Dynamic Programming: meaning, applications and construction

Meaning-

DP determines the optimum solution of a multivariable problem by decomposing it into stages, each stage comprising a single variable sub problem. The advantage of the decomposition is that the optimisation process at each stage involves one variable only, a simple task computationally than dealing with the entire variable simultaneously. A DP model is basically a recursive equation linking the different stages of the problem in a manner that guarantees that each stage's optimal feasible solution is also optimal and feasible for the entire problem.

Recursive nature of computations in DP

Computations in DP are done recursively, so that the optimum solution of one sub problem is used as an input to the next sub problem. By the time, the last sub problem is solved; the optimum solution for the entire problem is at hand. The manner in which the recursive computations are carried out depends on how we decompose the original problem. In particular, the sub problems are normally linked by common constraints. As we move from one sub problem to the next, the feasibility of these common constraints must be maintained.

Forward Recursive equation.

We now show how the recursive computation in above example can be expressed mathematically. Let fi(xi) be the shortest distance to node xi at stage i and define d(xi-1, xi) as the distance from node xi-1 to node xi; then fi is computed from fi-1 using following recursive equation

.fi(xi) = Min { d(xi-1, xi) + fi-1 (xi-1) }

Starting at i=1, the recursion sets f0(x0) =0. The equation shows that the shortest distances fi(xi) at stage i must be expressed in terms of the next node, xi. In the DP terminology, xi is referred to as the state of the system at stage i. In effect, the state of the system at stage i is the information that links the stages together, so that optimal decisions for the remaining stages can be made without re-examining how to decisions for the previous stages are reached. The proper definition of the state allows us to consider each stage separately and guarantee that the solution is feasible for all the stages.

Principle of optimality-

Future decisions for the remaining stages will constitute an optimal policy regardless of the policy adopted in previous stages. (Markovian property)

Forward and Backward recursion

In the example (discussed in class), we uses forward recursion in which the computations proceed from stage 1 to stage 3. The same example can be solved by backward recursion, starting at stage 3 and ending at stage 1.

Both the forward and backward recursions yield the same solution. Although the forward procedure appears more logical, DP literature invariably uses backward recursion. The reason for this preference is

that, in general, backward recursion may be more efficient computationally. Backward recursive equations for example (discussed in class) is

.fi(xi) =min {d(xi, xi+1) + fi+1 (xi+1) }

Where f4(x4) = 0 for x4=7.

Selected applications-

- Cargo-loading model
- Work force size model
- Equipment replacement model
- Investment model
- Inventory model

Cargo Loading Model

Cargo Loading Model (Knapsack/Fly away model)- Suppose W ton vessel can be loaded with one or more n items. The following table gives the unit weight, wi in tonnes and the unit revenue in thousands of dollars, ri.

Item i	.wi	.ri
1	.w1	.r1
2	.w2	.r2
.	.	.
.	.	.
N	.wn	.rn

Problem-how should the vessel be loaded to maximum the total return ?

Let mi = number of units of item i in the vessel. The general problem is represented by the following ILP (ILP are linear programs with some or all the variable restricted to integer (or discrete) values.

Maximise z= r1m1 + r2m2 + ... + rnmn

Subject to

.w1m1 +w2m2 + ... +wnmn $\leq$ W

.m1, m2, ... , mn $\geq$ 0 and integer

The three elements of model are

- Stage i is represented by item i, i=1, 2, 3...., n
- The alternatives at stage i are represented by mi, the number of units of item i included in the knapsack. The associated return is rimi. Defining [W/wi] as the largest integer less than or equal to W/wi, it follows that mi = 0, 1, 2, ..., W/wi

- The state at state i is represented by xi, the total weight assigned to stages (items) i, i+1, ..., n . This definition reflects the fact that the weigth constraint is the only restriction that links all stages together.
  Define fi(xi) = Maximum return for stages i, i+1, and n, given state xi.