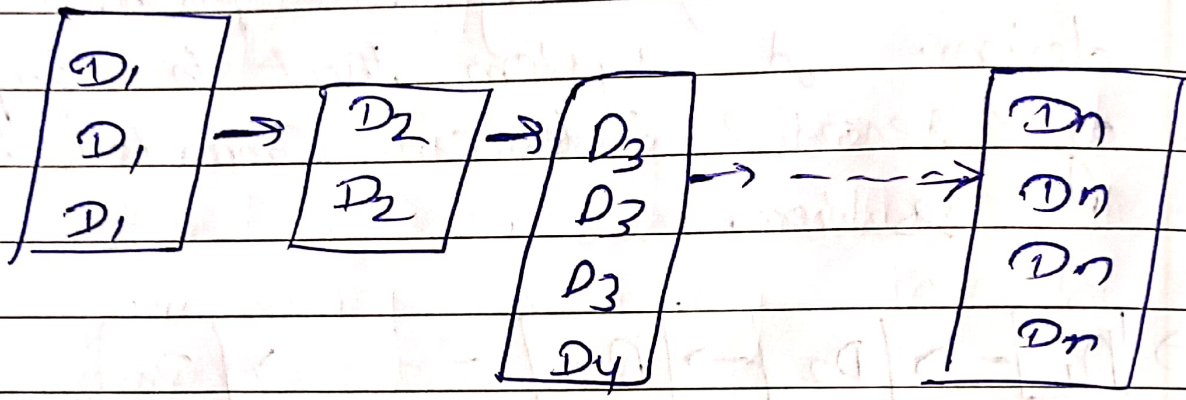
If we imagine $r_i$ as the reliability of a device

Then the reliability of a device of the function can be given as $\pi r_i$.

If $r = 0.99$ and $n = 10$ that n devices are set in a series $1 \le i \le 10$, then reliability of whole system $\pi r_i$ can be given as $\pi r_i = 0.904$.

So if we duplicate the devices at each stage then reliability of the system can be increased.

Say multiple copies of same device type are connected in parallel through the use of switching circuits.




Here switching circuit determines which devices in any given group are functioning properly they the make use of such devices at each stage, that result is increase in reliability at each stages.
If at each stage there are similarly type of device Di, then the probability that all my have male function is $(1-r_i)^{m_i}$, which is very very less

Then the maximization problem can be given as follow.

Maximize $\prod_{1 \le i \le n} \phi_i(m_i)$

subject to $\sum_{1 \le i \le n} c_i m_i \le c$

$m_i \ge 1$ and integer $1 \le i \le n$.

where $\phi(m_i)$ denotes the reliability of stages. The reliability of system can be given $\prod_{1 \le i \le n} \phi_i(m_i)$.

If we increase number of devices at any stages beyond the certain limit then also only cost will increase but reliability could not increase.

—— X —— X ——

Date:
P. No:

Q. Explain the Concepts of Dynamic programming write the different dynamic programming and greedy approach.

As

→ Dynamic programming like the divide and conquer method solved problem by combinning the solutions to supproblem.

As divide and conquer algorithm parition the problem into independent sub-prablem solve the subproblems recursively and then combine their solution to solve the original problem while in contrast dynamic programming is applicable when the subproblem are not independent that is when subproblem share Subproblem in this context, a divide and conquer algorithm does more work the necessary repeatally a solving the common subproblem

Date:
P. No:

* Difference between dynamic programming and Greedy approch?

| | | |
|---|---|---|
| 1) | It first makes as optimal choice without knowing solution to remaing subproblem (s) | It solve subproblem first first, then use those solution to make an optimal choice. |
| 2. | It follow top - down technique. | It follow bottom - up techniques. |
| 3. | It does not use principle of optimality. | It uses principle of optimality. |
| 4. | A greedy method that one can devise an algorithm that works in stages. | Dynamic programming is an algorithm design method that can be used when the solution to a problem of decision. |

Date:
P. No:

Q. find the optimal solution forward knsapack problem $(w_1, w_2, w_3, w_4) = (10, 15, 6, 9)$
$(P_1, P_2, P_3, P_4) = \{2, 5, 8, 1\}$ and $m = 30$.

