Modelling Transportation Systems

Deterministic Dynamic Programming

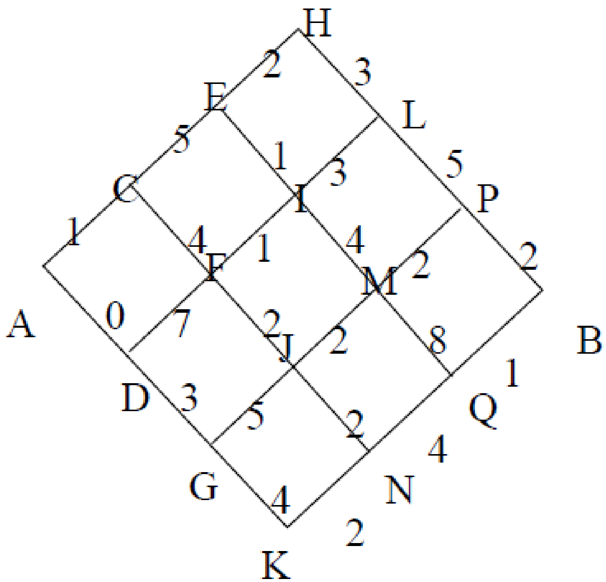
Dr Zhiyuan Liu
Email: zhiyuanl@seu.edu.cn

Learning Objectives

- **Part I: Fundamentals**
  - Taken through some simple path problems to understand the basic features of Dynamic Programming (DP)
  - Learn how to formulate DP
  - Understand the process of numerically solving a DP
- **Part II: Applications**
  - Understand the use of DP in various application areas
  - Know how to formulate DPs in application areas
  - Know how to numerically solve these DPs and interpret the solution for management issues

Elements of a DP model

- **Definitions**
- DP is an optimization procedure that is particularly suited to problems requiring a sequence of interrelated decisions.
- The goal of DP is to determine a sequence of decisions which in turn yields a sequence of situations, that maximizes (or minimizes) some objective function.



Example: Determine the shortest-time path from A to B

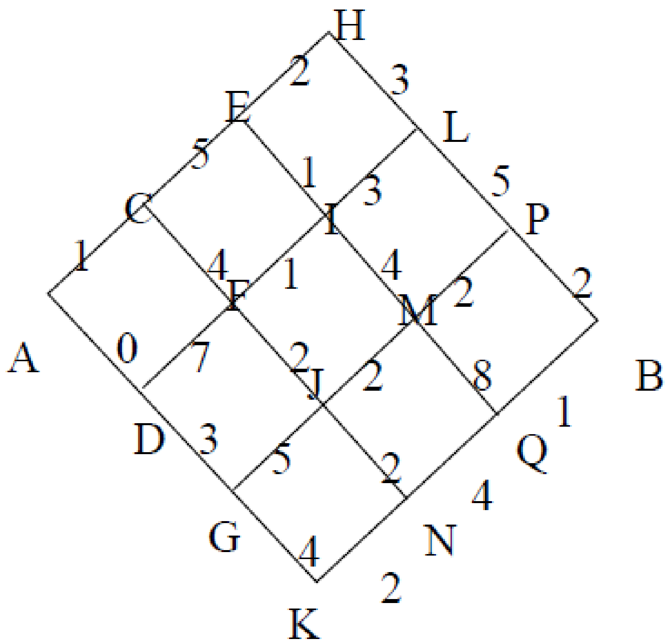
Solution: Brute force

20 paths * 5 additions
+ 19 comparisons

DP offers a more efficient method

Elements of a DP model

- **Could I say this?**
- Shortest path from A to B depends on the values for
  - Shortest path from C to B (S_C)
  - Shortest path from D to B (S_D)
- Knowing S_C and S_D, the decision can be made



Notes:

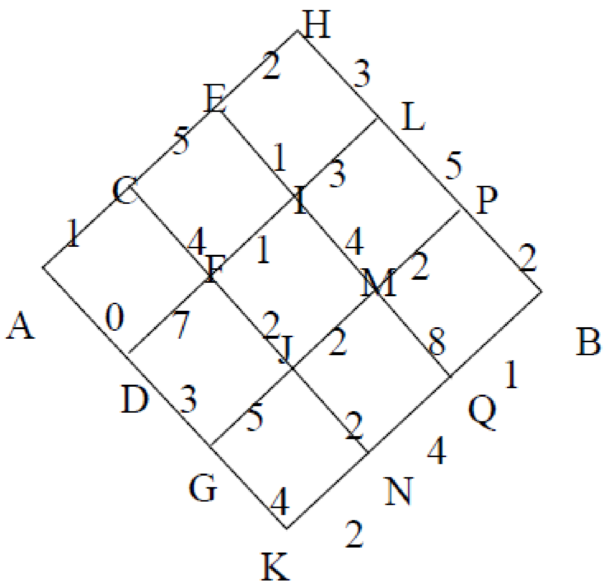
S_C in turn depends on S_E and S_F

S_D depends on S_F and S_G

- At each node we only need to know the best paths **ahead**.
- All inferior paths are irrelevant, which is **the principle of optimality.**

Elements of a DP model

- **Principle of Optimality**
- The best path from A to B has the following property: whatever the initial decision at A, the remaining path to B, starting from the next point after A, must be the best path from that point to B.

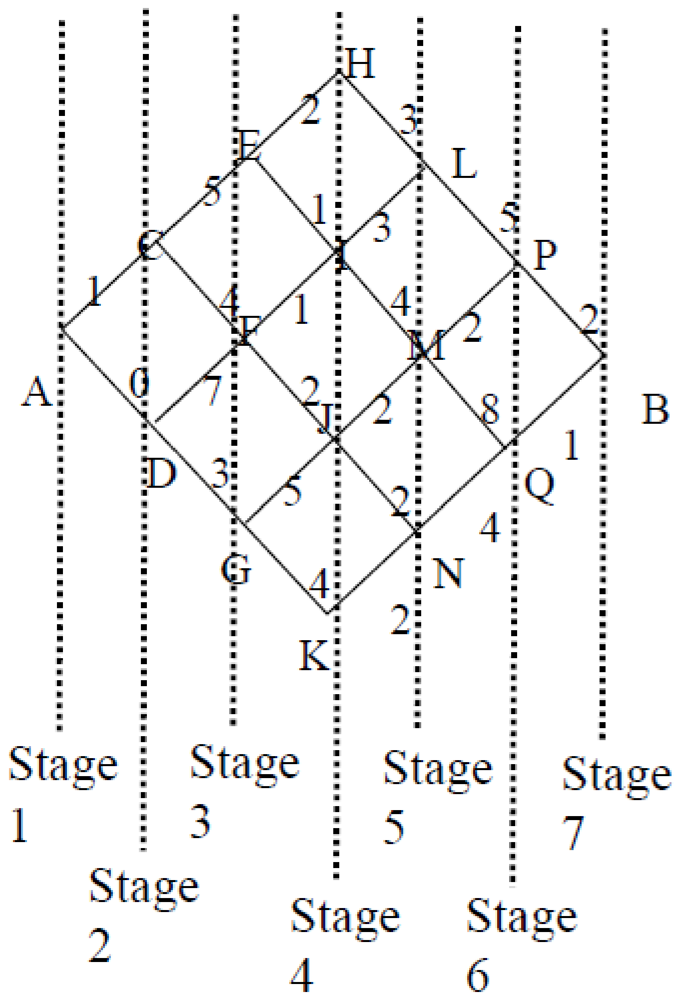


$$S_A = \min \begin{bmatrix} 1 + S_C \\ 0 + S_D \end{bmatrix}$$

$$S_C = \min \begin{bmatrix} 5 + S_E \\ 4 + S_F \end{bmatrix} \quad S_D = \min \begin{bmatrix} 7 + S_F \\ 3 + S_G \end{bmatrix}$$

- Note that there is a **recursive set** of relations so that if we know S_P and S_Q, we can begin to work back to get S_A.

Elements of a DP model



- Each stage consists of several states
- For any given stage, we can make a decision at a state (node) in going upward or downward, in order to find the shortest path from this node to node B.
- Whatever decision we choose at the current state, the remaining decisions must constitute the optimal path to node B.

Elements of a DP model

- **Richard Bellman’s (1953) Principle of Optimality**

An optimal policy has the property that whatever the initial state and the initial decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.

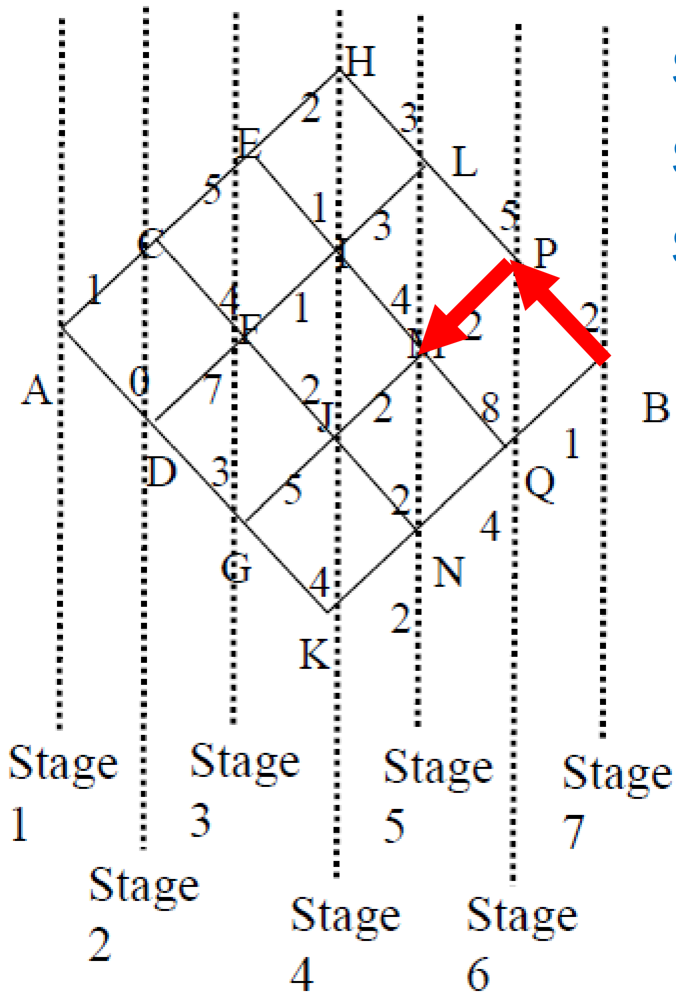
- **Rutherford Aris’ the restatement**

If you don't do the best with what you have happened to have got, you will never do the best with what you should have had.

- **Another equivalent definition**

From any point on an optimal trajectory, the remaining trajectory is optimal for the corresponding problem initiated at that point.

