EEEN 460
Optimal Control

2020 Spring

Lecture X
Dynamical Programming

Lecture X Part I

Dynamic Programming

Those who cannot remember the past
are condemned to repeat it.

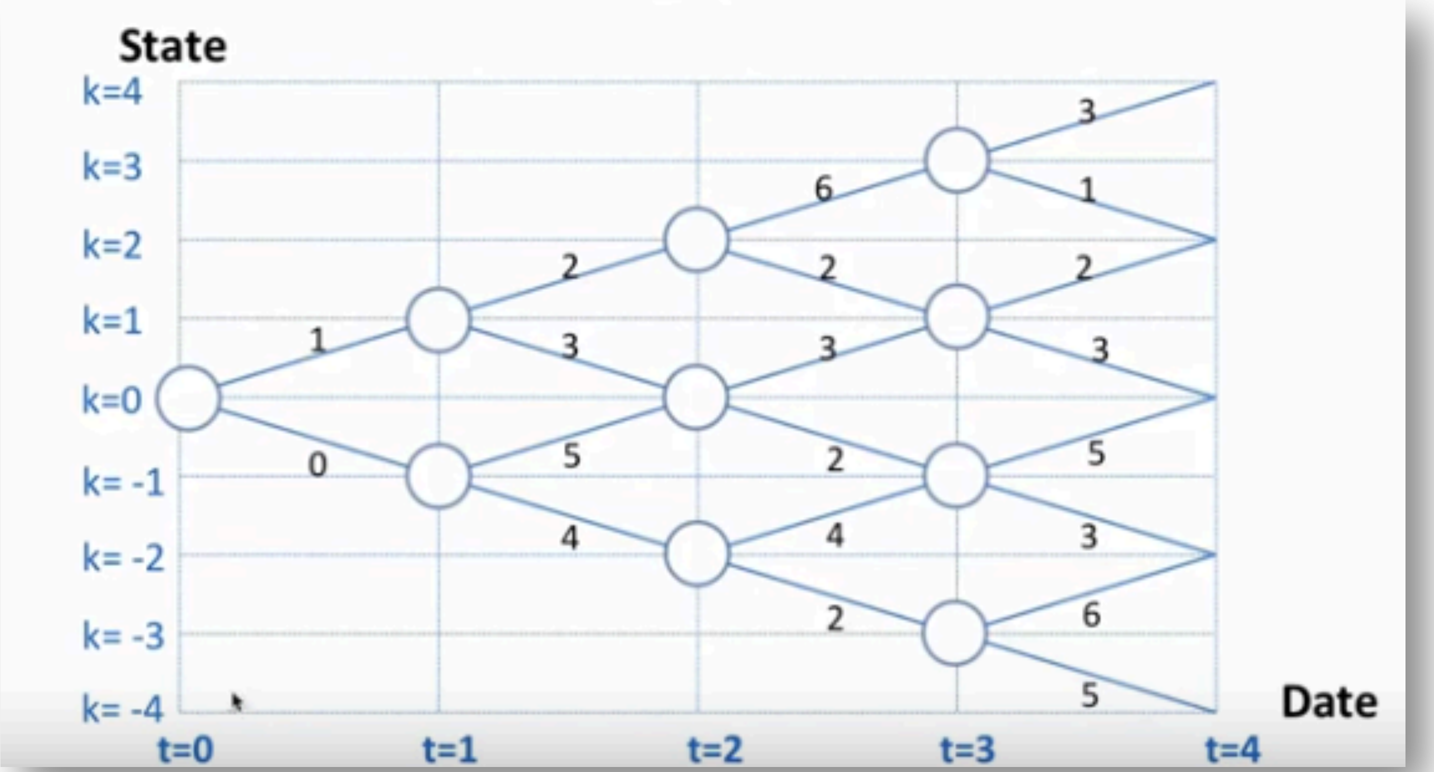
-Dynamic Programming

- [How should I explain dynamic programming to a beginner?]()

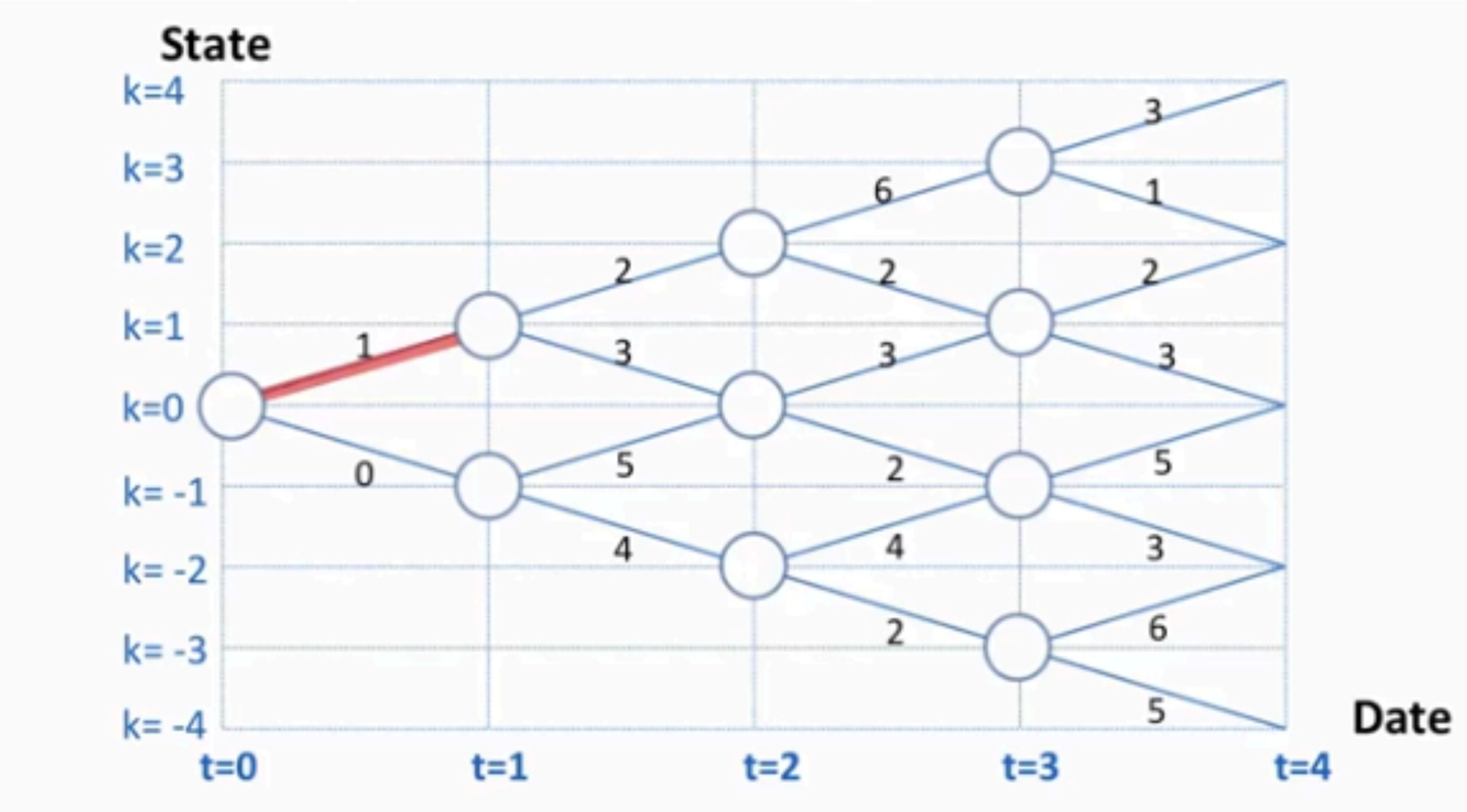
write down "1+1+1+1+1+1+1+1 =" on a sheet of paper
"What's that equal to?"
counting "Eight!"
write down another "1+" on the left
"What about that?"
quickly "Nine!"
"How'd you know it was nine so fast?"
"You just added one more"
"So you didn't need to recount because you remembered there were eight! Dynamic Programming is just a fancy way to say 'remembering stuff to save time later'"

A Simple Dynamic Optimization Problem

A Simple dynamic optimization problem is represented by this diagram



The horizontal axis shows time(date), there are four periods.The vertical axis represents state.

