Introducción a Programación Dinámica: Parte 2 - Fundamentos

Roger Z. Ríos

Programa de Posgrado en Ing. de Sistemas
Facultad de Ing. Mecánica y Eléctrica
Universidad Autónoma de Nuevo León

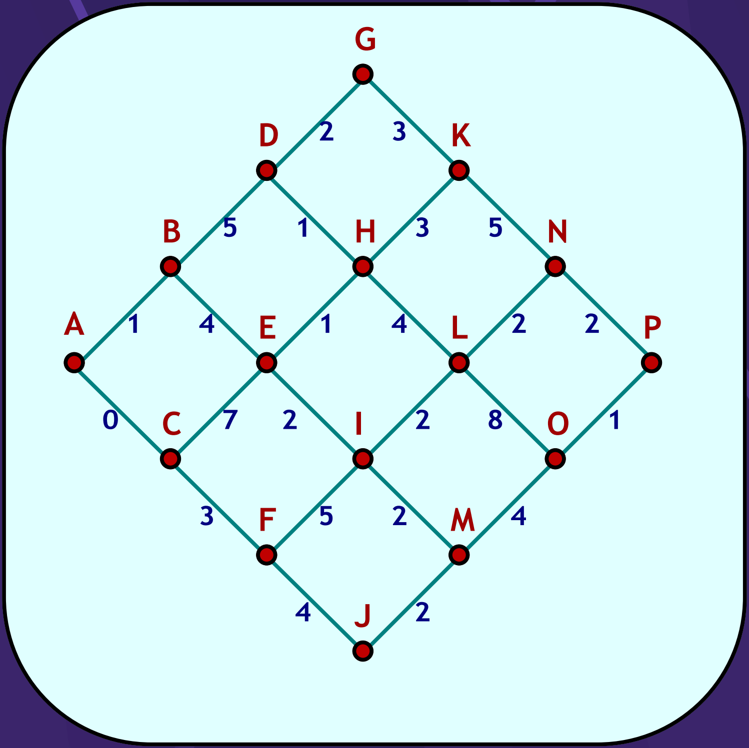
http://yalma.fime.uanl.mx/~roger

Congreso Anual de la Sociedad Matemática Mexicana
F C F M - U A N L
Cd. Universitaria 24-25 Octubre 2019

Agenda

- A simple path problem
- How to solve it?
  - Enumeration
  - Heuristics
  - Dynamic programming
- Computational efficiency of DP
- A coupon & turn problem
- An equipment replacement problem
- Closing remarks

A Simple Path Problem

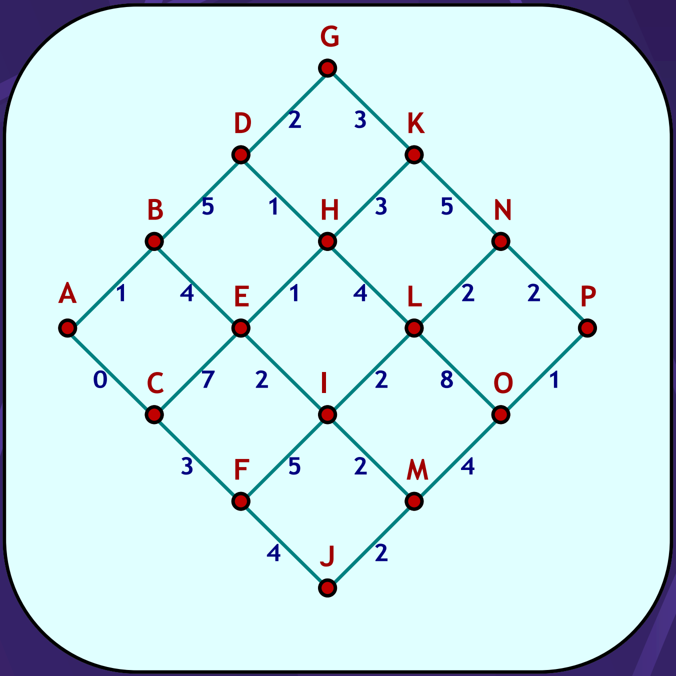


Find shortest path from A to P

How to solve it?