Elements of a DP model



Stage 7: $S_B = 0 \quad \text{(terminal value)}$

Stage 6: $S_P = 2 \quad S_Q = 1$

Stage 5: $S_L = 5 + S_P = 7(P)$

$$S_M = \min \begin{bmatrix} 2 + S_P \\ 8 + S_Q \end{bmatrix} = 4(P)$$

$$S_N = 4 + S_Q = 5(Q)$$

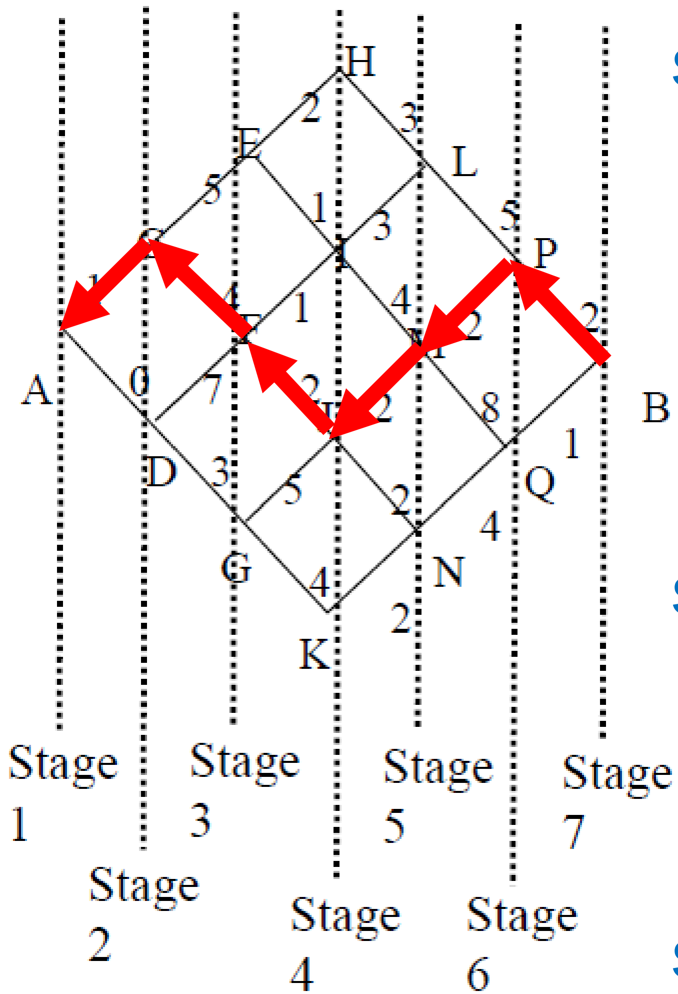
Stage 4: $S_H = 3 + S_L = 10(L)$

$$S_I = \min \begin{bmatrix} 3 + S_L \\ 4 + S_M \end{bmatrix} = 8(M)$$

$$S_J = \min \begin{bmatrix} 2 + S_M \\ 2 + S_N \end{bmatrix} = 6(M)$$

$$S_K = 2 + S_N = 7(N)$$

Elements of a DP model



Stage 3:

$$S_E = \min\begin{bmatrix} 2+S_H \\ 1+S_I \end{bmatrix} = 9(I)$$

$$S_F = \min\begin{bmatrix} 1+S_I \\ 2+S_J \end{bmatrix} = 8(J)$$

$$S_G = \min\begin{bmatrix} 5+S_J \\ 4+S_K \end{bmatrix} = 11(J, K)$$

Stage 2:

$$S_C = \min\begin{bmatrix} 5+S_E \\ 4+S_F \end{bmatrix} = 12(F)$$

$$S_D = \min\begin{bmatrix} 7+S_F \\ 3+S_G \end{bmatrix} = 14(G)$$

Stage 1:

$$S_A = \min\begin{bmatrix} 1+S_C \\ 0+S_D \end{bmatrix} = 13(C)$$

Elements of a DP model

❑Basic features of DP

- The problem can be divided into ***stages*** with a ***decision*** required at each stage.

The stages were defined by the structure of the graph. The decision was to go next.

- Each stage has a number of ***states*** associated with the beginning of that stage.

The states was the node reached.

- The effect of the decision at each stage is to transform the current state to a state associated with the beginning of the next stage.

The decision of where to go next defined where you arrived in the next stage.

Elements of a DP model

❑Basic features of DP

- The solution procedure is designed to find an optimal policy for the overall problem.
- Given the current state, the ***optimal decision*** for the remaining states is independent of the policy decisions adopted in previous stages. Therefore, the optimal immediate decision depends on only the current state and not on how you got there.

This is the ***principle of optimality*** for the DP.

- The solution procedure begins by finding the optimal policy for the last stage

The optimal policy for the last stage prescribes the optimal policy decision for each of the possible states at that stage.

Elements of a DP model

Basic features of DP

- ***A recursive relationship*** identifies the optimal ***decision*** for stage *j*, given that stage *j+1* has already been solved.

- When we use this recursive relationship, the solution procedure starts at the end and moves backward stage by stage — each time finding the optimal policy for that stage — until it finds the optimal solution for the entire problem.

Elements of a DP model

- **Key ideas**
  - The whole problem can be solved if the values of the best solutions of certain subproblems can be determined using the idea of the principle of optimality.
  - The end or near end of the whole problem, the subproblems are so simple as to have trivial solutions.
- **Optimal value function**
  - Rule that assigns values to subproblems
- **Arguments of the function**
  - Each argument refers to a particular subproblem
  - Denoted by stage and state

Elements of a DP model

- **Optimal policy function**
  - Rule that associates best decision with each subproblem.
- **Recurrence Relations**
  - Set of formulae relating various objective values derived from principle of optimality.
- **Boundary conditions**
  - Value of optimal value function for certain arguments assumed obvious from statement of problem and from definition of optimal value function with no computation required.

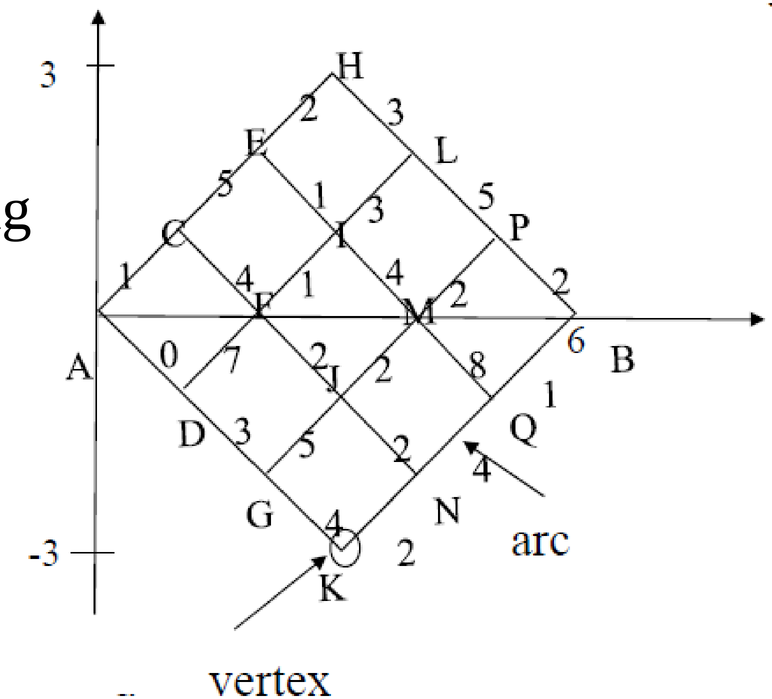
Elements of a DP model

- **Procedure**
  - Design the ***arguments*** (or devise stages and states)
  - Define the ***optimal value function*** to allow the use of the principle of optimality
  - Write the ***recurrence relations***
  - Starting with ***boundary conditions***, use recurrence relations to determine optimal value and policy functions at every argument
  - Best path decisions can be traced out using optimal policy function

Systematic representation of simple path problem

- **Optimal value function**

$$S(x,y) = \text{value of min-time path connecting vertex } (x,y) \text{ with terminal vertex } (6,0) \text{ or B}$$



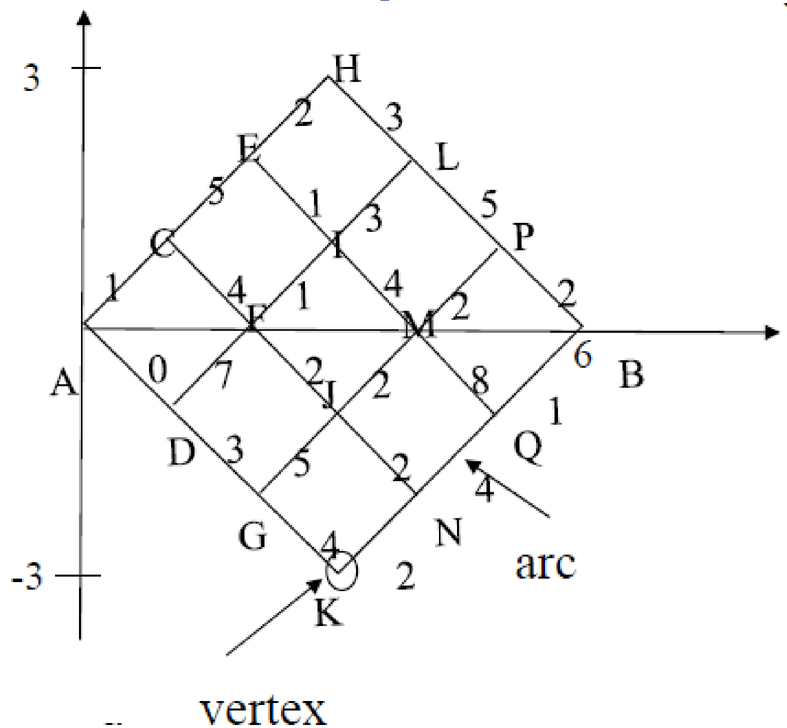
- **Recurrence relations**

$$S(x,y) = \min \begin{bmatrix} a_u(x,y) + S(x+1, y+1) \\ a_d(x,y) + S(x+1, y-1) \end{bmatrix}$$

Systematic representation of simple path problem

- **Boundary conditions**

$$S(6,0)=0 \quad \text{or} \quad \begin{aligned} S(5,1)&=2 \\ S(5,-1)&=1 \end{aligned}$$



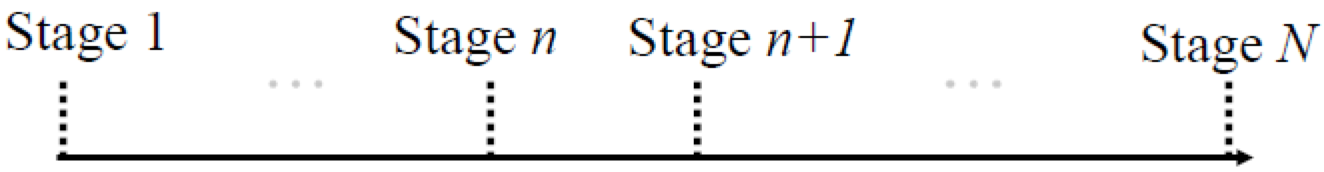
- **Define arcs**

$$a_u(x,y) = \text{cost of arc connecting } (x,y) \text{ to } (x+1, y+1)$$

$$a_d(x,y) = \text{cost of arc connecting } (x,y) \text{ to } (x+1, y-1)$$

$$a_u(x,y) = \infty, \text{denotes no such arc}$$

Forward DP



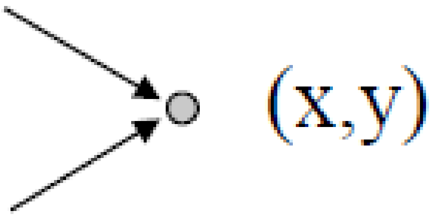
- The recurrence relations for the previous example were written for the ***backward*** procedure.
- It answers the question: **Which is the best route from *(x,y)* to the terminal?**
- Entire problem is solved when *(x,y)*=***origin***
- Another formulation can be obtained by answering the question: **Which is best route from origin to *(x,y)*?**
- Entire problem is solved when *(x,y)* = ***terminal***

Forward DP

- Entire problem is solved when *(x,y)* = ***terminal***
- **Recurrence relations**

$$S(x,y) = \min \begin{bmatrix} a_u(x-1,y-1) + S(x-1,y-1) \\ a_d(x-1,y+1) + S(x-1,y+1) \end{bmatrix}$$

- **Boundary conditions**

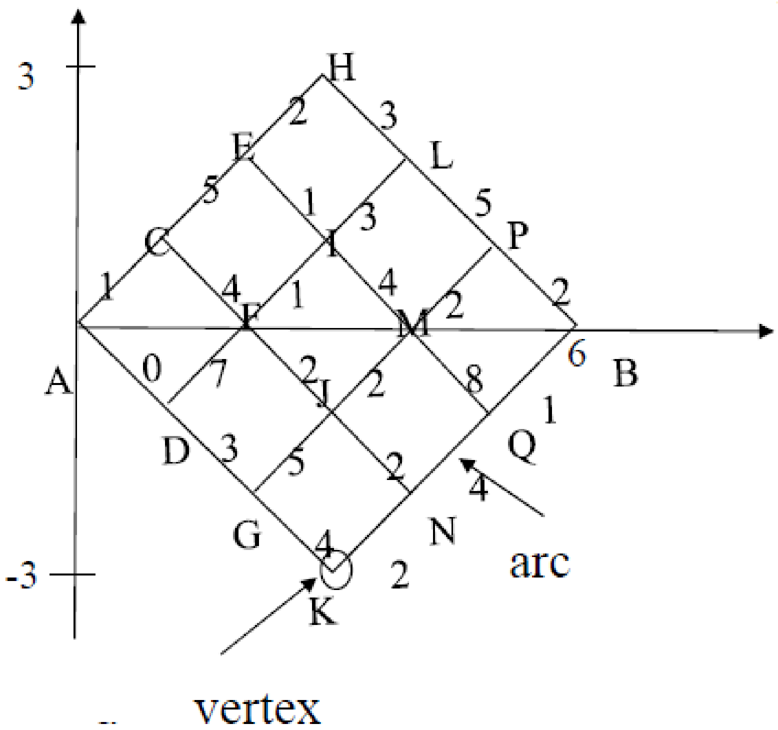


$$S(0,0) = 0$$

- **The results of backward and forward procedure will be the same.**

A more complicated example

- Following the same network of paths, we now include a twist to the problem. **An additional time of 3** is incurred with every turn at the vertices.