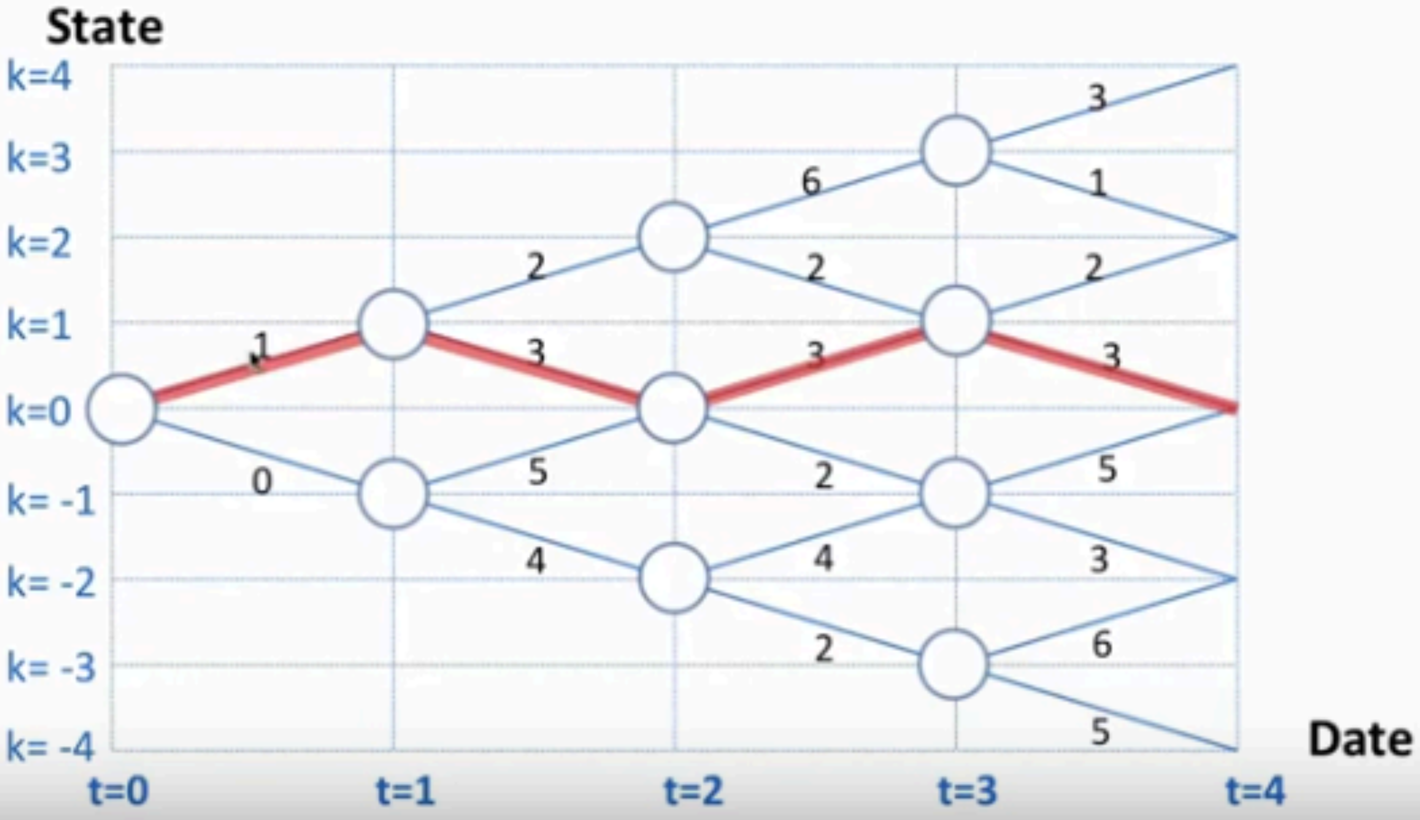


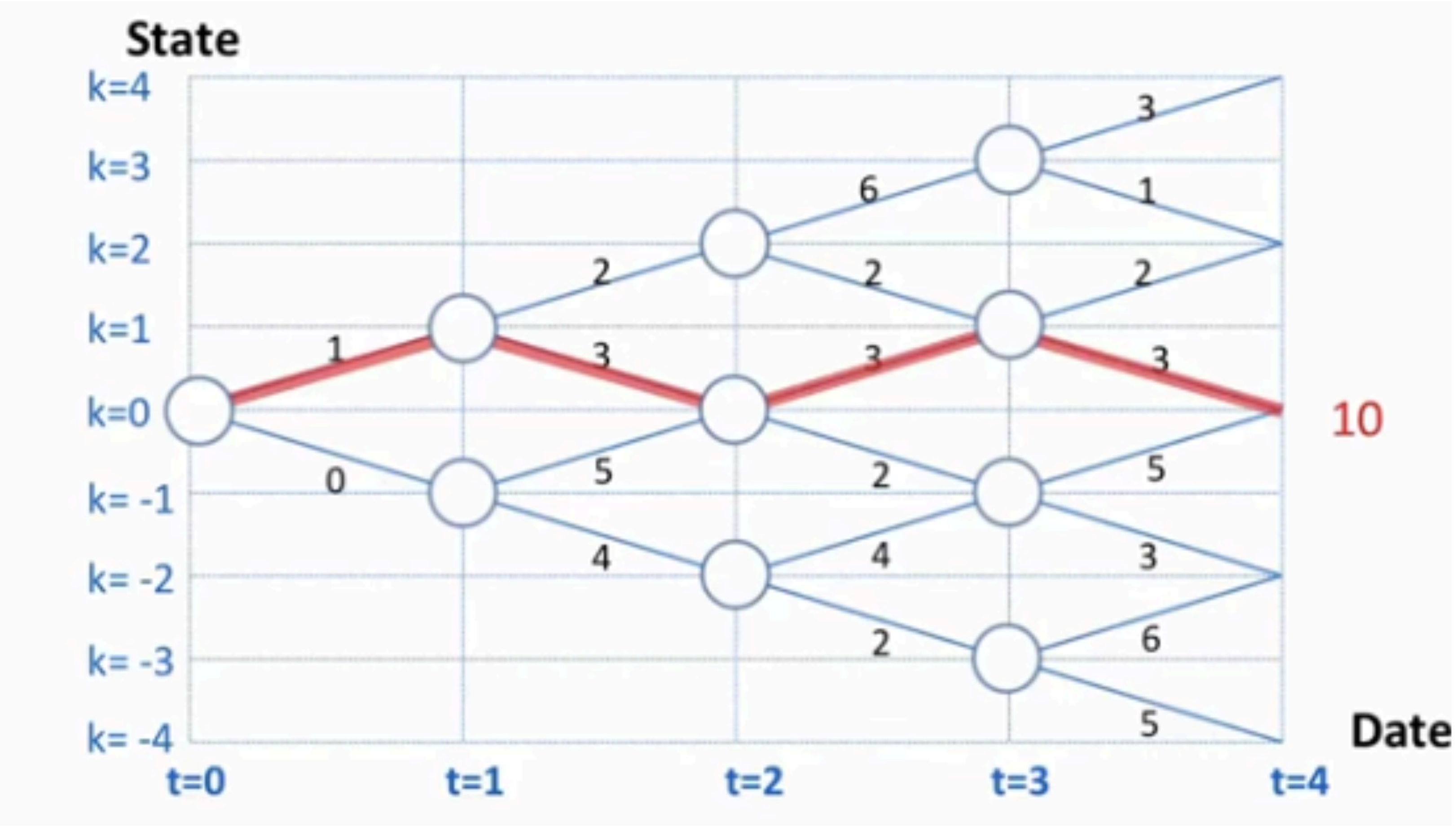
Starting from state k=0 at t=0, determine 'up' or 'down' for each state to maximize the path value.

The objective is to maximize the path value as we go from t=0 to t=4.

1. Backward Induction

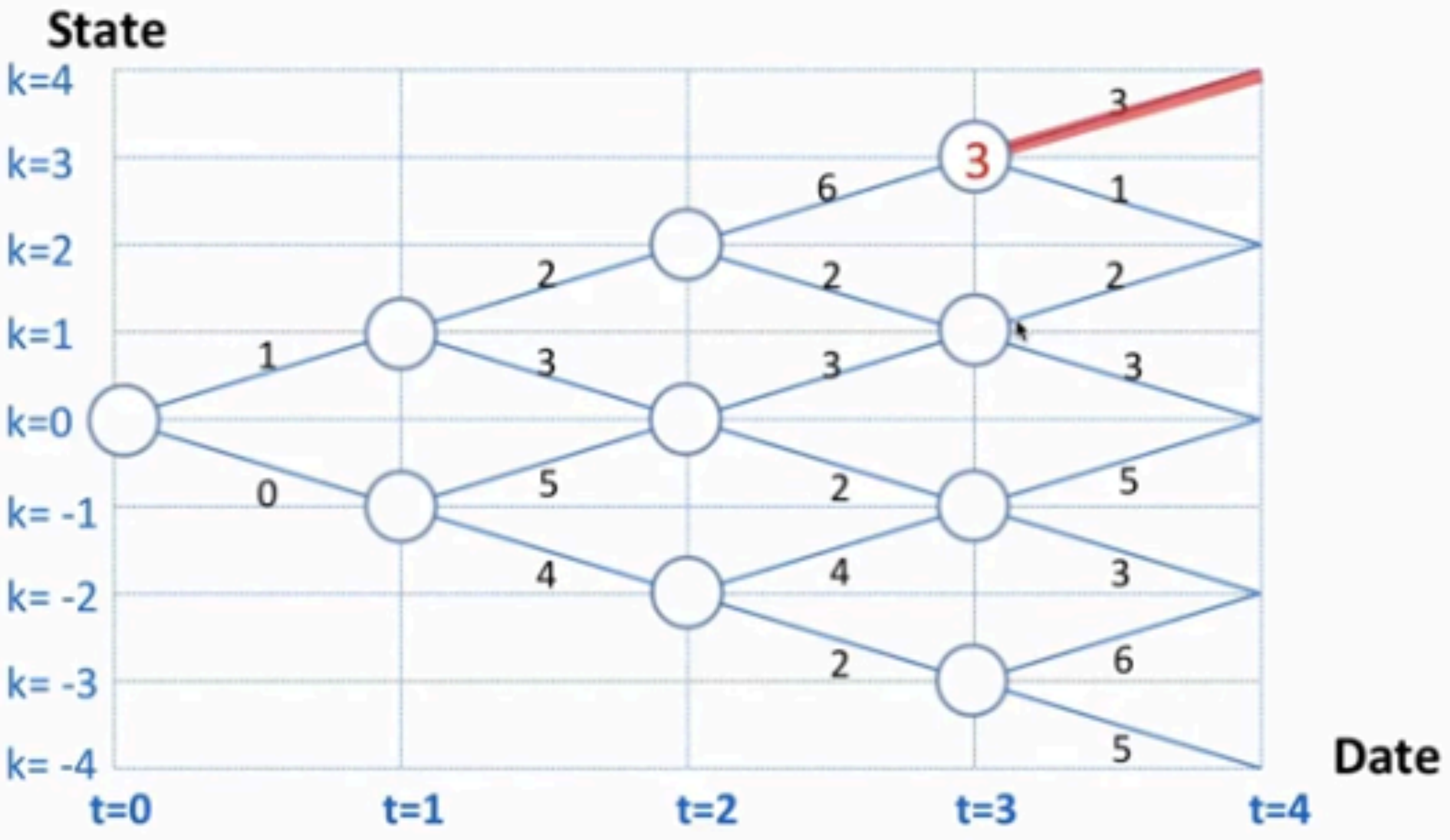
- Starting from t=0, k=0, determine 'up or down' for each date to maximise the sum of numbers you pass.





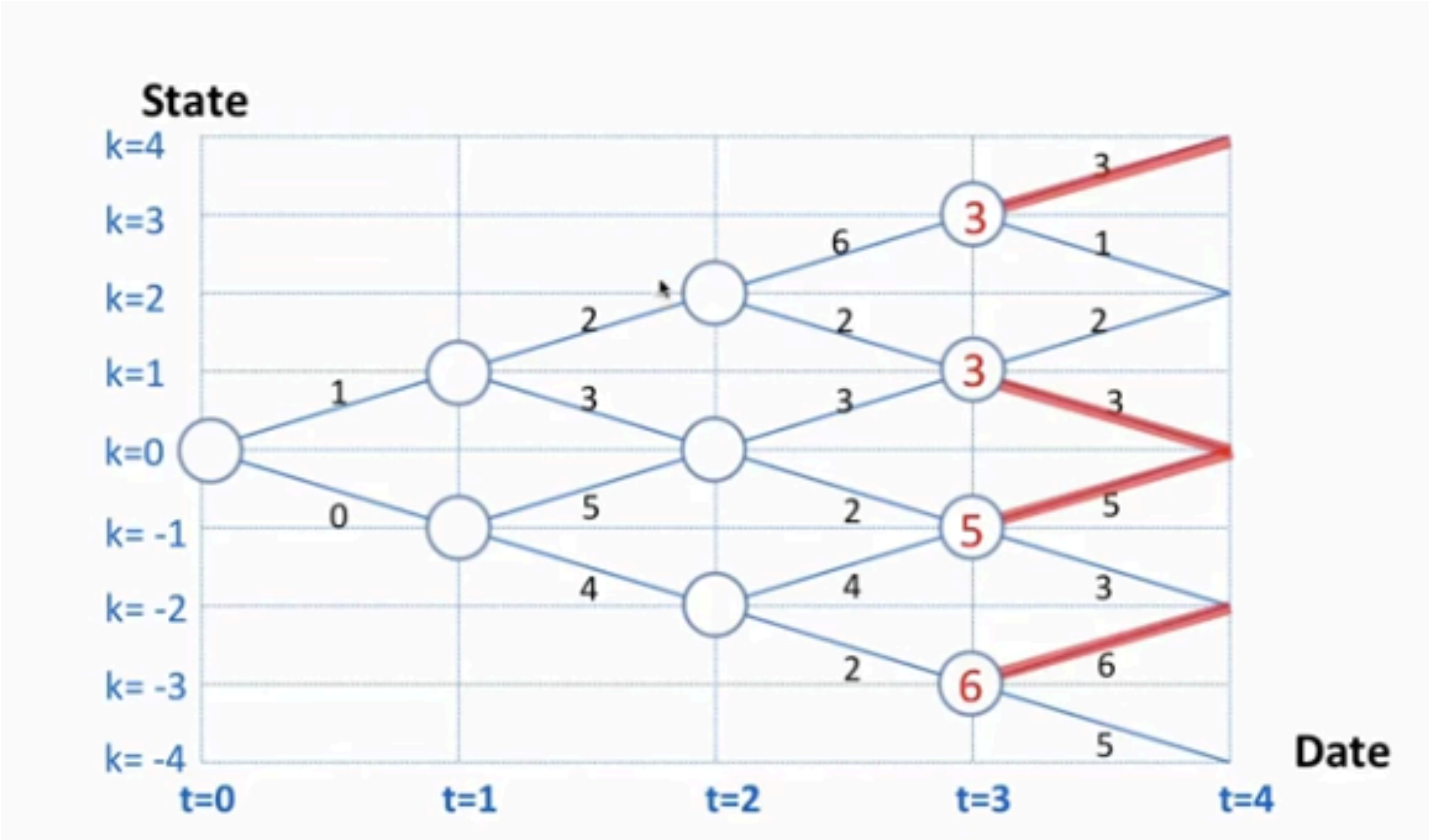
For example if we follow this path, its path value is:
1+3+3+3=10
There are 16 possible paths here. An obvious way to solution is to try these 16 paths one by one.