Sol

$n = 4$

$m = 30$

$(P_1, P_2, P_3, P_4) = \{2, 5, 8, 1\}$

$(w_1, w_2, w_3, w_4) = \{10, 15, 6, 8\}$

$S^0 = \{(0,0)\}$ ; $S_1 = \{(2,10)\}$

$S^1 = \{(0,0)\}$ ; $(2,10)$ ; $S_1' \{(5,15)(7,25)\}$

$S_0^2 = \{(0,0)(2,10)(5,15), (7,25)\}$

$S_1^2 = \{(8,6), (10,16), (13,21), (15,31)\}$

$S^3 = \{(0,0), (2,10), (5,15)\ (7,25)\ (8,6)$
$(10,16), (13,21)\ (15,31)$

using domain once rule we have,

$S^3 = \{(0,0)(8,6)\ (10,16)\ (13,21)(15,31)\}$

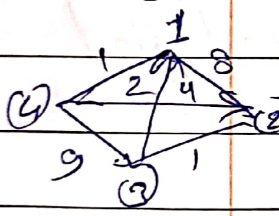
Date:
P. No:

$S_1^3 = \{(1,9)\,(9,15)\,(11,25)\,(14,30)\,(16,40)\}$

$S^4 = \{(0,6)\,(8,6)\,(1,9)\,(9,15)\,(11,25)$
$(14,30)\,(16,40)\}$

—— X —— X ——

Q. Use the floyed - warshall algorithm following graph.




$$D_0 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 8 & \infty & 1 \\ 2 & \infty & 0 & 1 & \infty \\ 3 & 4 & \infty & 0 & \infty \\ 4 & \infty & 2 & 9 & 0 \end{array}$$

$$D_1 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 8 & \infty & 1 \\ 2 & \infty & 0 & 1 & \infty \\ 3 & 4 & 12 & 0 & 5 \\ 4 & \infty & 2 & 9 & 0 \end{array}$$

$$D_2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 8 & 9 & 1 \\ 2 & \infty & 0 & 1 & \infty \\ 3 & 4 & 12 & 0 & 5 \\ 4 & \infty & 2 & 3 & 0 \end{array}$$

Date:
P. No:

$$D_3 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 8 & 9 & 1 \\ 2 & 5 & 0 & 1 & 6 \\ 3 & 4 & 12 & 0 & 5 \\ 4 & 7 & 2 & 3 & 0 \end{array}$$

---

Q4 ⇒ What is multistage Graph Problem Discuss its solution based on dynamic approach also give a suitable algorithm and find its Computing time.

Ans

The multistage graph problem is to find a minimum cost from source to sink i.e. target.

A multistage graph is a directed graph with multiple stages. It has denote a graph $G=(V,E)$ in which the vertices are partitioned into $K \geq 2$ disjoint sets $V_i$ $1 \leq i \leq K$ So that if there is an edge $<u,v>$ from u to V

Date:
P. No:

in E the $u \in V$ and $v \in V_{i+1}$ for some $i$, $1 \le i < k$ and sets $V$ and $V_k$ are such that $|V_1| = |V_k| = 1$

* Sol^n Based on dynamic programming.

A dynamic programming formulation for k-stage graph problem is obtained by first noticing that every source (s) to target (t) path is the result of a sequence of k-2 decise. the with decision takes into Consideration which wertex in $V_{i+1}$, $i \le k-2$ is to is to be on the path.

$\therefore$ $cost(i,j) = \min \{ c(j,l) + cost(i+1, l) \}$
$l \in V_{i+1}$
$(j,l) \in E$ — (1)

* multistage Graph pesudocode Corresponding to the forward approach,

1. algorithm f Graph(G, k, n, P)
2. // the input is a k-stage graph
$G = (V, E)$

```
// indexed in order of stage E is a
     set of edges and
// c[i,j] is the cost of <i,j>, p(i,k)
     is a minimum.
{
  cost [n] := 0.0;
  for j := n-1 to 1 step -1 do
    { // Compute cost [j]
      let r be a vertex such that <j,r> is
         an edge
      of G and c[j,r] + cost [r] is
         minimum
      cost [j] := c[j,r] + cost [r];
      d[j] := r;
    }
  // find a minimum cost path
```

Date:
P. No:

```
P[1] := 1; P[k] := n;
for i := 2 to k-1 do p[i] := d
                       p[i-1]
}
```

23/4/24