Definitely, more information is required to determine the objective value function

- Stage variables?
- State variables?
  - ✓ Vertex
  - ✓ Is there a turn?

This can be ascertained if for backward procedure we know in which direction it leaves vertex *(x,y)* as it proceeds towards B.

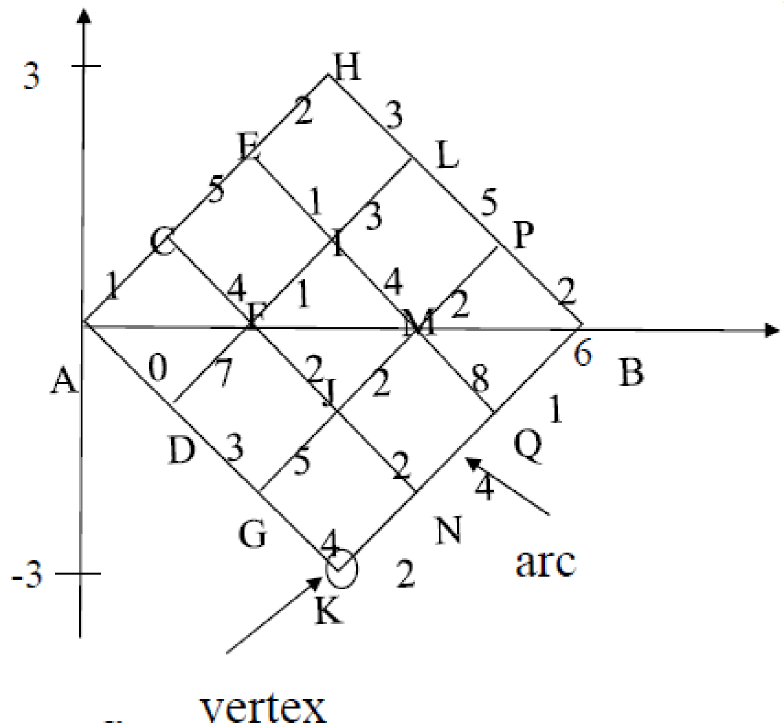
A more complicated example

- **Arguments:**
  - *S(x,y,z)* where *x* and *y* denote the vertex and *z* denotes the direction taken when leaving the vertex, and

$$z = 0, \text{upwards}$$
$$z = 1, \text{downwards}$$

- **Objective value function:**

$$S(x, y, z) = \text{value of min-time taken if start at vertex } (x, y) \text{ going to B, and move initially in the direction indicated by } z$$



A more complicated example

- **Recurrence relations**

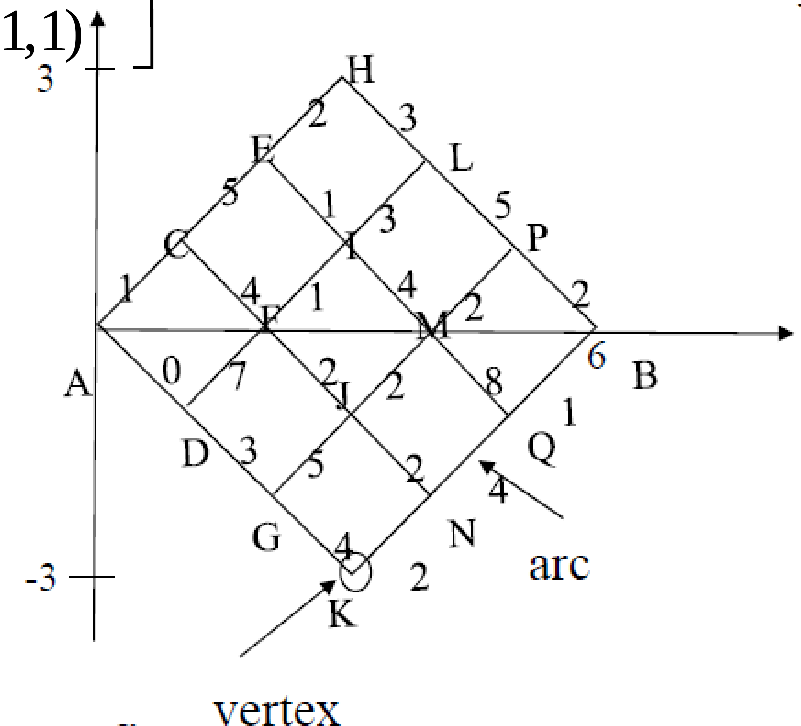
$$S(x,y,0) = a_u(x,y) + \min\begin{bmatrix} S(x+1,y+1,0) \\ S(x+1,y+1,1)+3 \end{bmatrix}$$

$$S(x,y,1) = a_d(x,y) + \min\begin{bmatrix} S(x+1,y-1,0)+3 \\ S(x+1,y-1,1) \end{bmatrix}$$

- **Objective value function:**

$$S(6,0,0) = 0$$

$$S(6,0,1) = 0$$



A more complicated example

- **Solution**
- **Stage 6:**

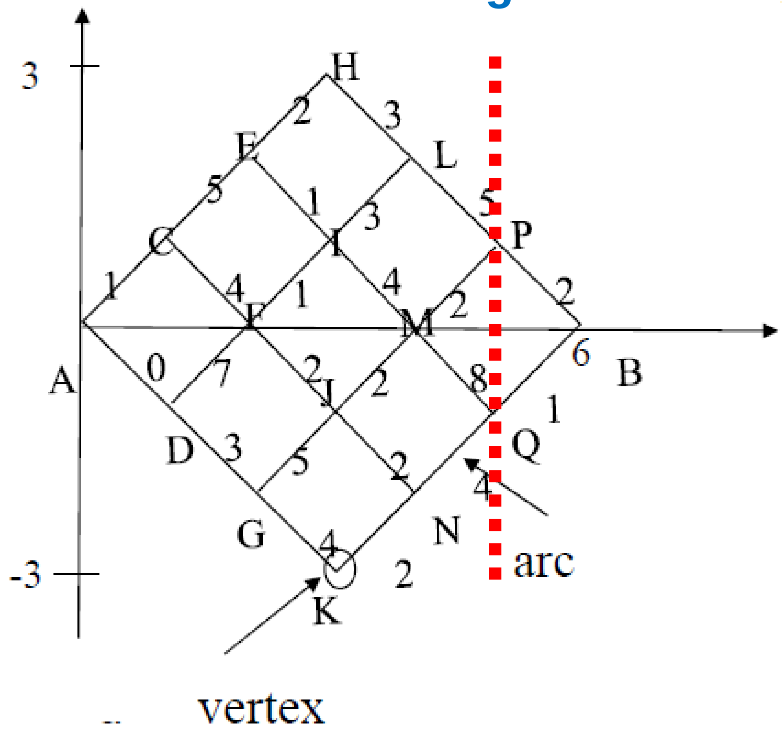
$$S(5,1,0)=\infty+\min\begin{bmatrix}\infty\\ \infty+3\end{bmatrix}=\infty$$

$$S(5,1,1)=2+\min\begin{bmatrix}3\\ 0\end{bmatrix}=2$$

$$S(5,-1,1)=1+\min\begin{bmatrix}3\\ 0\end{bmatrix}=1$$

$$S(5,-1,1)=\infty+\min\begin{bmatrix}\infty+3\\ \infty\end{bmatrix}=\infty$$

Stage 6



A more complicated example

Stage 5

- **Solution**
- **Stage 5:**

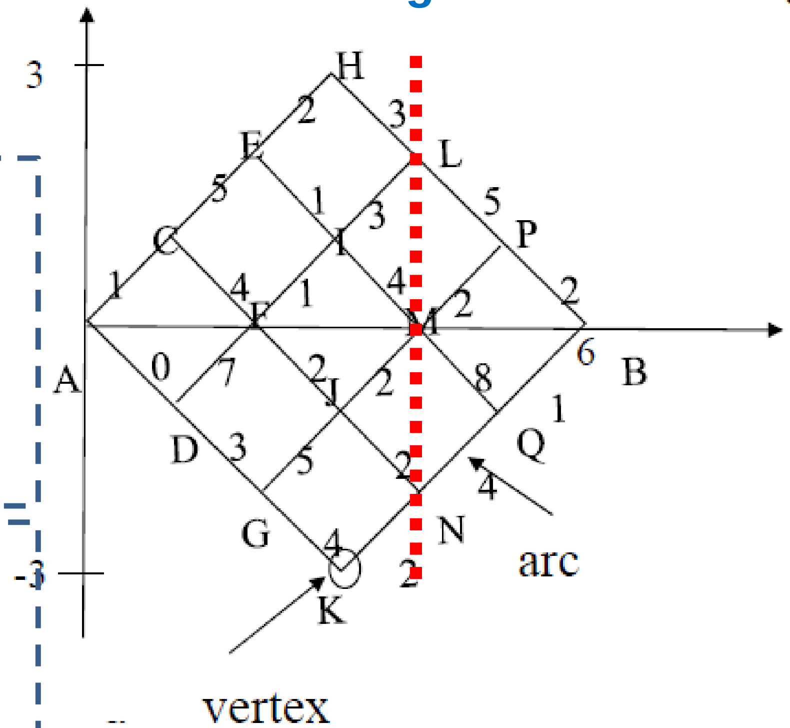
$$S(4,2,0)=\infty+\min\begin{bmatrix}\infty\\3+\infty\end{bmatrix}=\infty$$

$$S(4,2,1)=5+\min\begin{bmatrix}S(5,1,0)+3\\S(5,1,1)\end{bmatrix}=7(D)$$

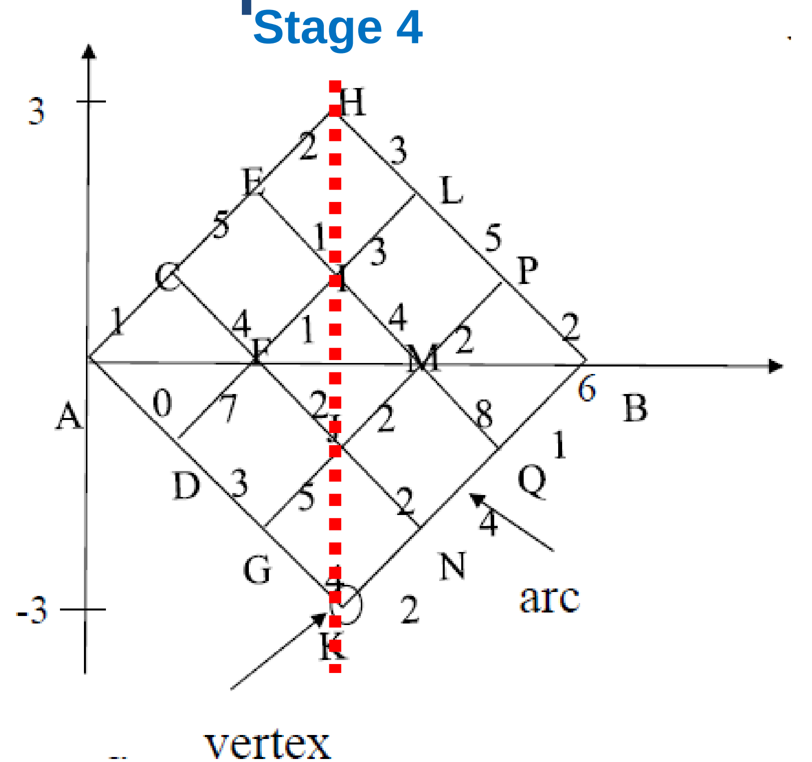
$$S(4,0,0)=2+\min\begin{bmatrix}S(5,1,0)\\S(5,1,1)+3\end{bmatrix}=7(D)$$

$$S(4,0,1)=8+\min\begin{bmatrix}S(5,-1,0)+3\\S(5,-1,1)\end{bmatrix}=12(U)$$

$$S(4,-2,0)=4+\min\begin{bmatrix}S(5,-1,0)\\S(5,-1,1)+3\end{bmatrix}=5(U) \qquad S(4,-2,1)=\infty$$