Backward Induction



If we start the problem from the back than the solution becomes much easier compared to calculating 16 paths one by one.

This is **backward induction**

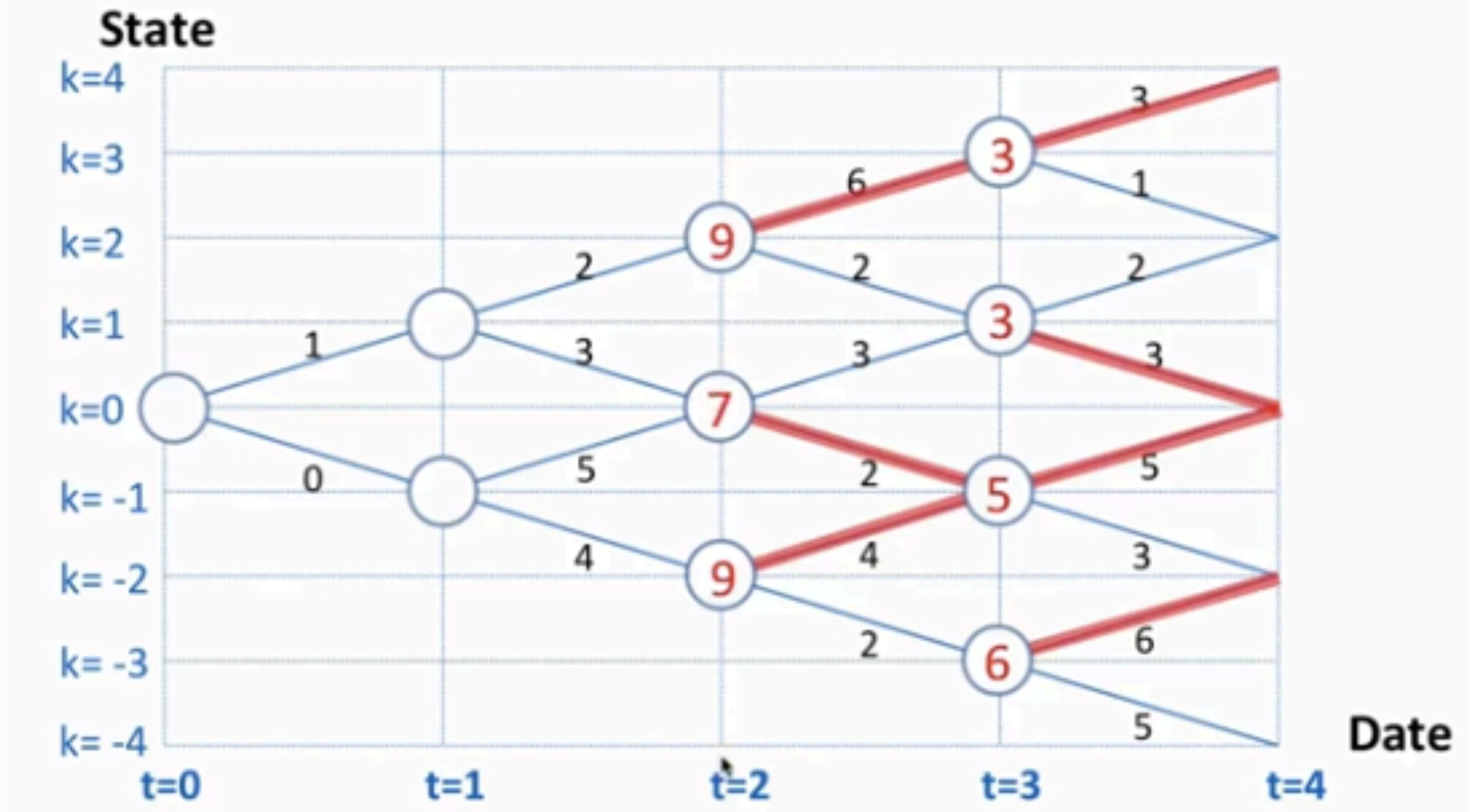


At t=3, state k=3 choose **UP**

k=1 choose **DOWN**

k=-1 choose **UP**

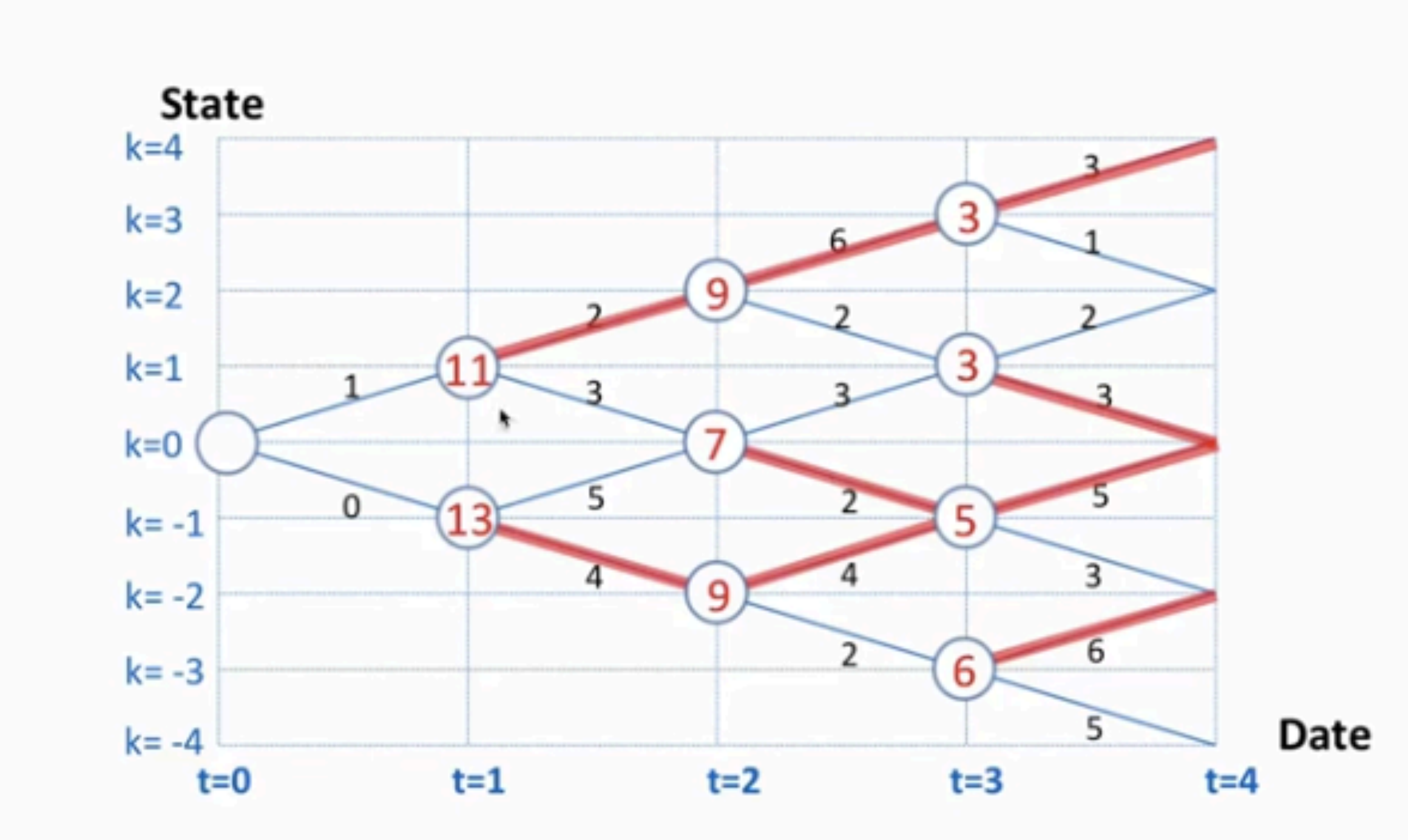
k=-3 choose **UP**



At t=2, state k=2 choose **UP**

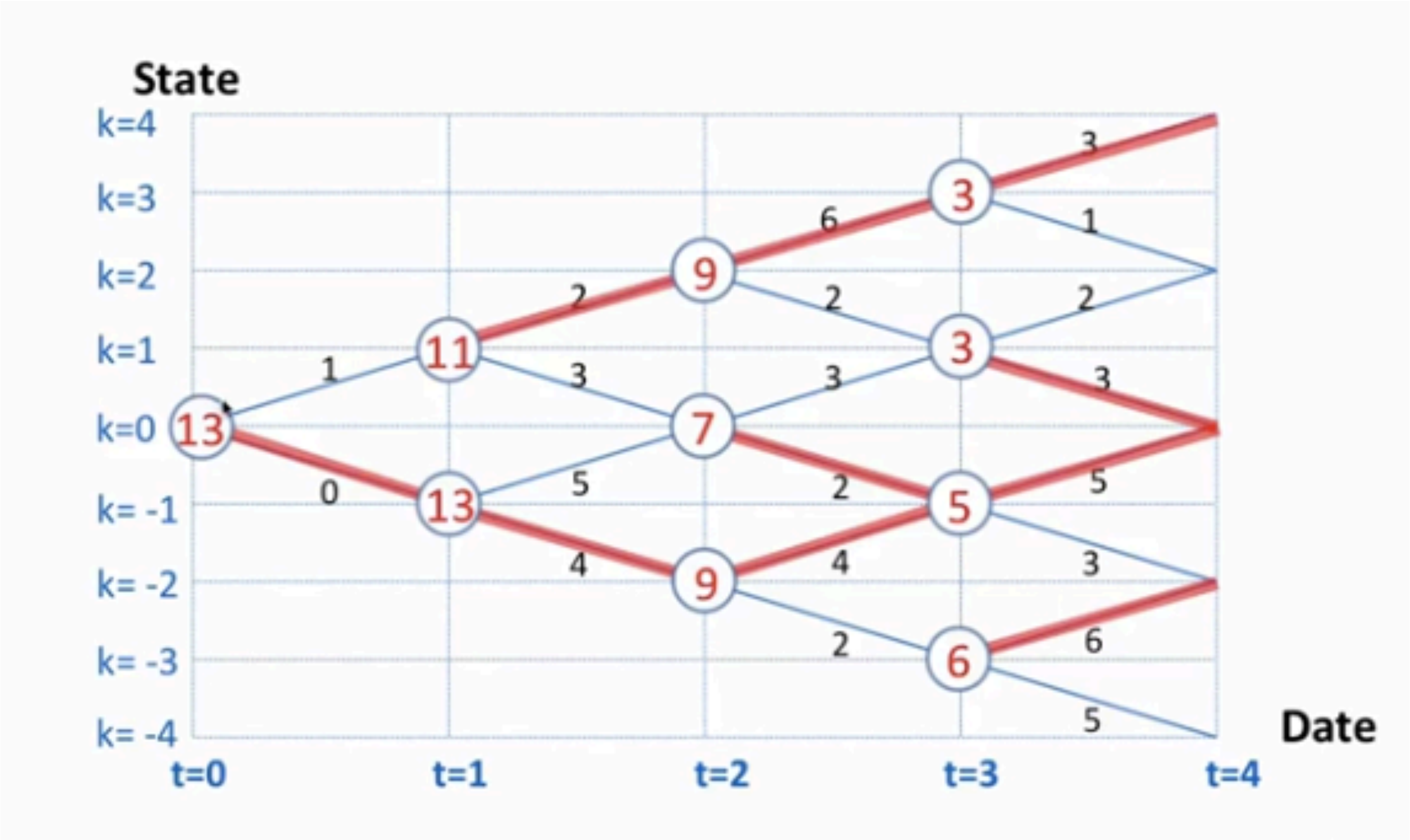
k=0 choose **DOWN**

k=-2 choose **UP**



At t=1, state k=1 choose **UP**

k=-1 choose **DOWN**



Finally, at the initial state t=0, k=0, you choose **DOWN**.
OPTIMAL SOLUTION: **DOWN**, **DOWN**, **UP**, **UP**.