- Enumeration
- Heuristics
- Dynamic Programming

Enumeration



Problem with 6 stages

- Find all possible paths (20)
- or C(6,3) = 6! / 3! 3!
- For each path (5 sums)
- Comparisons (19)
- **FLOPS = 119**

Problem with N stages

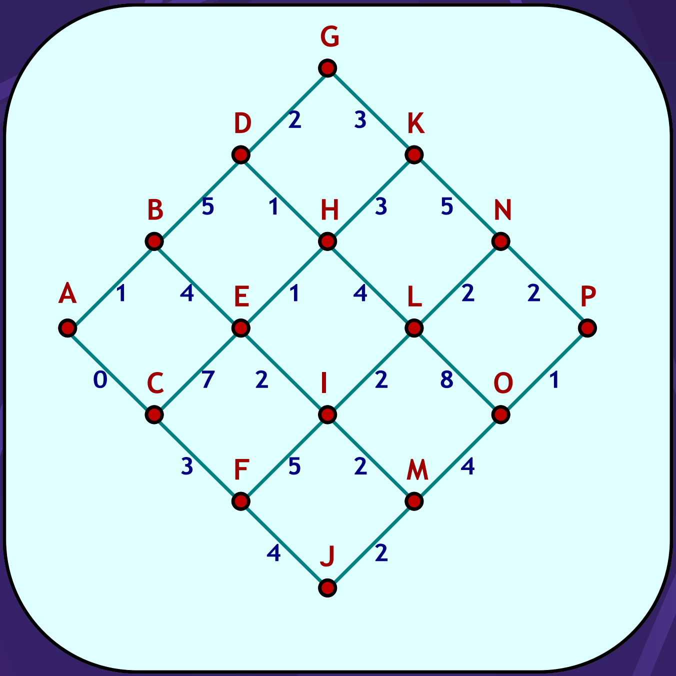
- # of paths = C(n, n/2)
- For each path (n-1 sums)
- # Comparisons = C(n, n/2) - 1
- **FLOPS = nC(n, n/2) - 1**

Enumeration

Computational Effort

N	FLOPS
6	119
20	$> 310^{6}$
100	$> 10^{30}$

Heuristics



Greedy

- Take "best" decision at each stage
- Path found:

  A → C → F → J → M → O → P
- Cost: 14 (optimal??)

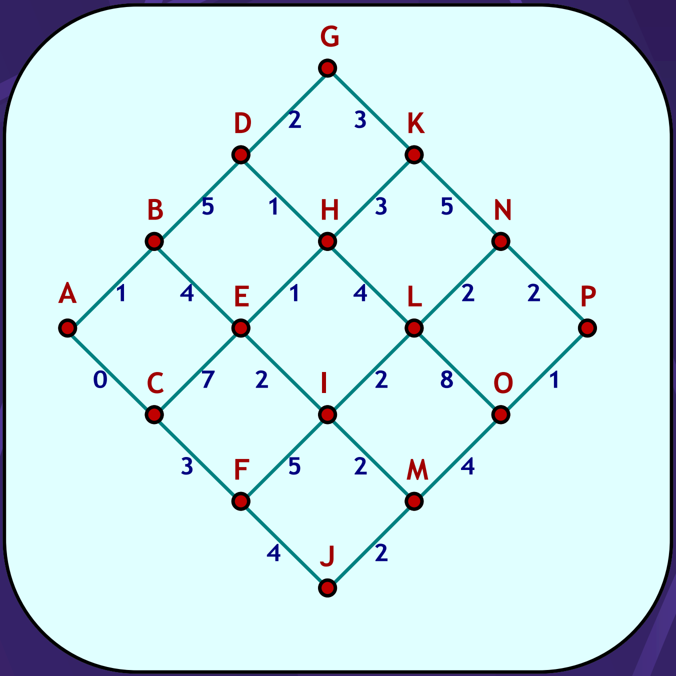
- Effort: 6 comparisons, 5 adds
- Effort(N): N comparisons, N-1 adds
- Optimality is not guaranteed

Heuristics

FLOPS

N	Enumeration	Heuristic
6	119	11
20	> 310^6	39
100	> 10^{30}	199
Optimal	Yes	“No”

Dynamic Programming



g(i) = least cost from i to P

Stage 0

$$g(A) = \min \begin{cases} \text{U}: 1 + g(B) \\ \text{D}: 0 + g(C) \end{cases}$$

Stage 1

$$g(B) = \min \begin{cases} \text{U}: 5 + g(D) \\ \text{D}: 4 + g(E) \end{cases} \qquad g(C) = \min \begin{cases} \text{U}: 7 + g(E) \\ \text{D}: 3 + g(F) \end{cases}$$