A more complicated example



- **Solution**
- **Stage 4:**

$$S(3,3,0) = \infty$$

$$S(3,3,1) = 3 + \min\begin{bmatrix} \infty + 3 \\ S(4,2,1) \end{bmatrix} = 10(D)$$

$$S(3,1,0) = 3 + \min\begin{bmatrix} \infty \\ S(4,2,1) + 3 \end{bmatrix} = 13(D)$$

$$S(3,1,1) = 4 + \min\begin{bmatrix} S(4,0,0) + 3 \\ S(4,0,1) \end{bmatrix} = 14(U)$$

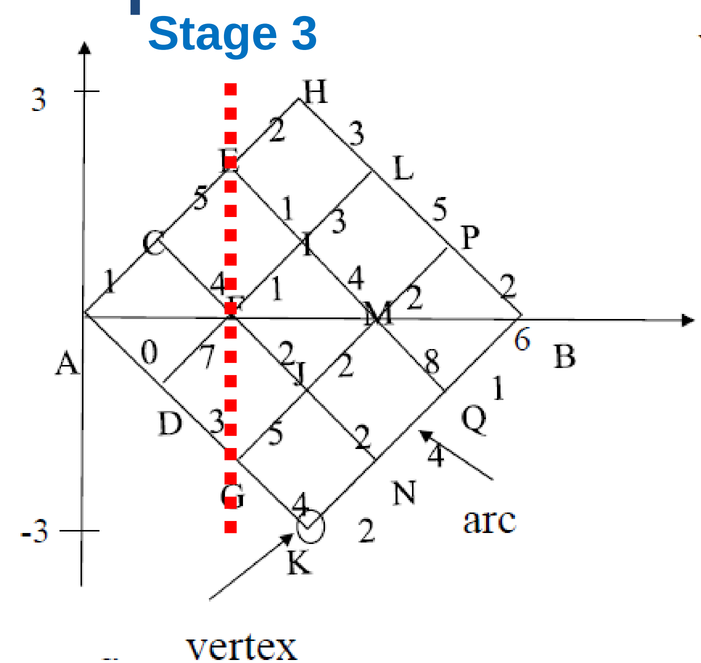
$$S(3,-1,0) = 2 + \min\begin{bmatrix} S(4,0,0) \\ S(4,0,1) + 3 \end{bmatrix} = 9(U)$$

$$S(3,-1,1) = 2 + \min\begin{bmatrix} S(4,-2,0) + 3 \\ S(4,-2,1) \end{bmatrix} = 10(U)$$

$$S(3,-3,0) = 2 + \min\begin{bmatrix} S(4,-2,0) \\ S(4,-2,1) \end{bmatrix} = 7(U)$$

$$S(3,-3,1) = \infty$$

A more complicated example



- **Solution**
- **Stage 3:**

$$S(2,2,0) = 2 + \min \begin{bmatrix} \infty \\ S(3,3,1)+3 \end{bmatrix} = 15(D)$$

$$S(2,2,1) = 1 + \min \begin{bmatrix} S(3,1,0)+3 \\ S(3,1,1) \end{bmatrix} = 15(D)$$

$$S(2,0,0) = 1 + \min \begin{bmatrix} S(3,1,0) \\ S(3,1,1)+3 \end{bmatrix} = 14(U)$$

$$S(2,0,1) = 2 + \min \begin{bmatrix} S(3,-1,0)+3 \\ S(3,-1,1) \end{bmatrix} = 12(D)$$

$$S(2,-2,0) = 5 + \min \begin{bmatrix} S(3,-1,0) \\ S(3,-1,1)+3 \end{bmatrix}$$

$$= 14(U)$$

$$S(2,-2,1) = 4 + \min \begin{bmatrix} S(3,-3,0)+3 \\ S(3,-3,1) \end{bmatrix}$$

$$= 14(U)$$

A more complicated example

- **Solution**
- **Stage 2:**

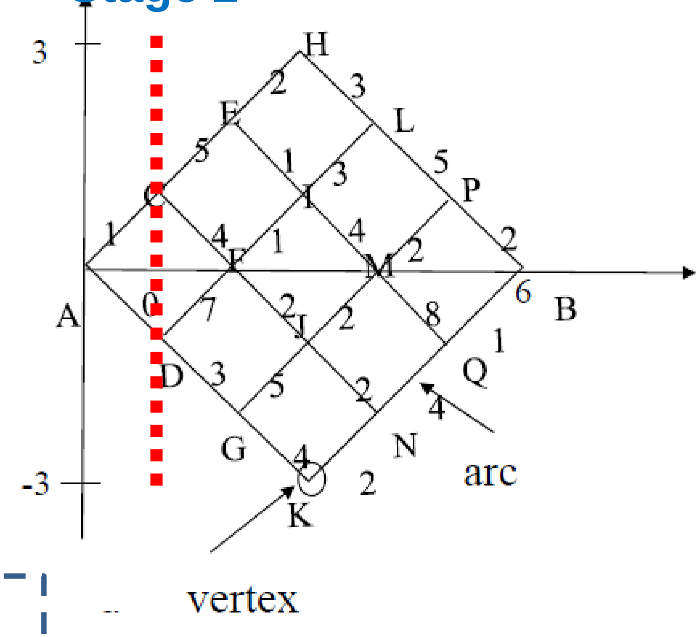
$$S(1,1,0)=5+\min\begin{bmatrix} S(2,2,0) \\ S(2,2,1)+3 \end{bmatrix}=20(U)$$

$$S(1,1,1)=4+\min\begin{bmatrix} S(2,0,0)+3 \\ S(2,0,1) \end{bmatrix}=16(D)$$

$$S(1,-1,0)=7+\min\begin{bmatrix} S(2,0,0) \\ S(2,0,1)+3 \end{bmatrix}=21(U)$$

$$S(1,-1,1)=3+\min\begin{bmatrix} S(2,-2,0)+3 \\ S(2,-2,1) \end{bmatrix}=17(D)$$

Stage 2

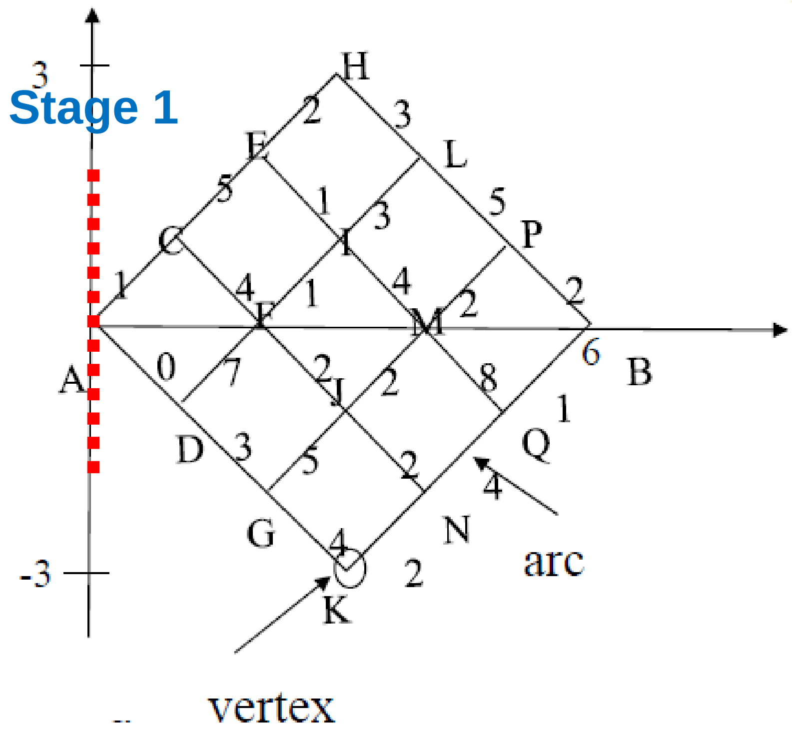


A more complicated example

- **Solution**
  - **Stage 1:**

$$S(0,0,0)=1+\min\begin{bmatrix} S(1,1,0) \\ S(1,1,1)+3 \end{bmatrix}=20(D)$$

$$S(0,0,1)=0+\min\begin{bmatrix} S(1,-1,0)+3 \\ S(1,-1,1) \end{bmatrix}=17(D)$$



Summary

- **Basic idea**
  - Decompose an optimization problem into a series of interrelated sub-problems
- **Principle**
  - Bellman's principle of optimality
- **Implementation**
  - Define an optimal value function
    - Stage variable, Stage variables
  - Write the appropriate recurrence (recursive) relation
  - Optimal police function
  - Boundary conditions
- **It is an art to determine the stage variables, state variables and the resultant optimal value function**

Part II: Applications

Learning Objectives

- **Part II: Applications**
  - Understand the use of DP in various application areas
    - Traveling Salesman Problem (TSP)
    - Equipment replacement problems
    - Resource allocation problems
  - Know how to formulate DPs in application areas
  - Know how to numerically solve these DPs and interpret the solution for management issues

Bellman's Optimality Principle

- **An optimal decision process (or path) has the property:**
  - (1) Whatever the initial conditions and control variables (choices or decisions) ***over some initial period***;
  - (2) The control (or decision variables) chosen over the remaining period must be optimal for ***the remaining problem***, ***with the state resulting*** from the early decisions taken to be the initial condition.

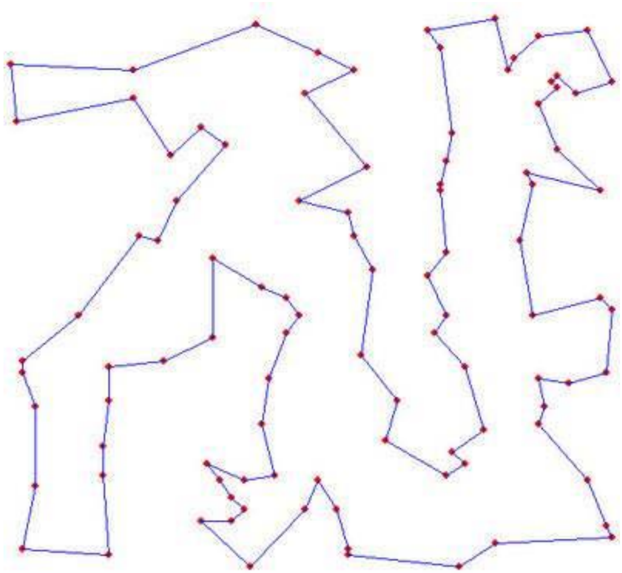
DP formulation in the TSP

❑Problem description

We have a general network consisting of N cities (or nodes). The cities are arbitrarily numbered from 1 to N, and d_{ij} is the distance (or travel time, or cost, etc.) from city i to city j. A salesman would like to begin at city 1, visit each of the other cities once and only once, and then return to city 1 (such a path is called a *tour*).