End of Part I

Lecture X Part II

- To Mathematically express this method Let us introduce some notions.

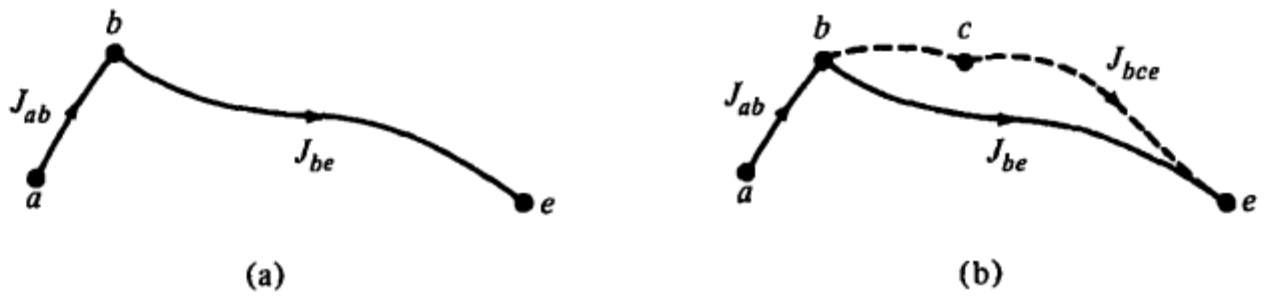
THE PRINCIPLE OF OPTIMALITY

The *optimal* path for a multistage decision process is shown in Fig. 1(a). Suppose that the first decision (made at *a)* results in segment *a-b* with cost J_{ab}

(a) (b)

and that the remaining decisions yield segment *b-e* at a cost of J_{be}, . The *minimum* cost J_{ae}^* from *a* to *e* is therefore

$$J_{ae}^* = J_{ab} + J_{be}.$$



ASSERTION: If *a-b-e* is the optimal path from *a to e,* then *b-e* is the optimal path from *b t o e.*

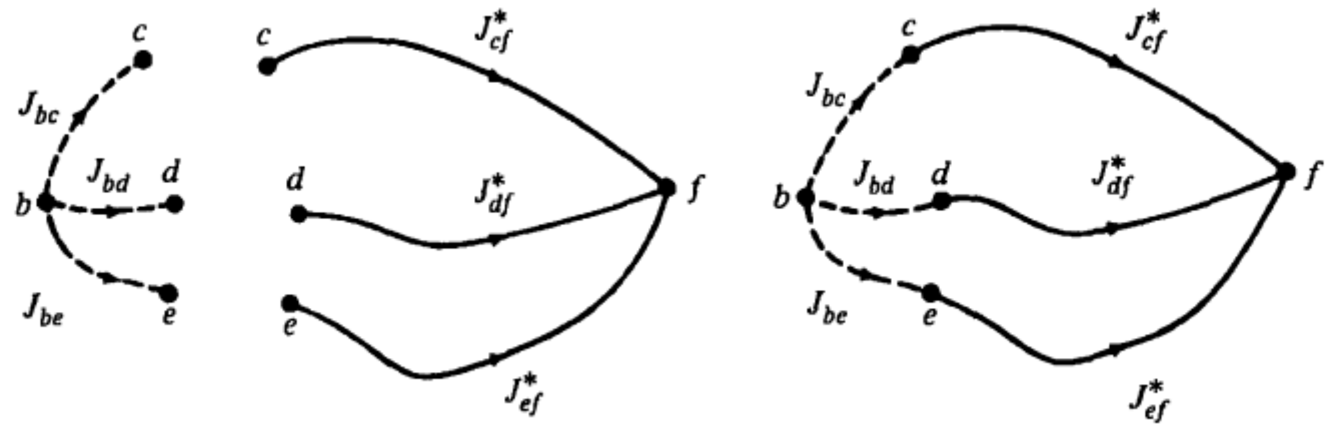
Proof by contradiction: **Suppose *b-c-e* in the figure above is the optimal path from *b to e;* then**

$$J_{bce} < J_{be},$$

and

$$J_{ab} + J_{bce} < J_{ab} + J_{be} = J^*_{ae}$$

but this can be satisfied only by violating the condition that *a-b-e* is the optimal path from *a to e.* Thus the assertion is proved.



(a) (b) (c)

Consider a process whose current state is *b.* The paths resulting from all allowable decisions at *b* are shown in Fig. 2(a). The optimal paths from *c, d,* and *e* to the terminal point/ are shown in Fig. 2(b).

The paths in Fig. 2(c) are the only candidates for the optimal trajectory from *b* to *f.* The optimal trajectory that starts at *b* is found by comparing

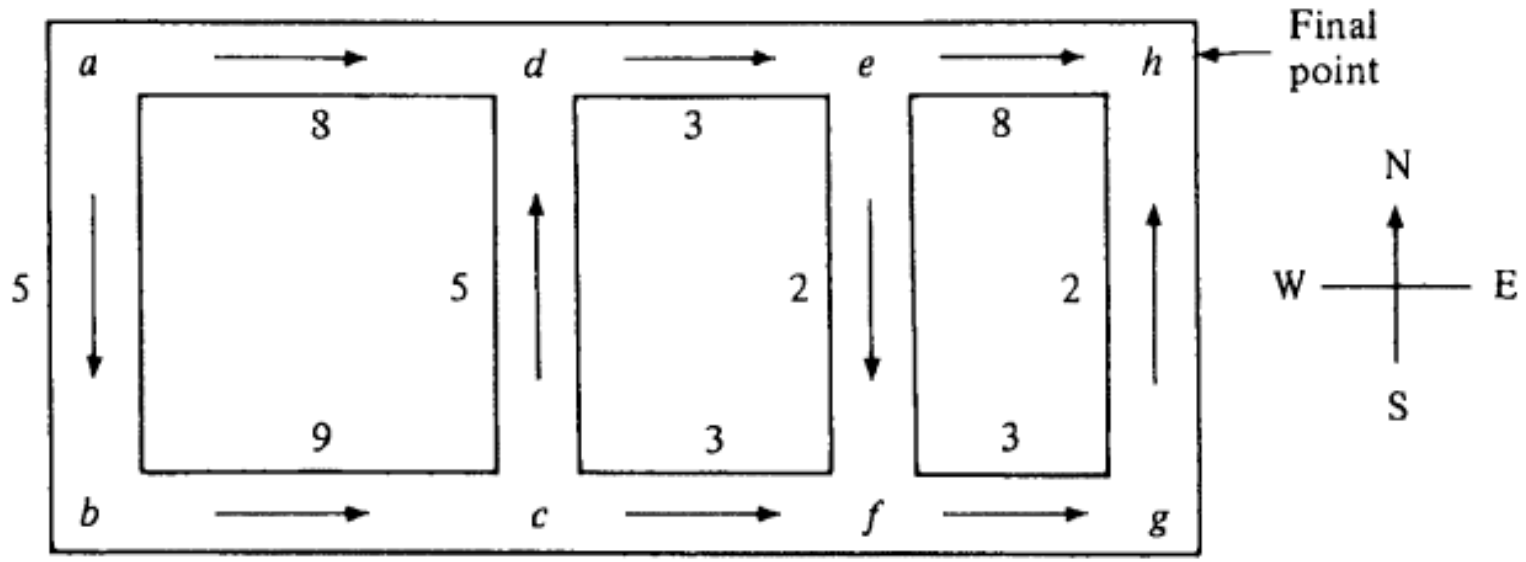
$$C^*_{bcf} = J_{bc} + J^*_{cf}$$
$$C^*_{bdf} = J_{bd} + J^*_{df}$$
$$C^*_{bef} = J_{be} + J^*_{ef}.$$

The minimum of these costs must be the one associated with the optimal decision at point *b.*

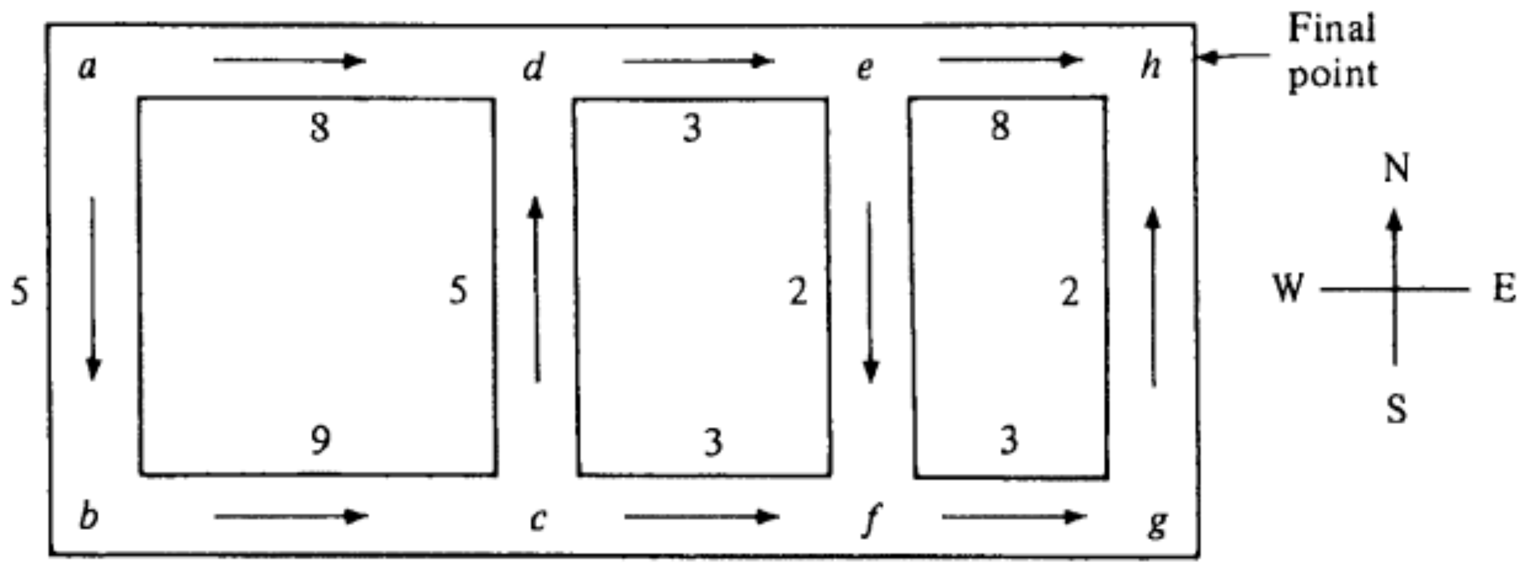
Dynamic Programming

Dynamic programming is a computational technique which extends the above decision-making concept to *sequences* of decisions which together define an optimal policy and trajectory.