Stage …

Stage 5

$$g(N) = \begin{cases} \\ \text{D}: 2 + g(P) \end{cases} \qquad g(O) = \begin{cases} \text{U}: 1 + g(P) \\ \end{cases}$$

g(P)=0

Dynamic Programming

- Solve backward with “boundary condition” g(P)=0
- p(i) := Optimal decision taken at node i

Dynamic Programming

Start with g(P) = 0

Stage 5

g(O) = min{ U: 1 + g(P) } = 1
p(O) = U

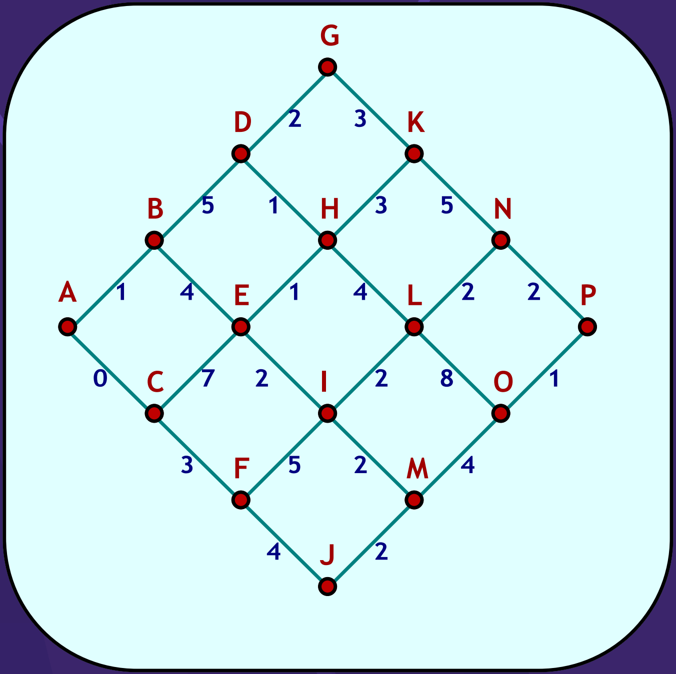
g(N) = min{ D: 2 + g(P) } = 2
p(N) = D

Stage 4

g(M) = min{ U: 4 + g(O) } = 5
p(M) = U

g(L) = min{ U: 2 + g(N), D: 8 + g(O) } = 4
p(L) = U

g(K) = min{ D: 5 + g(N) } = 7
p(K) = D



Dynamic Programming

Optimal Solution

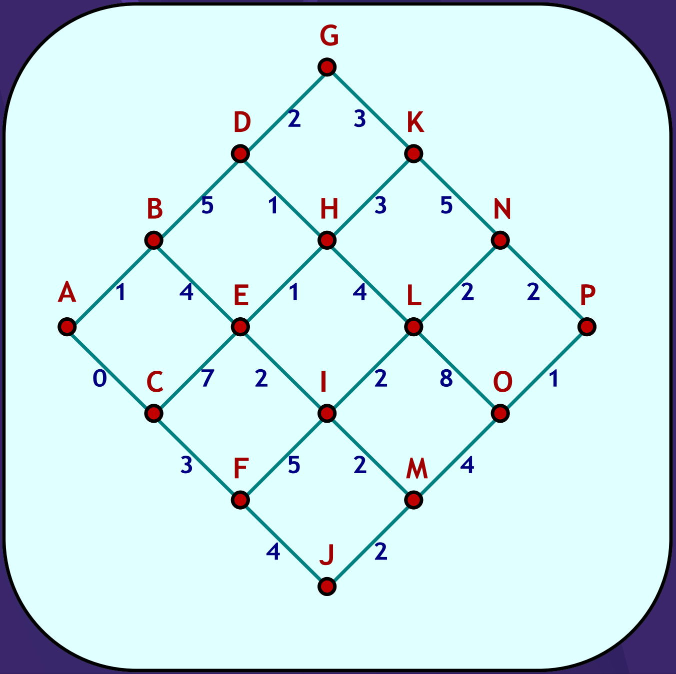
Node i	g(i)	p(i)
A	13	U → B
B	12	D → E
C	14	D → F
D	9	D → H
E	8	D → I
F	11	U → I
G	10	D → K
H	8	D → L
I	6	U → L
J	7	U → M
K	7	D → N
L	4	U → N
M	5	U → O
N	2	D → P
O	1	U → P
P	0	

Optimal Path
A→B→E→I→L→N→P

Cost: 13

Computational Efficiency

- N nodes with 1 branch (1 add)
- $(N/2)^2$ nodes with 2 branches (2 adds + 1 comp)
- FLOPS = $3(N/2)^2 + N$