❑Question

In what order should he visit the cities so that the total distance he travels is minimized?



DP formulation in the TSP

- **DP formulation**

Let $N_j = \{2, 3, \ldots, j, \ldots, N\}$ and let S be a subset of N_j containing *i* members.

- **Optimal value function:**

$$f_i(j, S) = \text{the length of the shortest path from 1 to city } j \text{ via the set of } i \text{ intermediate cities } S \text{ (the stage variable } i \text{ indicates the number of cities in } S) \quad (1)$$

- **Recurrence relation:**

$$f_i(j, S) = \min_{k \in S}[f_{i-1}(k, S - \{k\}) + d_{kj}],\ i = 1, 2, \ldots, N-2,\ j \neq 1, S \subseteq N_j \quad (2)$$

- **Boundary condition:** $f_0(j, -) = d_{1j}$ (3)
- **Answer:**

$$\min_{j=2,3,\ldots,N} [f_{N-2}(j, N_j) + d_{j1}] \quad (4)$$

DP formulation in the TSP

- **Justification of the recurrence relation**

$$f_i(j,S) = \min_{k \in S}[f_{i-1}(k, S-\{k\}) + d_{kj}],\ i = 1,2,\ldots,N-2,\ j \neq 1, S \subseteq N_j \qquad (2)$$

- Suppose that we would like to compute $f_i(j,S)$.
- Consider the shortest path from city 1 to city *j* via *S* which has city *k* (a member of *S*) immediately preceding city *j*.
- Since the cities in $S-\{k\}$ must be visited in an optimal order, the length of this path is

$$f_{i-1}(k, S-\{k\}) + d_{kj}$$

- Furthermore, since we are free to choose city *k* optimally, it is clear that $f_i(j,S)$ is correctly given by Eq. (2).

DP formulation in the TSP

- **Justification of the recurrence relation**

$$f_i(j,S) = \min_{k \in S}[\, f_{i-1}(k, S-\{k\}) + d_{kj}\,],\ i = 1, 2, \ldots, N-2,\ j \neq 1, S \subseteq N_j \qquad (2)$$

- To compute the length of the shortest tour, we begin by computing $f_1(j,S)$ (for every *j, S* pair) from the values of f_0 .
- We then compute $f_2(j,S)$ from the values of f_1 .

  …..

- Until $f_{N-2}(j,N_j)$ has been computed for every city *j*.
- Then length of the shortest tour is given by

$$\min_{j=2,3,\ldots,N} [\, f_{N-2}(j,N_j) + d_{j1}\,] \qquad (4)$$

since some city, *j*, must be the last city visited before returning to city 1.

A TSP example

- Given a distance matrix for a network consisting of 5 cities.

(the optimal policy function is given in parentheses)

i \ j	1	2	3	4	5
1	0	3	1	5	4
2	1	0	5	4	3
3	5	4	0	2	1
4	3	1	3	0	3
5	5	2	4	1	0

- **Stage 0:**

$$f_0(2,-) = d_{12} = 3(1)$$

$$f_0(3,-) = d_{13} = 1(1)$$

$$f_0(4,-) = d_{14} = 5(1)$$

$$f_0(5,-) = d_{15} = 4(1)$$

- **Stage 1:**

Why d_{32}? Not d_{23}?

$$f_1(2,\{3\}) = f_0(3,-) + \boxed{d_{32}} = 1 + 4 = 5(3)$$

$$f_1(2,\{4\}) = 5 + 1 = 6(4)$$

$$f_1(2,\{5\}) = 4 + 2 = 6(5)$$

$$f_1(3,\{2\}) = 3 + 5 = 8(2)$$

$$f_1(3,\{4\}) = 5 + 3 = 8(4)$$

$$f_1(3,\{5\}) = 4 + 4 = 8(5)$$

$$f_1(4,\{2\}) = 3 + 4 = 7(2)$$

$$f_1(4,\{3\}) = 1 + 2 = 3(3)$$

$$f_1(4,\{5\}) = 4 + 1 = 5(5)$$

$$f_1(5,\{2\}) = 3 + 3 = 6(2)$$

$$f_1(5,\{3\}) = 1 + 1 = 2(3)$$

$$f_1(5,\{4\}) = 5 + 3 = 8(4)$$

A TSP example

$$1 \to 3 \to 4 \to 2 \ or$$
$$1 \to 4 \to 3 \to 2$$

- **Stage 2:**

$$f_2(2,\{3,4\}) = \min[f_1(3,\{4\}) + d_{32},\ f_1(4,\{3\}) + d_{42}]$$
$$= \min[8+4, 3+1] = 4(4)$$
$$f_2(2,\{3,5\}) = \min[8+4, 2+2] = 4(5)$$
$$f_2(2,\{4,5\}) = \min[5+1, 8+2] = 6(4)$$
$$f_2(3,\{2,4\}) = \min[6+5, 7+3] = 10(4)$$
$$f_2(3,\{2,5\}) = \min[6+5, 6+4] = 10(5)$$
$$f_2(3,\{4,5\}) = \min[5+3, 8+4] = 8(4)$$
$$f_2(4,\{2,3\}) = \min[5+4, 8+2] = 9(2)$$
$$f_2(4,\{2,5\}) = \min[6+4, 6+1] = 7(5)$$
$$f_2(4,\{3,5\}) = \min[8+2, 2+1] = 3(5)$$

i \ j	1	2	3	4	5
1	0	3	1	5	4
2	1	0	5	4	3
3	5	4	0	2	1
4	3	1	3	0	3
5	5	2	4	1	0

$$f_2(5,\{2,3\}) = \min[5+3, 8+1] = 8(2)$$
$$f_2(5,\{2,4\}) = \min[6+3, 7+3] = 9(2)$$
$$f_2(5,\{3,4\}) = \min[8+1, 3+3] = 6(4)$$

A TSP example

$$1 \to 3 \to 4 \to 5 \to 2 \ \mathit{or}\ 1 \to 3 \to 5 \to 4 \to 2$$
$$1 \to 4 \to 3 \to 5 \to 2 \ \mathit{or}\ 1 \to 4 \to 5 \to 3 \to 2$$
$$1 \to 5 \to 3 \to 4 \to 2 \ \mathit{or}\ 1 \to 5 \to 4 \to 3 \to 2$$

- **Stage 2:**

$$f_3(2,\{3,4,5\}) = \min[f_2(3,\{4,5\}) + d_{32},\ f_2(4,\{3,5\}) + d_{42},\ f_2(5,\{3,4\}) + d_{52}]$$
$$= \min[8+4, 3+1, 6+2] = 4(4)$$

$$f_3(2,\{3,4,5\}) = \min[6+5, 7+3, 9+4] = 10(4)$$

$$f_3(4,\{2,3,5\}) = \min[4+4, 10+2, 8+1] = 8(2)$$

$$f_3(5,\{2,3,4\}) = \min[4+3, 10+1, 9+3] = 7(2)$$

i \ j	1	2	3	4	5
1	0	3	1	5	4
2	1	0	5	4	3
3	5	4	0	2	1
4	3	1	3	0	3
5	5	2	4	1	0

- **Answer:** $\min\limits_{j=2,3,4,5} [f_3(j,\{2,3,4,5\} - \{j\}) + d_{j1}]$

$$= \min[4+1, 10+5, 8+3, 7+5] = 5(2)$$

Therefore, the length of the shortest tour is 5, and the corresponding shortest tour is $1 \to 3 \to 5 \to 4 \to 2 \to 1$

Equipment replacement problem

- **Problem description**
- This problem concerns a type of machine which deteriorates with age, and the decision to replace it. We assume that we must own such a machine during each of the next *N* time periods (called years in what follows, although the same ideas solve problems with monthly, daily, etc. decisions) and that the cost of operating a machine for one year is a known quantity and depends on the age of the machine. Whenever a machine gets intolerably old, we may replace it, paying a known amount for a new machine and receiving in return a given amount, which depends upon the age of our old machine, as a trade-in value. When the process ends after *N* years, we receive a salvage value which depends upon the age of our old machine.

Equipment replacement problem

- **Problem description**
- We must be told the duration *N* of the process, the age *y* of the machine with which we start the process, and the following data:

$$c(i) = \text{cost of operating for one year a machine is of age } i \text{ at the start of the year,}$$

$$p = \text{price of a new machine (of age 0),}$$

$$t(i) = \text{trade-in value received when a machine which is of age } i \text{ at the start of a year is traded for a new machine at the start of the year,}$$

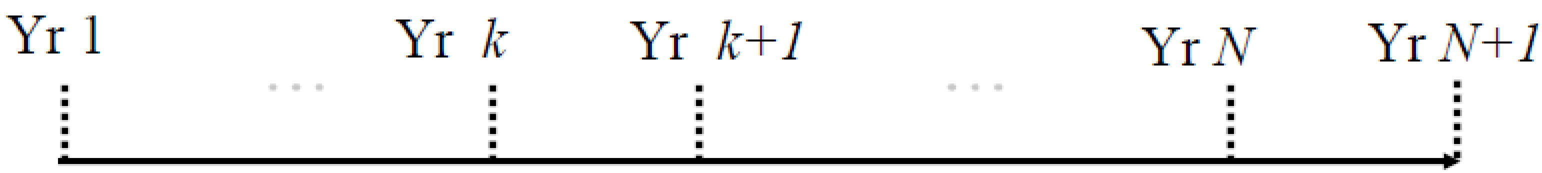
$$s(i) = \text{salvage value received for a machine that has just turned age } i \text{ at the end of year } N.$$

Equipment replacement problem

- **Question**
- The problem is to decide
  - when (or if) to replace the incumbent machine,
  - when (or if) to replace its replacement, etc.,
  - so as to minimize the total cost during the next *N* years.

Equipment replacement problem

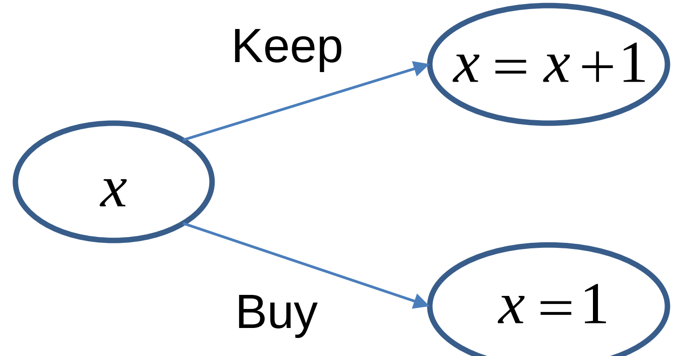
- **Stage:**



- **State:**

At stage *k* (beginning of year *k*), how many possible states (situations) has the machine had if we assume that machine is of age i_0 at the beginning of year 1?