DYNAMIC PROGRAMMING APPLIED TO A ROUTING PROBLEM

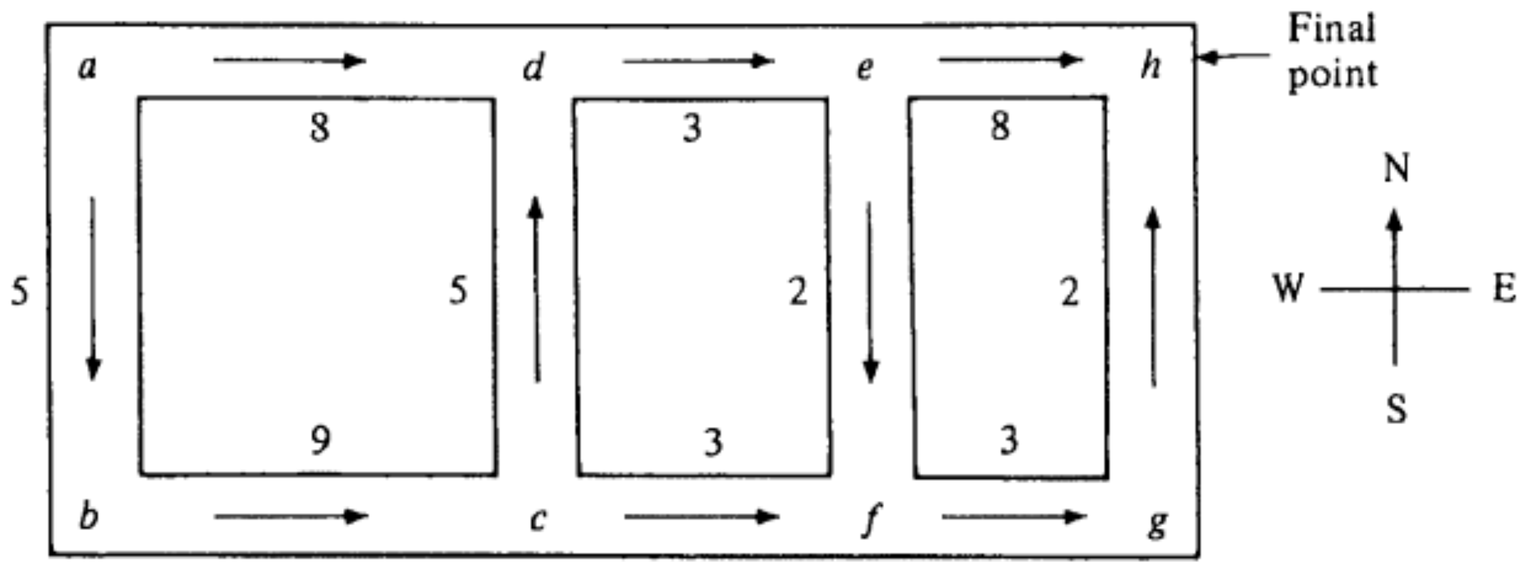


A motorist wishes to know how to minimize the cost of reaching some destination *h* from his current location. He can only travel (one-way as indicated) on the streets shown on his map (Figure above), and the intersection-to-intersection costs are given.



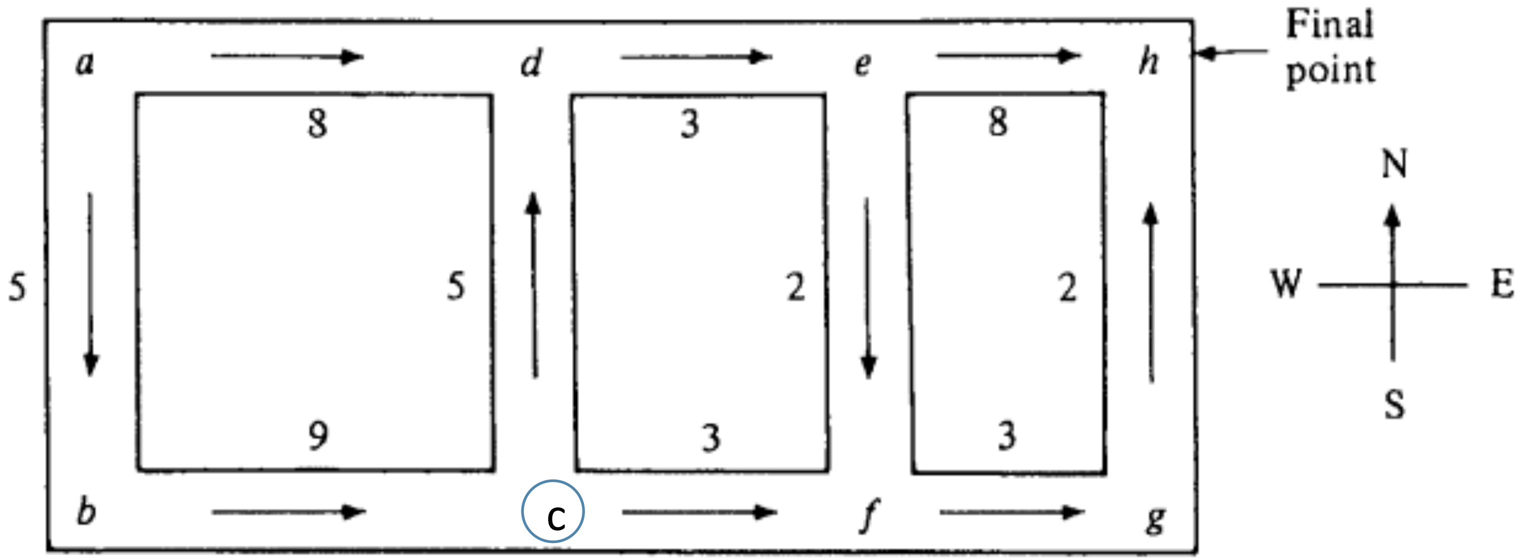
Instead of trying all allowable paths leading from each intersection to *h* and selecting the one with lowest cost (an exhaustive search), consider the application of the principle of optimality.

In this problem, "state" refers to the intersection and a "decision" is the choice of heading (control) elected by the driver when he leaves an intersection.



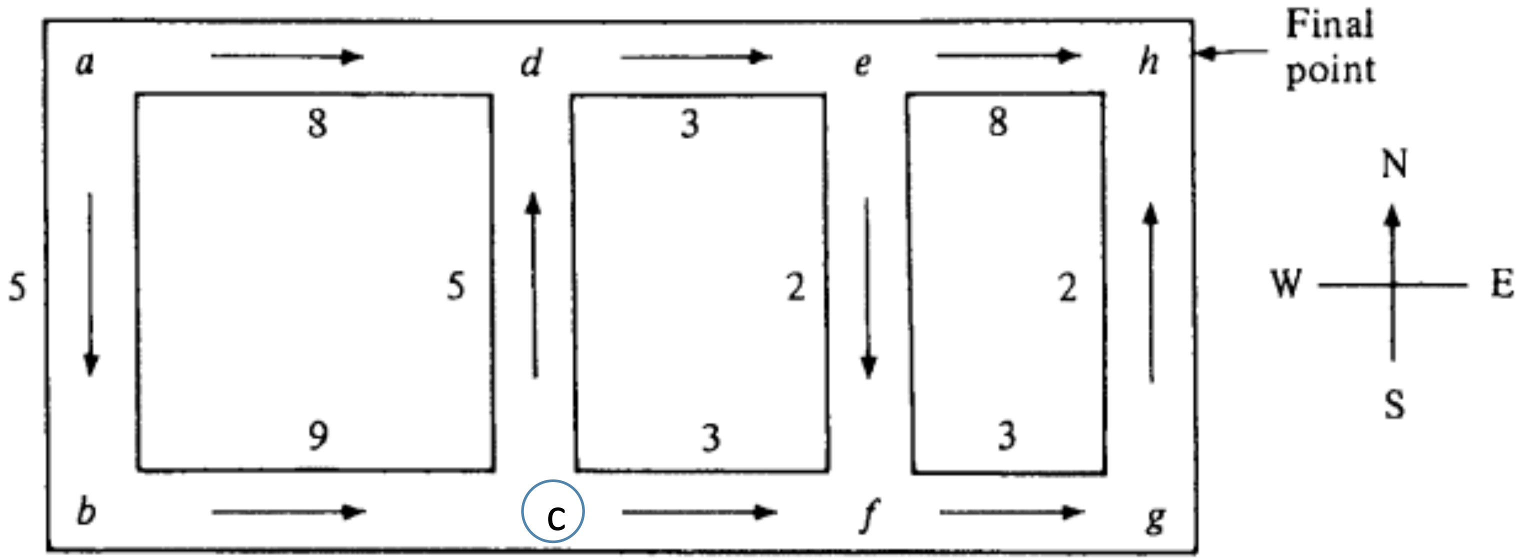
Instead of trying all allowable paths leading from each intersection to *h* and selecting the one with lowest cost (an exhaustive search), consider the application of the principle of optimality.

In this problem, "state" refers to the intersection and a "decision" is the choice of heading (control) elected by the driver when he leaves an intersection.



Suppose the motorist is at *c;*
from there he can go, only to *d* or *f,*

Which one is the optimal decision ?



Suppose the motorist is at *c*;

from there he can go, only to *d* or *f*,

and then

on to *h*.

Let J_{cd} denote the cost of moving from *c* to *d* and J_{cf} the cost from *c* to *f*.

J^*_{dh} = 10 and J^*_{fh} = 5.

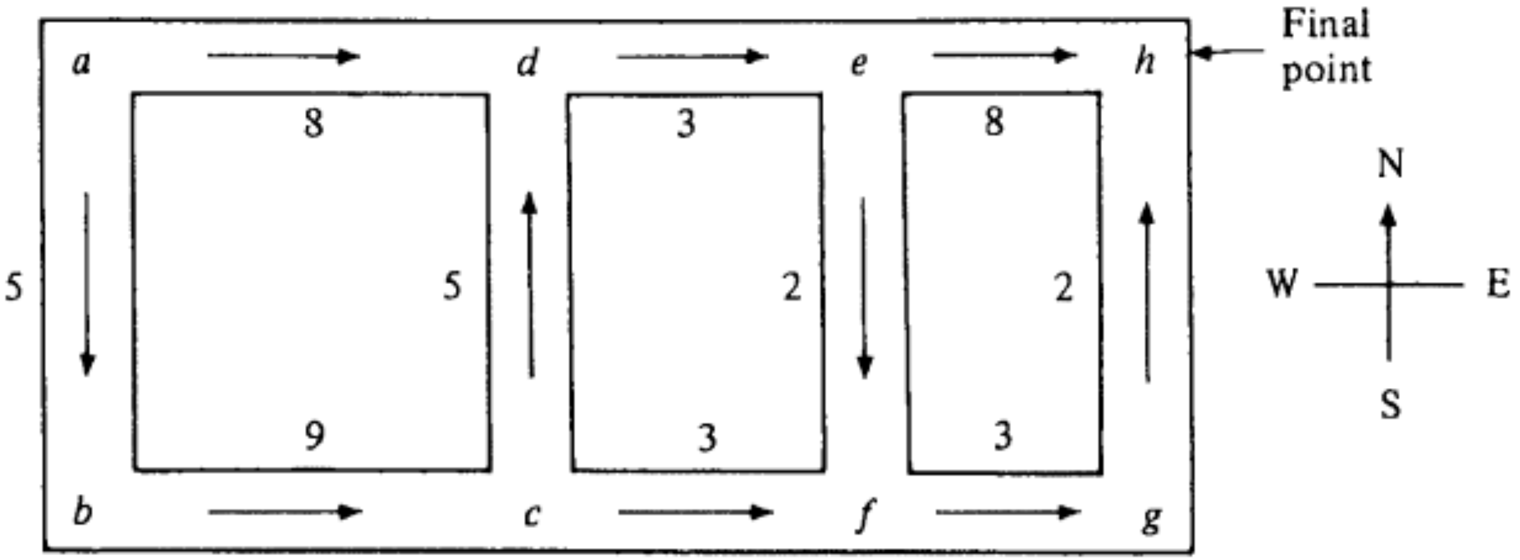
Then the minimum cost J^*_{ch} to reach *h* from *c* is the smaller of

$C^*_{cdh} = J_{cd} + J^*_{dh}$ **and** $C^*_{cfh} = J_{cf} + J^*_{fh}$

Thus

$$
\begin{aligned}
J^*_{ch} &= \min\{C^*_{cdh}, C^*_{cfh}\} \\
&= \min\{15, 8\} \\
&= 8
\end{aligned}
$$

Then optimal decision at c is to go to f



Notation

α is the current state (intersection).

u_i is an allowable decision (control) elected at the state α. In this example i can assume one or more of the values 1, 2, 3, 4, corresponding to the headings N, E, S, W.

x_i is the state (intersection) adjacent to α which is reached by application of u_i at α.

h is the final state.