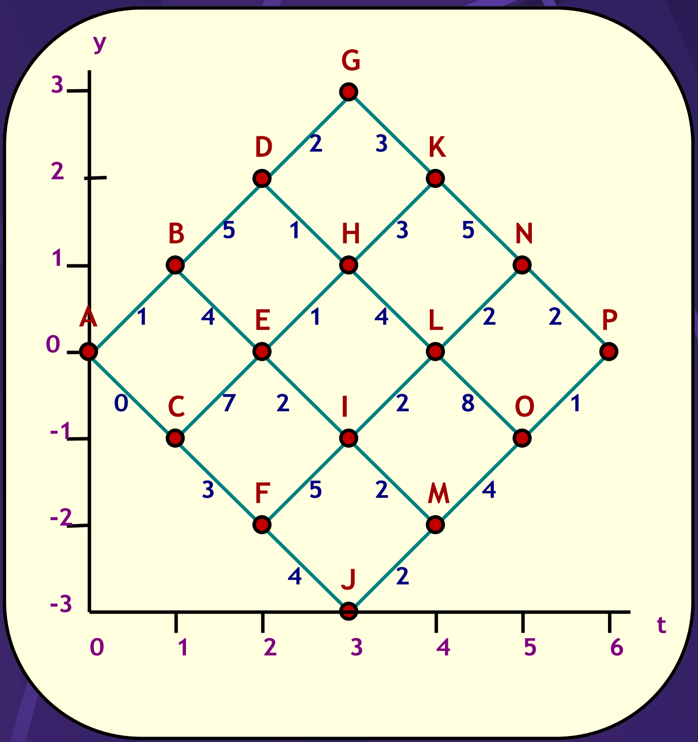


Computational Efficiency

FLOPS

N	Enumeration	Heuristic	DP
6	119	11	33
20	$> 310^6$	39	320
100	$> 10^{30}$	199	7600
Optimal	Yes	“No”	Yes

Terminology



- $g_t(y)$:= optimal value function
  [min cost from (t,y) to (N,0)]
- t := stage variable (0,1,…,6)
- y := state vector (-3,-2,…,2,3)
- Data: CU(t,y), CD(t,y) := cost of "U", "D" decision at (t,y)
- Recurrence relation

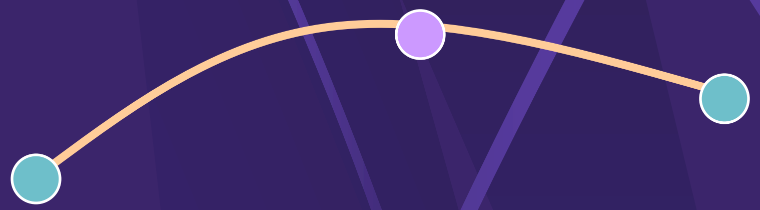
$$g_t(y) = \min \begin{cases} \mathrm{CU}(t,y) + g_{t+1}(y+1) \\ \mathrm{CD}(t,y) + g_{t+1}(y-1) \end{cases}$$

- Boundary condition $g_N(0) = 0$
- $p_t(y)$:= optimal decision at (t,y)

Terminology

- **Principle of Optimality [Richard Bellman]**

  "Any subpolicy of an optimal policy must be optimal"

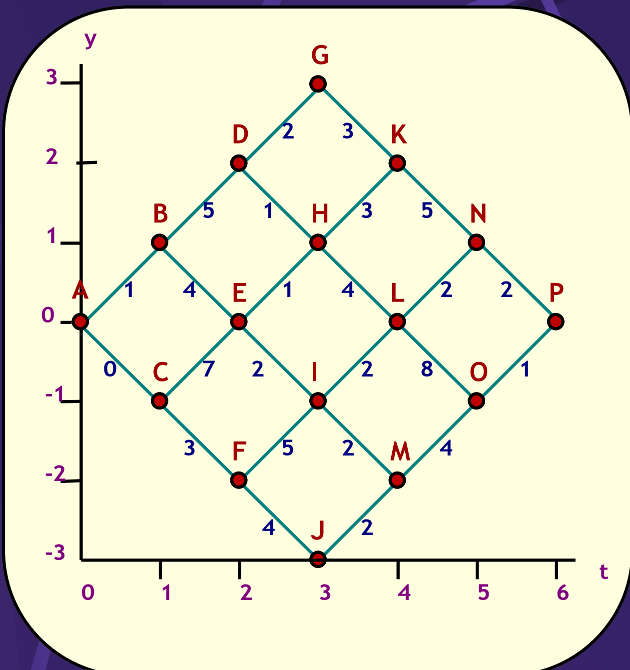


- **Consultant Question**

  "What do I have to know in order to take optimal decisions from now on?"