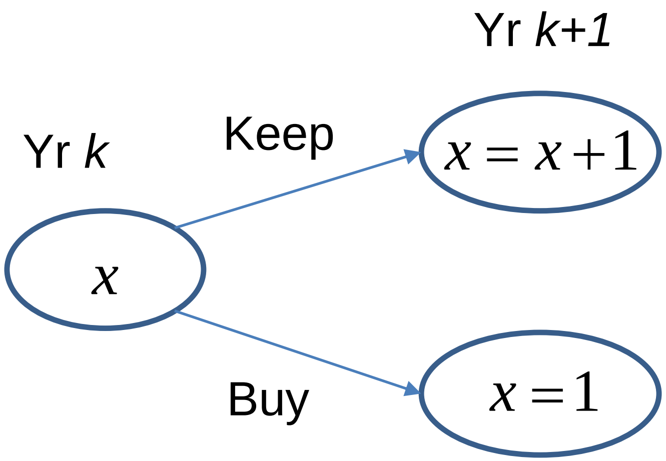
$$\text{Age } x = 1, 2, \ldots, k + i_0 - 1$$



- **Decision:** Keep or buy

Equipment replacement problem

- **Optimal value function**



$$S(x,k) = \text{minimum cost of owning equipment from year } k \text{ through } N\text{, starting year } k \text{ with it just turned age } x$$

- **Recurrence relations**

$$S(x,k) = \min \begin{bmatrix} B: & p - t(x) + c(0) + S(1, k+1) \\ K: & c(x) + S(x+1, k+1) \end{bmatrix}$$

- **Optimal policy function** $P(x,k) = B \text{ or } K$

- **Boundary condition** $S(x, N+1) = -s(x)$

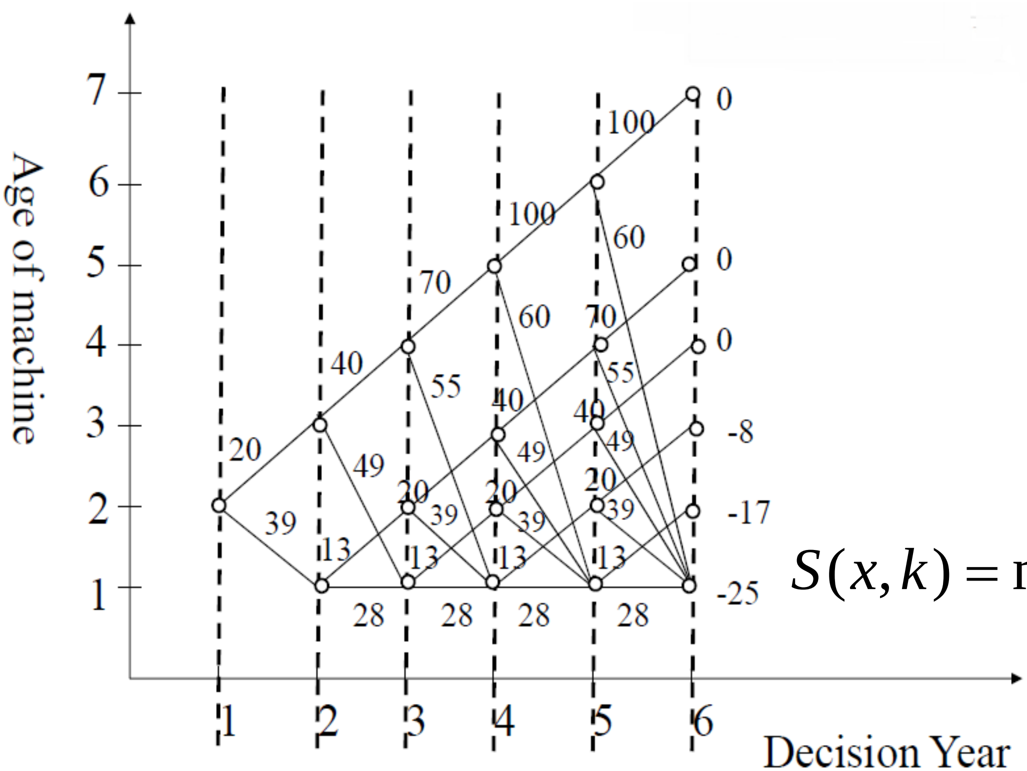
Equipment replacement problem

- **Example**

	0	1	2	3	4	5	6	7
Operating cost, c(j)	10	13	20	40	70	100	100	
Trade-in value, t(j)		32	21	11	5	0	0	
Salvage value, s(j)		25	17	8	0	0	0	0

- **N = 5 years**
- **Age of machine at start of year 1 = 2**
- **Price of new machine = 50**

Equipment replacement problem



Planning period, *N=5*
End is start of year *6*

- **Optimal value function**

$S(\mathrm{x},\mathrm{k}) =$ minimum cost of owning equipment from year k through N, starting year k with it just turned age x

- **Recurrence Relations**

$$S(x,k) = \min \begin{bmatrix} B: & p - t(x) + c(0) + S(1,k+1) \\ K: & c(x) + S(x+1,k+1) \end{bmatrix}$$

- **Boundary conditions**

$$S(x, N+1) = -s(x)$$

Equipment replacement problem

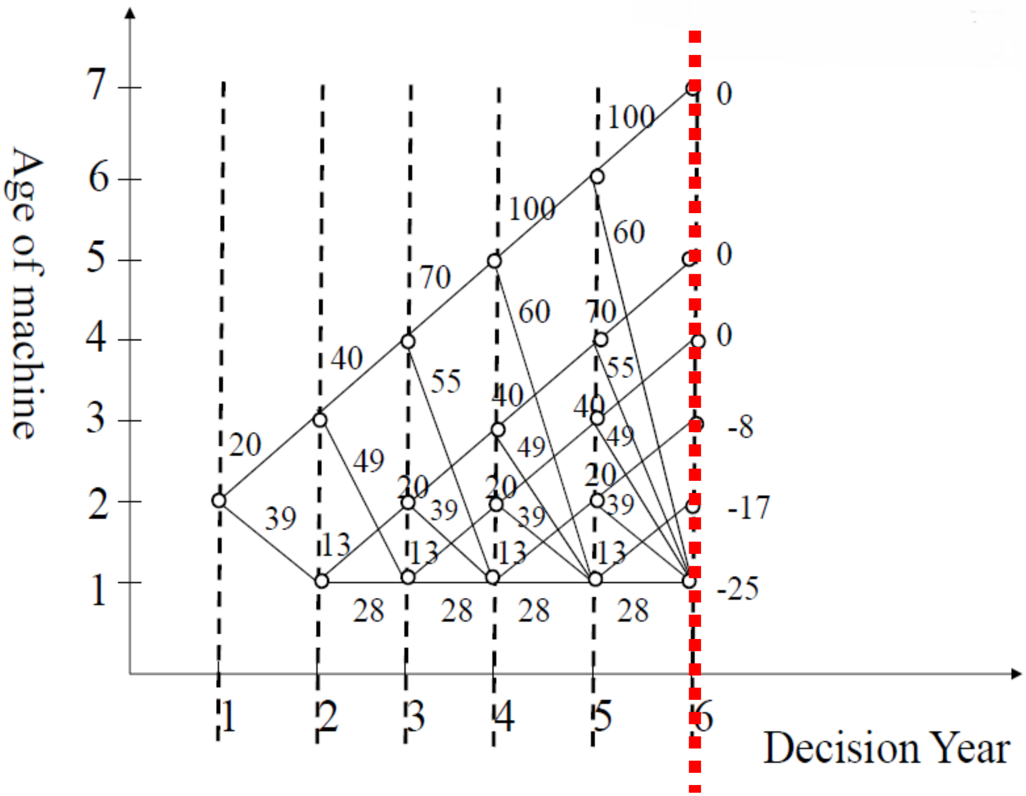
- **Decision year 6**

$$S(1,5+1)=S(1,6)=-25$$

$$S(2,6)=-17$$

$$S(3,6)=-8$$

$$S(4,6)=S(5,6)=S(7,6)=0$$



Equipment replacement problem

- **Decision year 5**

$$S(x,k)=\min\begin{bmatrix} B: & p-t(x)+c(0)+S(1,k+1) \\ K: & c(x)+S(x+1,k+1) \end{bmatrix}$$

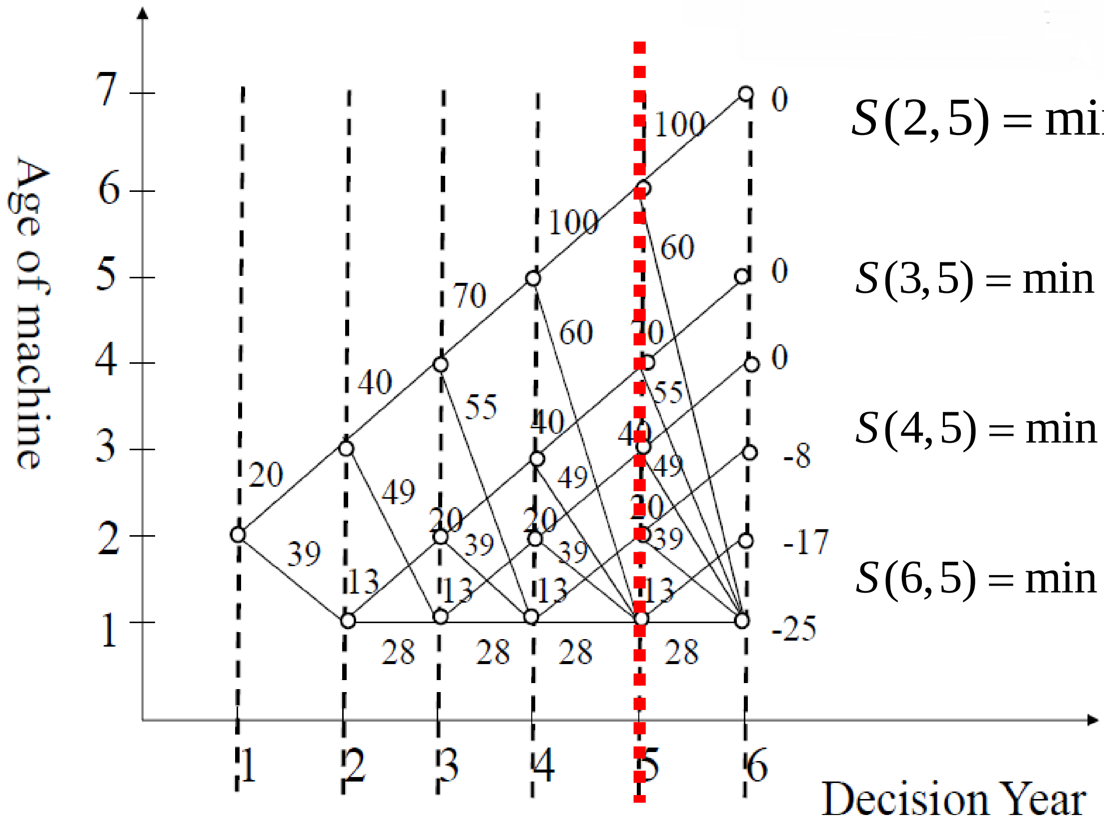
$$S(1,5)=\min\begin{bmatrix} B: & 50-32+10+S(1,6) \\ K: & 13+S(2,6) \end{bmatrix}=-4(K)$$

$$S(2,5)=\min\begin{bmatrix} B: & 50-21+10+S(1,6) \\ K: & 20+S(3,6) \end{bmatrix}=12(K)$$

$$S(3,5)=\min\begin{bmatrix} B: & 50-11+10+S(1,6) \\ K: & 40+S(4,6) \end{bmatrix}=\begin{bmatrix} 24 \\ 40 \end{bmatrix}=24(B)$$