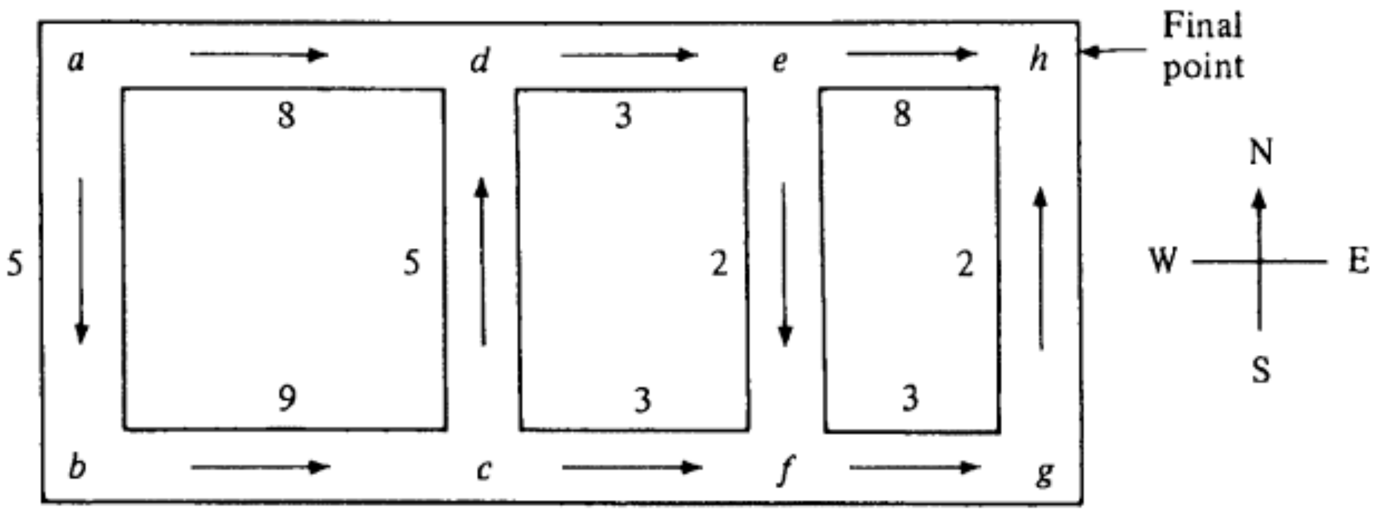
$J_{\alpha x_i}$ is the cost to move from α to x_i.

$J^*_{x_i h}$ is the *minimum cost* to reach the final state h from x_i.

$C^*_{\alpha x_i h}$ is the minimum cost to go from α to h *via* x_i.

$J^*_{\alpha h}$ is the minimum cost to go from α to h (by any allowable path).

$u^*(\alpha)$ is the optimal decision (control) at α.



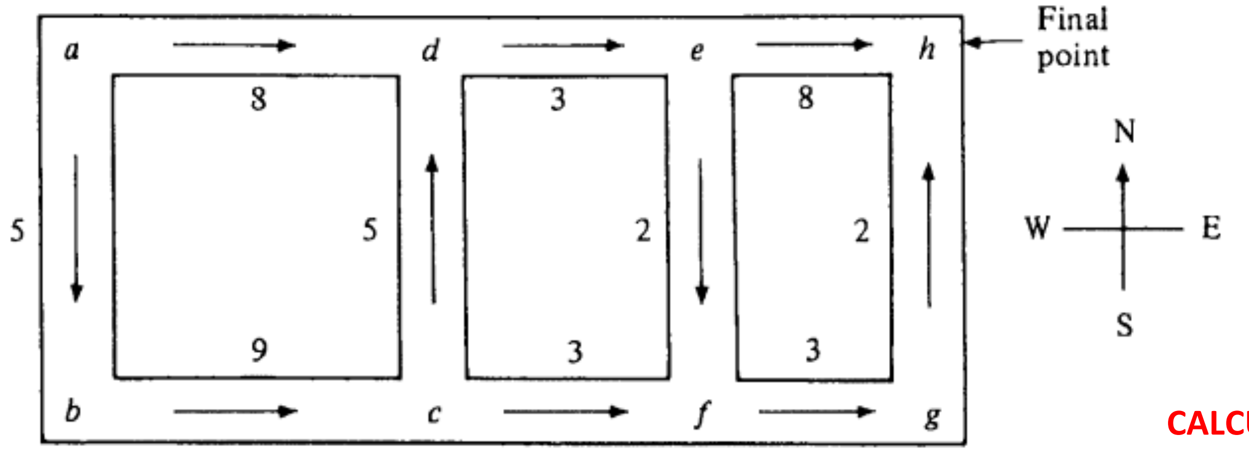
When this notation is used, the principle of optimality implies that

$$C^*_{\alpha x_i h} = J_{\alpha x_i} + J^*_{x_i h}$$

And the optimal decision is the decision that leads to

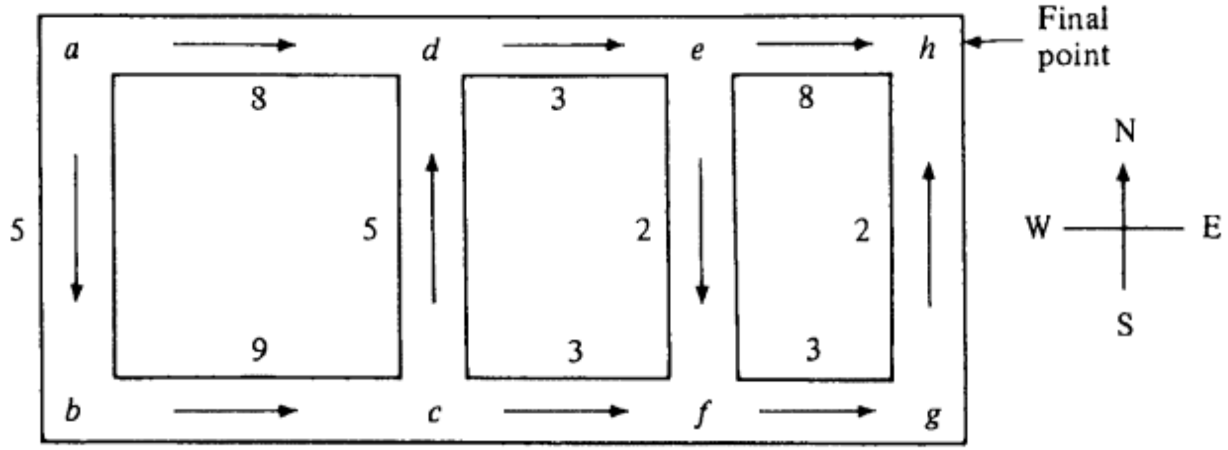
$$J^*_{\alpha h} = \min\{C^*_{\alpha x_1 h}, C^*_{\alpha x_2 h}, \ldots, C^*_{\alpha x_i h}, \ldots\}.$$

These two equations define the algorithm called dynamic programming.



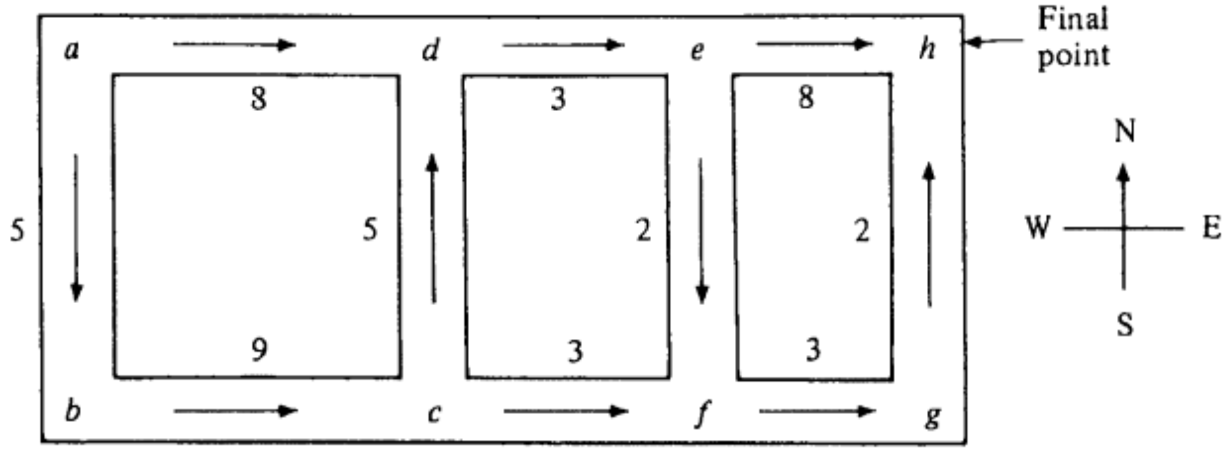
CALCULATION OF OPTIMAL HEADINGS BY DYNAMIC PROGRAMMING

Current intersection α	*Heading* u_i	*Next intersection* x_i	*Minimum cost from α to h via x_i* $J_{\alpha x_i} + J^*_{x_i h} = C^*_{\alpha x_i h}$	*Minimum cost to reach h from α* $J^*_{\alpha h}$	*Optimal heading at α* $u^*(\alpha)$
g	N	h	$2 + 0 = 2$	2	N
f	E	g	$3 + 2 = 5$	5	E
e	E	h	$8 + 0 = 8$		
	S	f	$2 + 5 = 7$	7	S
d	E	e	$3 + 7 = 10$	10	E
c	N	d	$5 + 10 = 15$		
	E	f	$3 + 5 = 8$	8	E
b	E	c	$9 + 8 = 17$	17	E
a	E	d	$8 + 10 = 18$	18	E
	S	b	$5 + 17 = 22$		