  Min info required → state variables

Example B: A Coupon & Turn Problem



- Wish to find a min cost path from (0,0) to (6,0)
- Every time you make a "turn" you pay $2
- You are given 2 coupons for "free" arcs

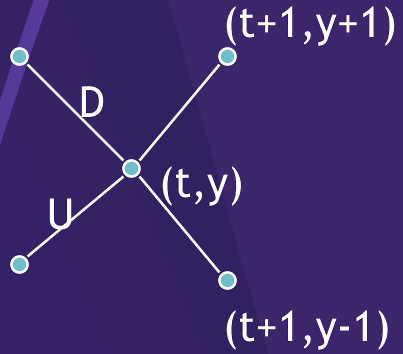
DP Formulation:

- $g_t(y,x,z)$:= Min cost effort from node (t,y) to (N,0) when arrival to (t,y) comes from a "x" direction (x={U,D}) and with z coupons left
- Optimal solution $g_0(0,-,2)$
- Boundary condition $g_N(0,x,z) = 0$ for any x, z

Example B

DP Formulation:

- Recurrence relationship



$$g_t(y,U,z)=\min\begin{cases}\mathrm{CU}(t,y)+g_{t+1}(y+1,U,z) & \text{U, don't use coupon}\\ g_{t+1}(y+1,U,z-1) & \text{U, use coupon}\\ \mathrm{CD}(t,y)+2+g_{t+1}(y-1,D,z) & \text{D, don't use coupon}\\ 2+g_{t+1}(y-1,D,z-1) & \text{D, use coupon}\end{cases}$$

$$g_t(y,D,z)=\min\begin{cases}\mathrm{CU}(t,y)+2+g_{t+1}(y+1,U,z) & \text{U, don't use coupon}\\ 2+g_{t+1}(y+1,U,z-1) & \text{U, use coupon}\\ \mathrm{CD}(t,y)+g_{t+1}(y-1,D,z) & \text{D, don't use coupon}\\ g_{t+1}(y-1,D,z-1) & \text{D, use coupon}\end{cases}$$

Example C: Equipment Replacement

- Own a "machine" which deteriorates with age
- Must own machine during next N years
- At start of year 1, age of incumbent machine is y
- Decision at start of each year is either to keep machine or replace it

- ***c(i)*** := yearly cost of operating machine of age i
- ***p*** := price of a new machine (age 0)
- ***t(i)*** := trade-in amount received for old machine of age i
- ***s(i)*** := salvage value received at end of year N for a machine of age i

- **Must find an optimal replacement policy which minimizes the total cost during the next N years**

Example C: Equipment Replacement

DP Formulation

- Optimal value function:

  $J_t(x)$:= min cost of owning a machine from year t to N, starting year t with a machine of age x

- Recurrence relation:

$$J_t(x) = \min \begin{cases} p - t(x) + c(0) + J_{t+1}(1) & \text{(replace)} \\ c(x) + J_{t+1}(x+1) & \text{(keep)} \end{cases}$$

- Boundary condition:

$$J_{N+1}(x) = -s(x)$$

- Optimal solution:

$$J_1(y)$$

Closing Remarks

- DP applies in “sequential” decisions (deterministic or stochastic)
- Very efficient solution procedure (backward & forward formulations)
- Applications:
  - Equipment replacement
  - Resource allocation
  - Pipeline network systems
  - Inventory systems
  - Control systems

References

- **R. Bellman** (1957). *Dynamic Programming.* Princeton University Press, Princeton, EUA, 1957
- **U. Bertelè and F. Brioschi** (1972). *Nonserial Dynamic Programming.* Academic Press, New York, EUA.
- **D. P. Bertsekas** (1987). *Dynamic Programming: Deterministic and Stochastic Models.* Prentice-Hall, Englewood Cliffs, EUA.
- **S. E. Dreyfus and A. M. Law** (1977). *The Art and Theory of Dynamic Programming.* Academic Press, Orlando, EUA.
- **S. Ross** (1995). *Introduction to Stochastic Dynamic Programming.* Academic Press, New York, EUA.

Questions?

http://yalma.fime.uanl.mx/~roger

roger@yalma.fime.uanl.mx