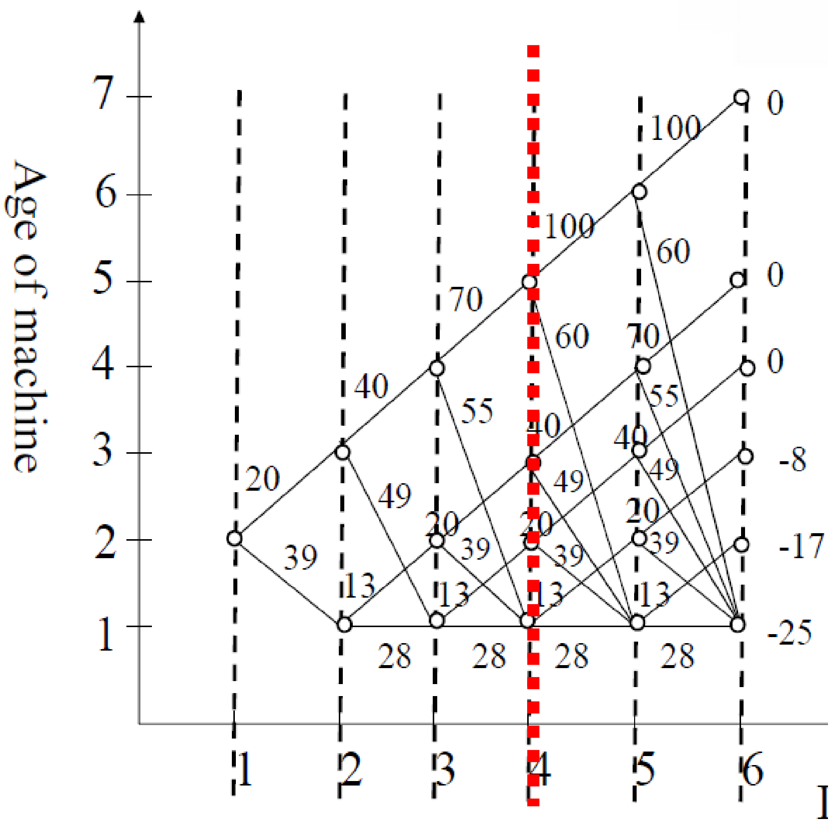
$$S(4,5)=\min\begin{bmatrix} B: & 50-5+10+S(1,6) \\ K: & 70+S(5,6) \end{bmatrix}=\begin{bmatrix} 30 \\ 70 \end{bmatrix}=30(B)$$

$$S(6,5)=\min\begin{bmatrix} B: & 50-0+10+S(1,6) \\ K: & 100+S(7,6) \end{bmatrix}=\begin{bmatrix} 35 \\ 70 \end{bmatrix}=35(B)$$



Equipment replacement problems

- **Decision year 4**



$$S(1,4)=\min\begin{bmatrix} B: & 50-32+10+S(1,5) \\ K: & 13+S(2,5) \end{bmatrix}=\begin{bmatrix} 24 \\ 25 \end{bmatrix}=24(B)$$

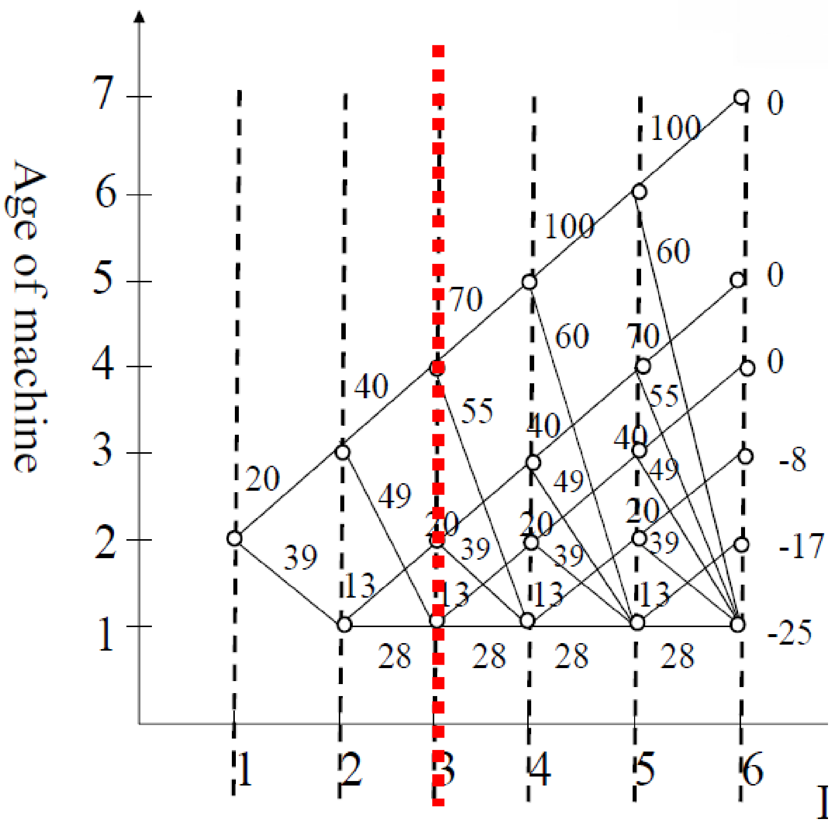
$$S(2,4)=\min\begin{bmatrix} B: & 50-21+10+S(1,5) \\ K: & 20+S(2,5) \end{bmatrix}=\begin{bmatrix} 35 \\ 44 \end{bmatrix}=35(B)$$

$$S(3,4)=\min\begin{bmatrix} B: & 50-11+10+S(1,5) \\ K: & 40+S(4,5) \end{bmatrix}=\begin{bmatrix} 45 \\ 70 \end{bmatrix}=45(B)$$

$$S(5,4)=\min\begin{bmatrix} B: & 50-0+10+S(1,5) \\ K: & 100+S(6,5) \end{bmatrix}=\begin{bmatrix} 56 \\ 135 \end{bmatrix}=56(B)$$

Equipment replacement problem

- **Decision year 3**



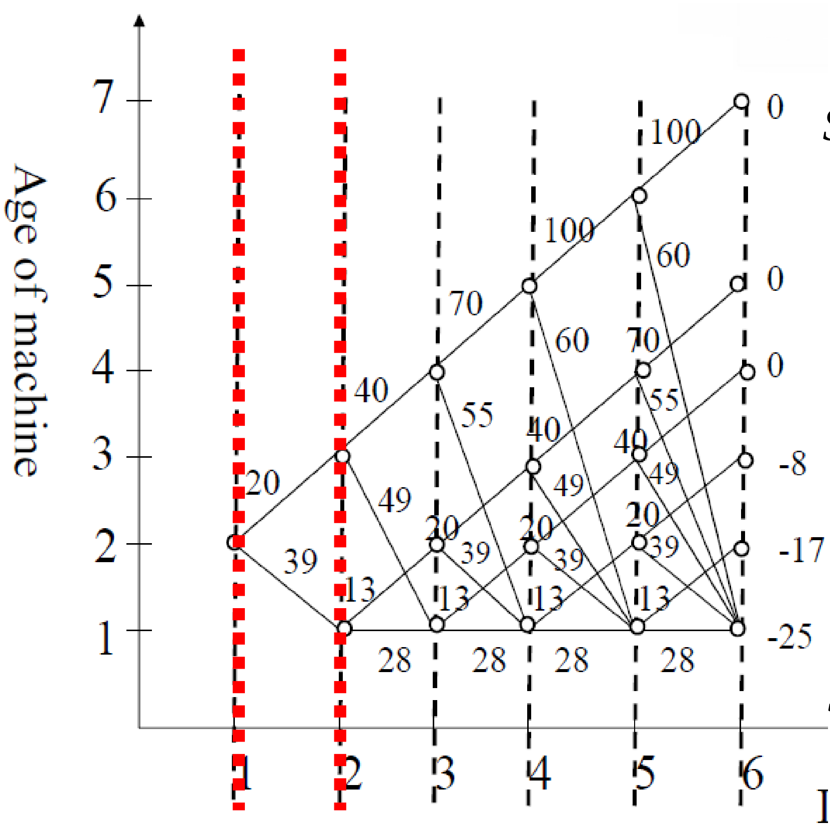
$$S(1,3)=\min\begin{bmatrix} B: & 50-32+10+S(1,4) \\ K: & 13+S(2,4)\end{bmatrix}=\begin{bmatrix}52\\48\end{bmatrix}=48(K)$$

$$S(2,3)=\min\begin{bmatrix} B: & 50-21+10+S(1,4) \\ K: & 20+S(3,4)\end{bmatrix}=\begin{bmatrix}63\\65\end{bmatrix}=63(B)$$

$$S(4,3)=\min\begin{bmatrix} B: & 50-5+10+S(1,4) \\ K: & 70+S(5,4)\end{bmatrix}=\begin{bmatrix}79\\126\end{bmatrix}=79(B)$$

Equipment replacement problem

- **Decision year 2**



$$S(1,2)=\min\begin{bmatrix} B: & 50-32+10+S(1,3) \\ K: & 13+S(2,3)\end{bmatrix}=\begin{bmatrix}76\\76\end{bmatrix}=76(B,K$$

$$S(3,2)=\min\begin{bmatrix} B: & 50-11+10+S(1,3) \\ K: & 40+S(4,3)\end{bmatrix}=\begin{bmatrix}97\\119\end{bmatrix}=97(B)$$

$$S(4,3)=\min\begin{bmatrix} B: & 50-5+10+S(1,4) \\ K: & 70+S(5,4)\end{bmatrix}=\begin{bmatrix}79\\126\end{bmatrix}=79(B)$$

- **Decision year 1**

$$S(2,1)=\min\begin{bmatrix} B: & 50-21+10+S(1,2) \\ K: & 20+S(3,2)\end{bmatrix}=\begin{bmatrix}115\\117\end{bmatrix}=115(B$$

- **Answer: BKBBK or BBKBK**

Equipment replacement problem

- A more complex problem

Additional decision: “overhaul” besides “keep”, “replace”.

An overhauled machine is better than one not overhauled, but not as good as a new one.

Performance depends on actual age of equipment and on the number of years since last overhauled, but is independent of its previous overhauls.

$$e(k,i,j) = \text{cost of exchanging machine of age } i\text{, last overhauled at age } j\text{, for a new machine at the start of year } k$$

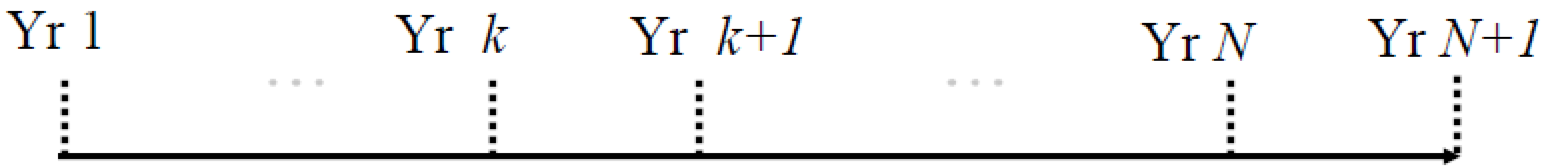
$$c(\mathrm{k,i,j}) = \text{net operating cost during year } k \text{ of machine age } i\text{, last overhauled at age } j$$

$$\mathrm{o}(k,i) = \text{cost of overhauling machine age } i \text{ at start of year } k$$

$$s(i) = \text{salvage value at the end of year } N \text{ of machine which as just become age } i\text{, last overhauled at age } j$$

Equipment replacement problem

- **Stage:**



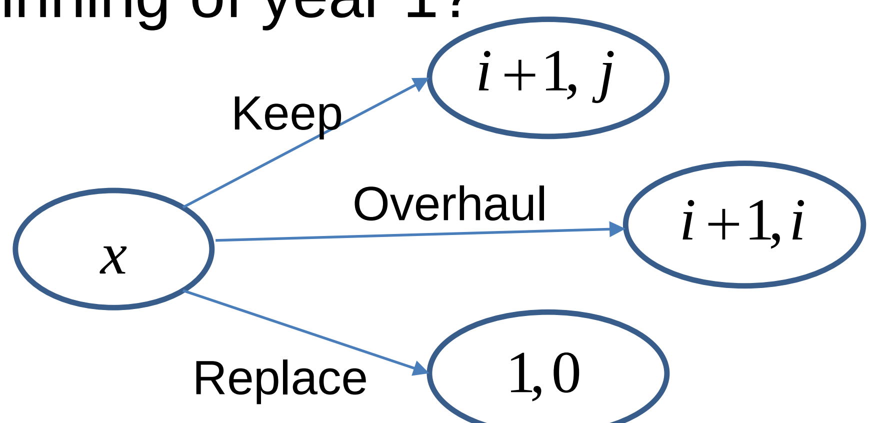
- **State:**

At stage *k* (beginning of year *k*), how many possible states (situations) has the machine had if we assume that machine is of age i_0 at the beginning of year 1?