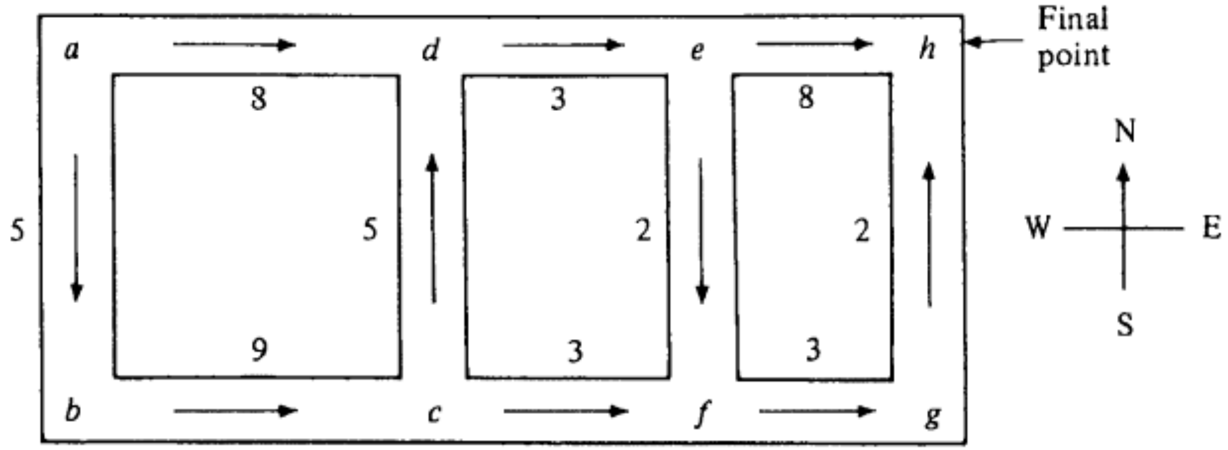


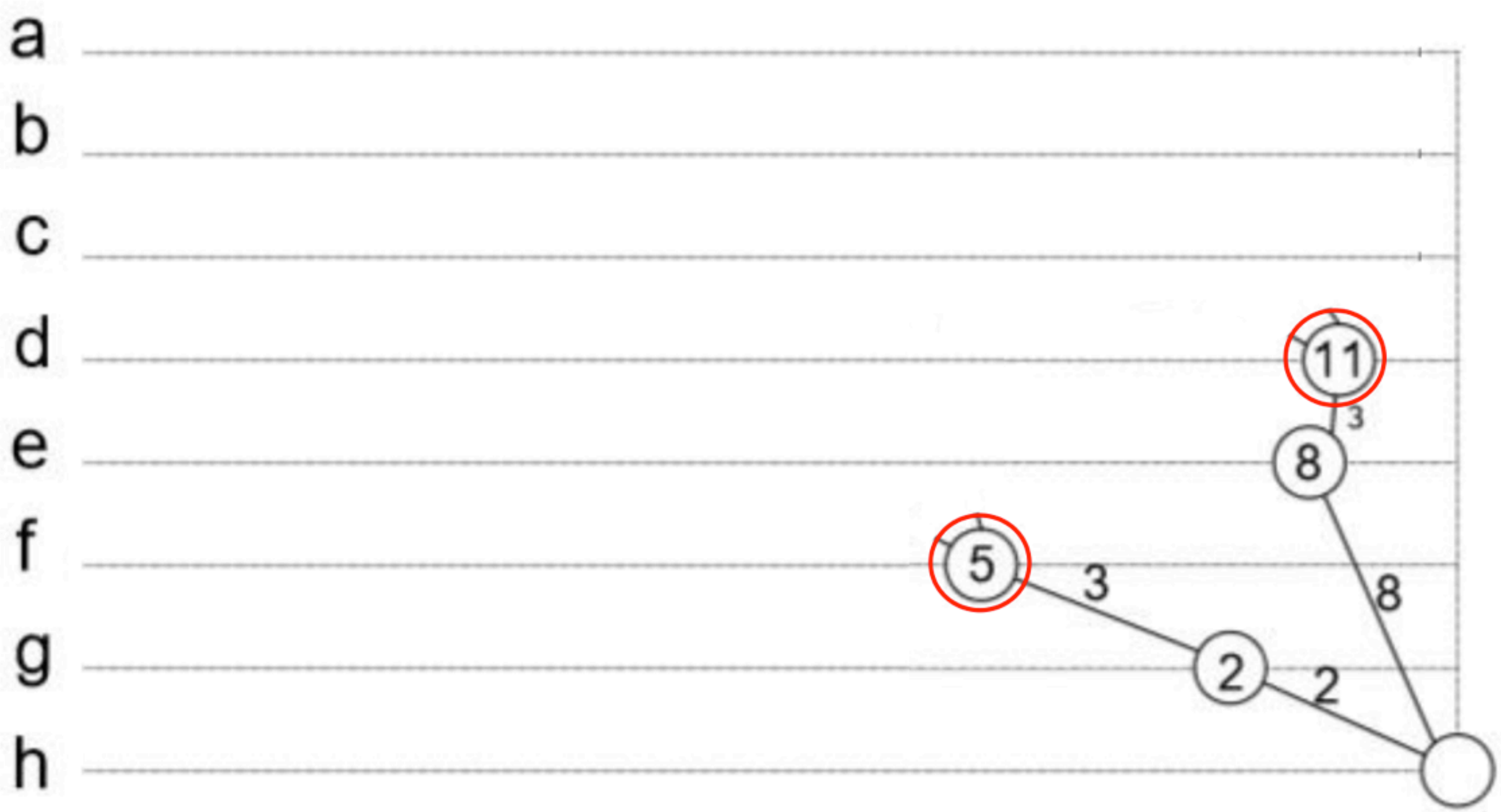
a
d
e
h
Final point
8
3
8
5
5
2
2
9
3
3
b
c
f
g
N
W
E
S

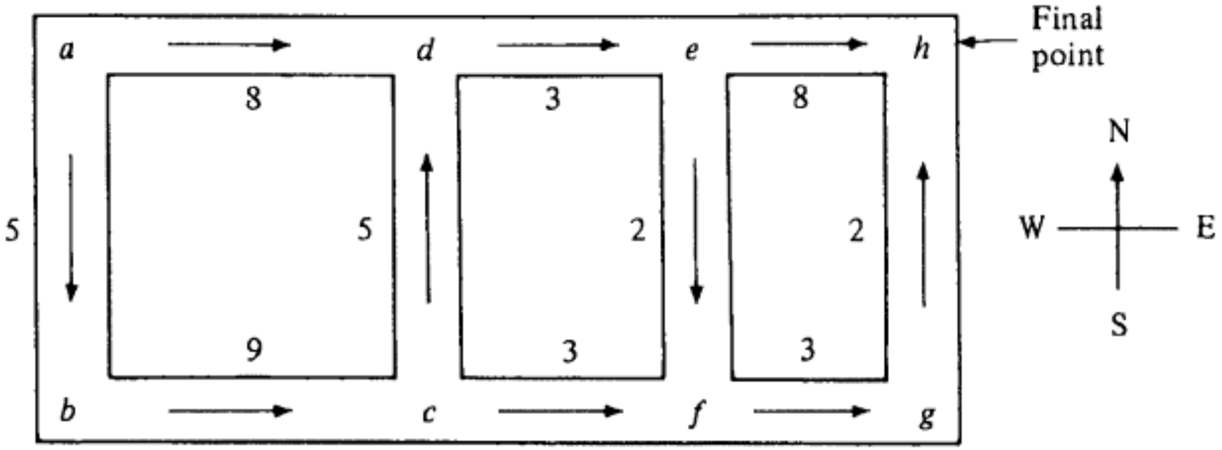
a
b
c
d
e
f
g
h

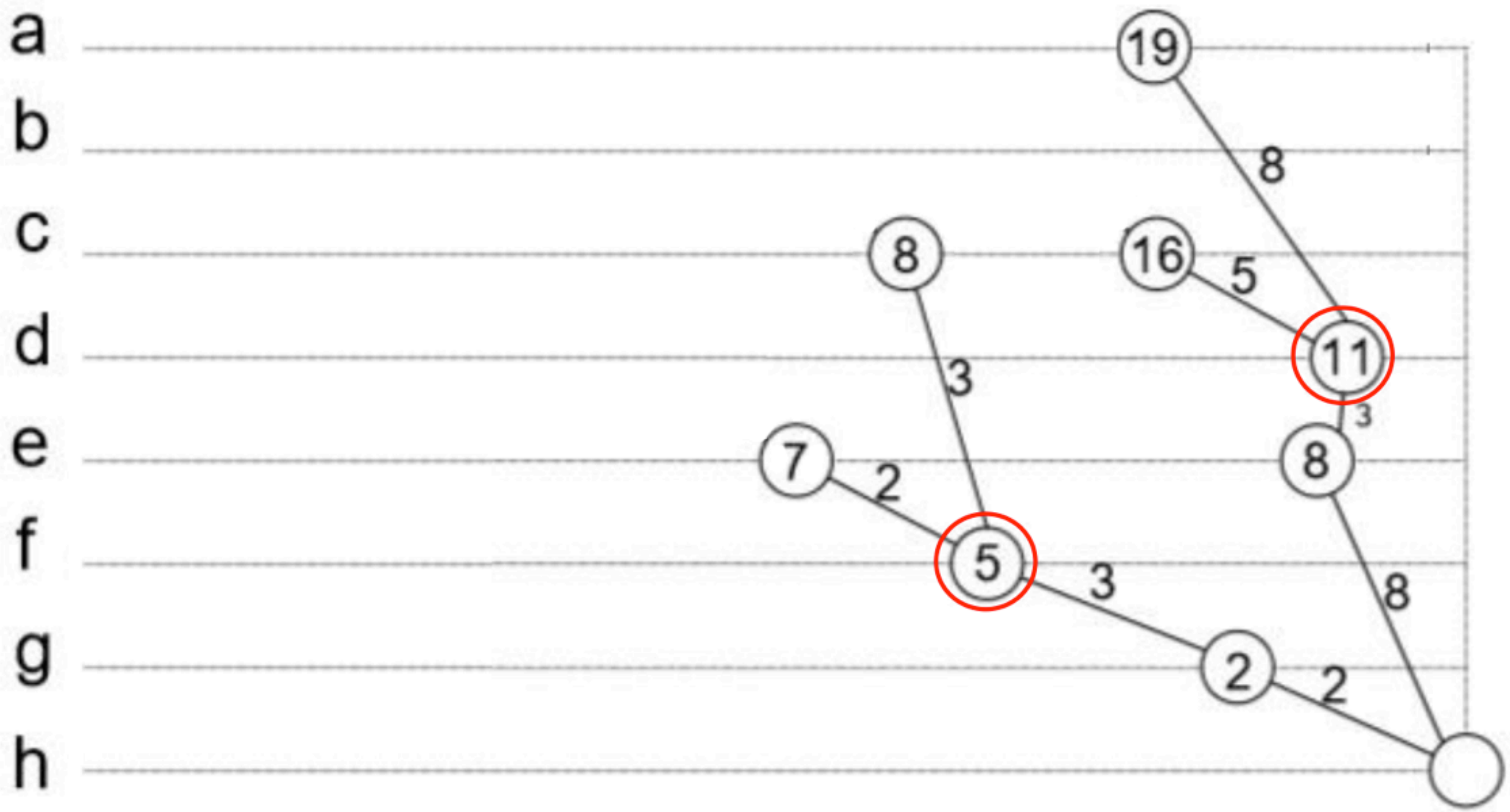

a
d
e
h
Final point
8
3
8
N
W
E
S
5
5
2
2
9
3
3
b
c
f
g

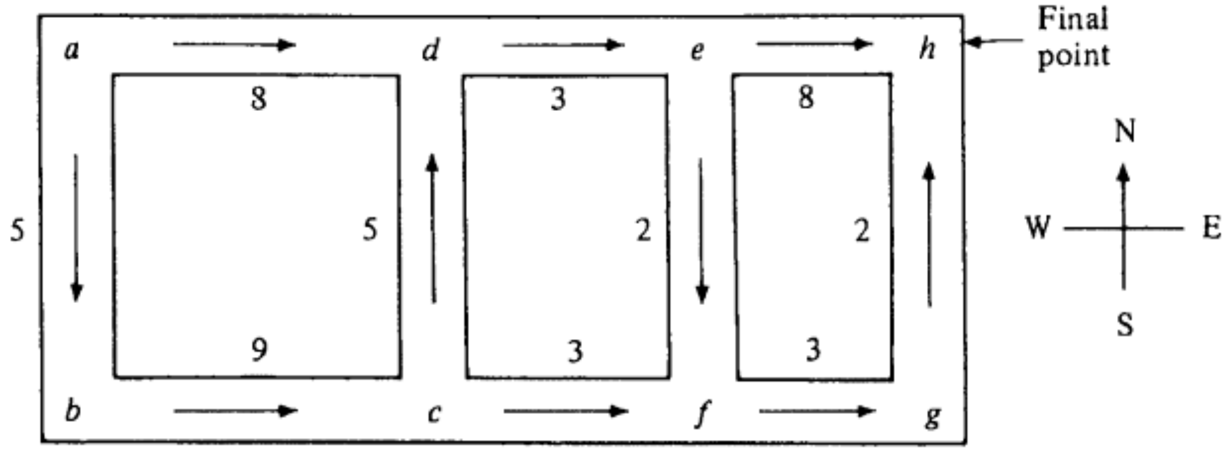

a
b
c
d
e
f
g
h
8
8
2
2


a
d
e
h
Final point
8
3
8
N
W
E
S
5
5
2
2
9
3
3
b
c
f
g

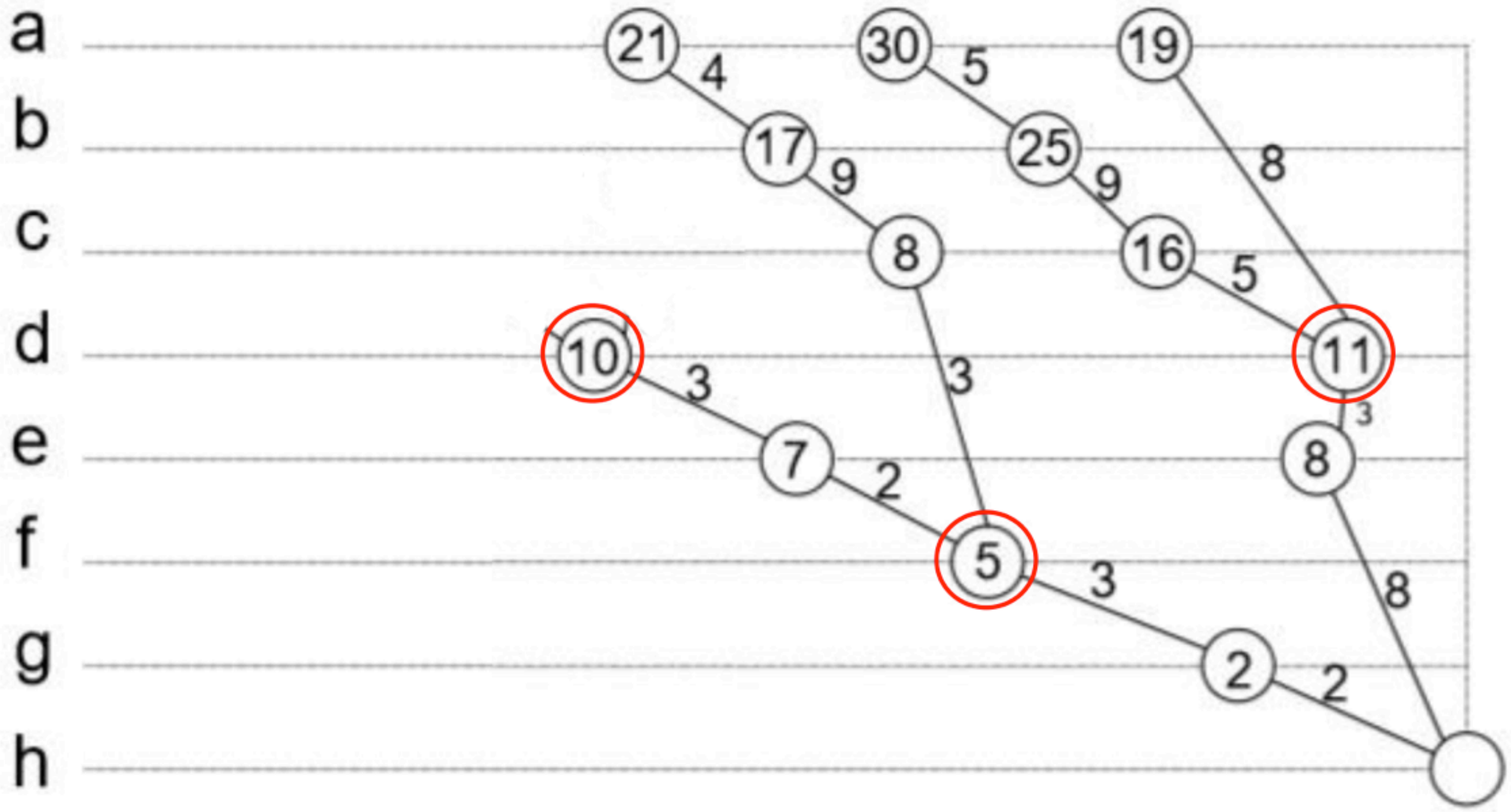

a
b
c
d
e
f
g
h
11
3
8
5
3
8
2
2


a
d
e
h
Final point
8
3
8
5
5
2
2
9
3
3
b
c
f
g
N
W
E
S

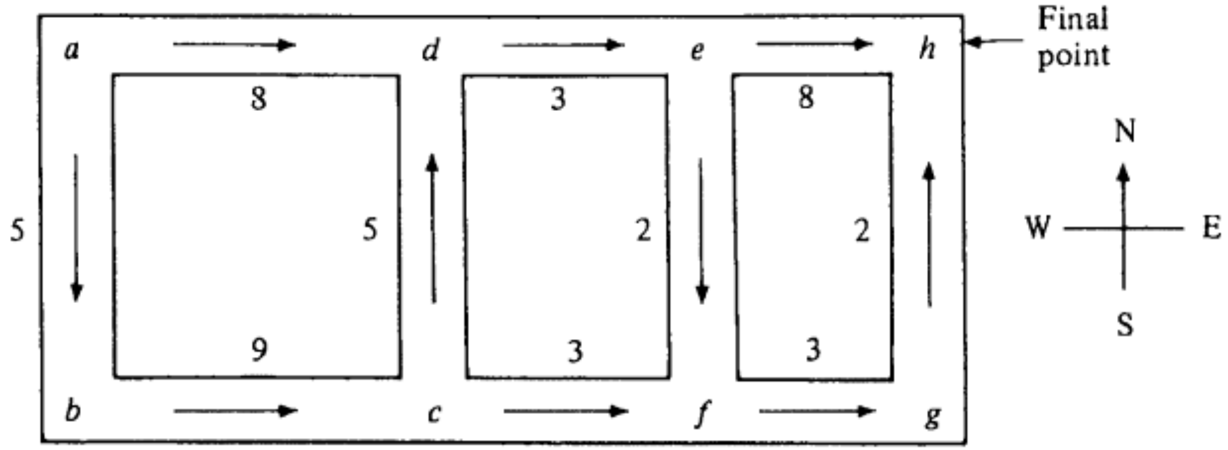

a
b
c
d
e
f
g
h
19
8
8
16
5
11
3
3
7
2
8
5
3
8
2
2

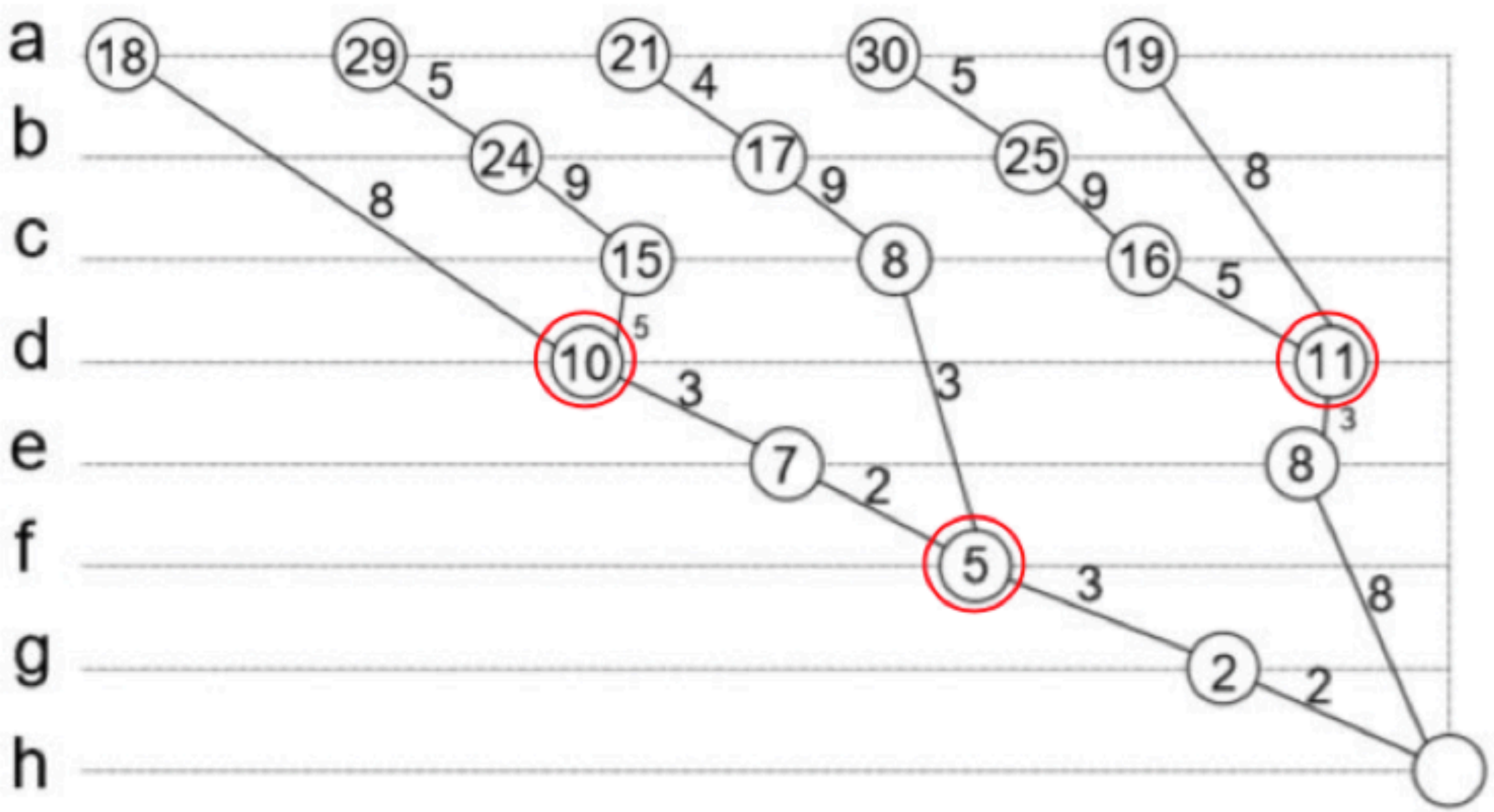


a
d
e
h
Final point
8
3
8
5
5
2
2
9
3
3
b
c
f
g
N
W
E
S



a
b
c
d
e
f
g
h
21
30
19
4
5
17
25
9
9
8
8
16
5
10
11
3
3
3
7
8
2
5
3
8
2
2


a
d
e
h
Final point
8
3
8
5
5
2
2
9
3
3
b
c
f
g
N
W
E
S


a
b
c
d
e
f
g
h
18
29
21
30
19
24
17
25
15
8
16
10
11
7
8
5
2
5
4
5
9
9
9
8
8
5
5
3
3
3
2
3
8
2

End of Lecture X