Age $x = 1, 2, \ldots, k + i_0 - 1$

- **Overhauled at age j**

$$j = 1, 2, \ldots, k + j_0 - 1$$



Equipment replacement problem

- What would you require to know before you can proceed to solve?
- Age of machine, age last overhauled and where we are
- **Optimal value function**

$f(k,i,j) =$ optimal cost during remaining process given we start year k with machine age i years, last overhauled at age j years

- **Recurrence relations**

$$S(x,k) = \min \begin{bmatrix} R: e(k,i,j) + c(k,0,0) + f(k+1,1,0) \\ K: c(k,i,j) + f(k+1,i+1,j) \\ O: o(k,i) + c(k,i,j) + f(k+1,i+1,i) \end{bmatrix}$$

- **Boundary condition**

$$f(N+1,i,j) = -s(i,j)$$

Resource allocation problem

- **Problem description**
- Given *X* units of a resource and told that this resource must be distributed among *N* activities. And given *N* data tables $r_i(x)(\text{for } i=1,\ldots,N \textit{ and } x=0,1,\ldots,X)$ representing the return realized from an allocation of *x* units of resource to activity *i*. The problem is to allocate all of the *X* units of resource to the activities so as to maximize the total return, i.e. to choose *N* nonnegative integers x_i, that maximize

$$\sum_{i=1}^{N} r_i(x_i)$$

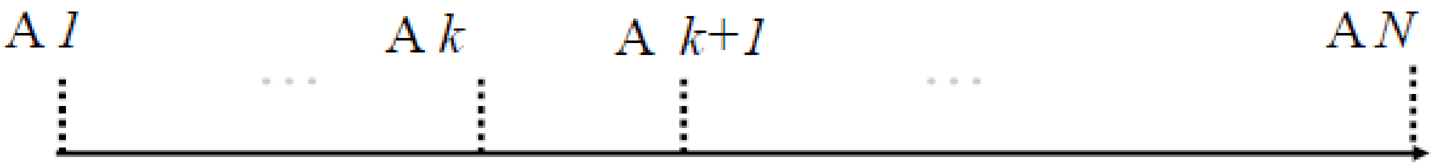
subject to

$$\sum_{i=1}^{N} x_i = X$$

Note that
This problem illustrates the application of DP to problem areas that are not really dynamic (requiring a sequence of decisions).

Resource allocation problem

- **Stages**



- **To use DP, we view the problem in this way:**

With the X units we first allocate an amount x_1 to activity 1.

With remaining X- x_1 units we allocate x_1 *to activity 2.*

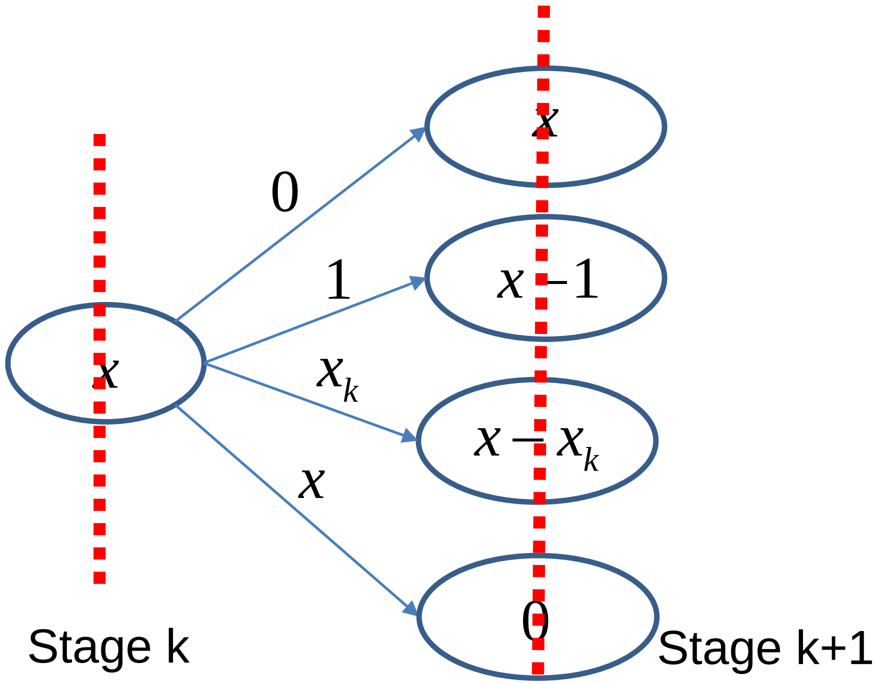
And so on

Resource allocation problem

- **States**
  - At each activity k, what is the information required?
  - How many units left?
  - Which type of decision will we make?

Remaining units

$$x = 0, 1, 2, \ldots X$$



Resource allocation problem

- **Recurrence relation**

$$f_k(x) = \max_{x_k = 0,1,\ldots,x} [r_k(x_k) + f_{k+1}(x - x_k)]$$

where x_k is the allocation to activity k and $f_k(x)$ must be computed for $x = 0, \ldots, X$

- **Boundary condition** $f_N(x) = r_N(x)$

- **The answer is** $f_1(X)$

Resource allocation problem

- **Example**

x_1	$r_1(x_1)$	x_2	$r_2(x_2)$	x_3	$r_3(x_3)$	x_4	$r_4(x_4)$
0	0	0	0	0	0	0	0
1	3	1	1	1	2	1	1
2	7	2	2	2	4	2	3
3	10	3	4	3	6	3	6
4	12	4	8	4	8	4	9
5	13	5	13	5	10	5	12
6	14	6	17	6	12	6	14
7	14	7	19	7	14	7	16
8	14	8	20	8	16	8	17

- **Stage 3**

$$f_3(0) = 0(0)$$

$$f_3(1) = 2(1)$$

$$f_3(2) = 4(2)$$

$$f_3(3) = 6(0,3)$$

$$f_3(4) = 9(0)$$

$$f_3(5) = 12(0)$$

$$f_3(6) = 14(0,1)$$

$$f_3(7) = 16(0,1,2)$$

$$f_3(8) = 18(1,2,3)$$

Resource allocation problem

- **Example**

x_1	$r_1(x_1)$	x_2	$r_2(x_2)$	x_3	$r_3(x_3)$	x_4	$r_4(x_4)$
0	0	0	0	0	0	0	0
1	3	1	1	1	2	1	1
2	7	2	2	2	4	2	3
3	10	3	4	3	6	3	6
4	12	4	8	4	8	4	9
5	13	5	13	5	10	5	12
6	14	6	17	6	12	6	14
7	14	7	19	7	14	7	16
8	14	8	20	8	16	8	17

- **Stage 2**

$$f_2(0) = 0(0)$$

$$f_2(1) = \max[0+2, 1+0] = 2(0)$$

$$f_2(2) = \max[0+4, 1+2, 2+0] = 4(0)$$

$$f_2(3) = \max[0+6, 1+4, 2+2, 4+0] = 6($$

$$f_2(4) = 9(0)$$

$$f_2(5) = 13(5)$$

$$f_2(6) = 17(6)$$

$$f_2(7) = 19(6,7)$$

$$f_2(8) = 21(6,7)$$

Resource allocation problem

- **Example**

x_1	$r_1(x_1)$	x_2	$r_2(x_2)$	x_3	$r_3(x_3)$	x_4	$r_4(x_4)$
0	0	0	0	0	0	0	0
1	3	1	1	1	2	1	1
2	7	2	2	2	4	2	3
3	10	3	4	3	6	3	6
4	12	4	8	4	8	4	9
5	13	5	13	5	10	5	12
6	14	6	17	6	12	6	14
7	14	7	19	7	14	7	16
8	14	8	20	8	16	8	17

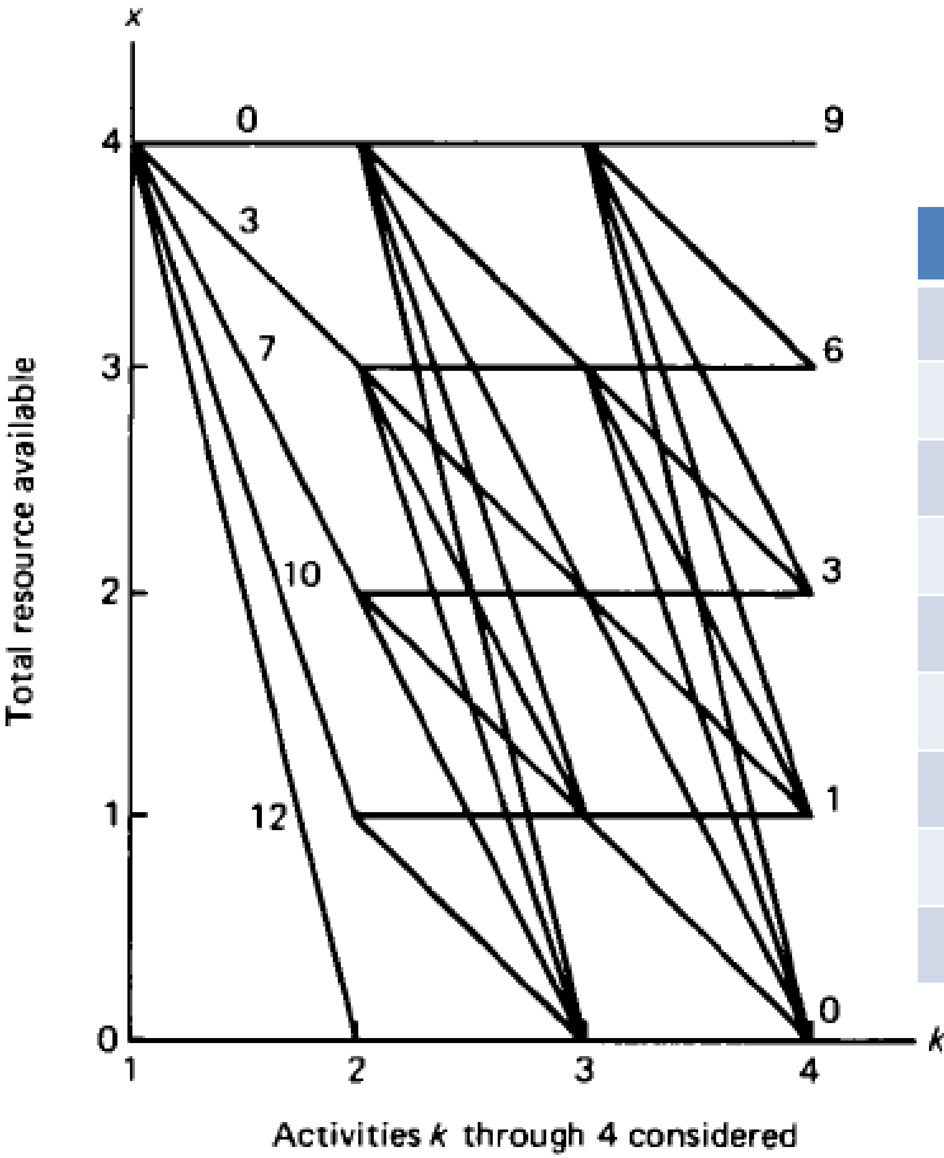
- **Stage 1**

$$f_1(8) = \max[0+21, 3+19, 7+17, 10+13, 12+9, 13+6, 14+4, 14+2, 14+0] = 24(2)$$

- **Result**

Allocate 2 units to activity 1, and 6 units to 2

Resource allocation problem



x_1	$r_1(x_1)$	x_2	$r_2(x_2)$	x_3	$r_3(x_3)$	x_4	$r_4(x_4)$
0	0	0	0	0	0	0	0
1	3	1	1	1	2	1	1
2	7	2	2	2	4	2	3
3	10	3	4	3	6	3	6
4	12	4	8	4	8	4	9
5	13	5	13	5	10	5	12
6	14	6	17	6	12	6	14
7	14	7	19	7	14	7	16
8	14	8	20	8	16	8	17