Today

- Resource allocation
- Travelling salesperson problem

Today's material is covered in:

- Chapter 3 of Dreyfus and Law

Resource Allocation Problem

The problem of determining the allocation of resources that maximizes total benefit subject to the limited resource availability is

$$\max \sum_{t=1}^{t=T} r_t(a_t)$$

$$\text{s.t.} \sum_{t=1}^{t=T} f_t(a_t) \leq w$$

where a_t, the amount of activity t, must be a member of {0,1,2,…}.

- $r_t(a_t)$ is the reward obtained for choosing activity t at level a_t
- $f_t(a_t)$ is the utilization of activity t given level a_t

Example: Resource-Allocation

To use linear programming three assumptions must hold:

- **Assumption 1** : The amount of a resource assigned to an activity may be any non negative number
- **Assumption 2** : The benefit obtained from each activity is proportional to the resource assigned to the activity
- **Assumption 3**: The total utilization for each activity is proportional to the resources assigned to the activity

Even if assumptions 1, 2 and 3 do not hold, DP can be used to solve resource-allocation problems efficiently

DP Formulation

Define $\mathrm{v_t}(\mathrm{s_t})$ to be the maximum reward that can be obtained from activities $t, t+1, \ldots, T$ if $\mathrm{s_t}$ units of the resource may be allocated to remaining activities $t, t+1, \ldots, T$.

We can generalize the recursions to this situation by writing

$$v_t(s_t) = \max_{a_t \in A_t(s_t)} \{\mathrm{r_t(a_t)} + \mathrm{v_{t+1}}\big(\mathrm{s_t} - \mathrm{f_t(a_t)}\big)\}, \quad \forall s_t$$

$$v_T(s_T) = 0, \forall s_T$$

where a_t must be a non-negative integer satisfying:

$$f_t(a_t) \leq s_t$$

Example: Integer Knapsack Problem

Suppose a knapsack with capacity 10lb is to be filled with the following items:

Item 1: 4 lbs, Benefit 11

Item 2: 3 lbs, Benefit 7

Item 3: 5 lbs, Benefit 12

Therefore

$$r_1(a_1) = 11a_1, \quad r_2(a_2) = 7a_2, \quad r_3(a_3) = 12a_3 ,$$

$$f_1(a_1) = 4a_1, \quad f_2(a_2) = 3a_2, \quad f_3(a_3) = 5a_3$$

Find the optimal items to select using a DP

Example: Knapsack Problem

Let $v_t(s_t)$ be the maximum benefit that can be earned for item t given an s_t-pound knapsack:

$v_3(s_3) = \max\{12a_3\}$ where $5a_3 \leq s_3$ and a_3 is an integer

It follows that in stage 3: $v_3(10) = 24, v_3(5) = v_3(6) = v_3(7) = v_3(8) = v_3(9) = 12, v_3(0) = v_3(1) = v_3(2) = v_3(3) = v_3(4) = 0$

Optimal decisions in stage 2 are governed by:

$v_2(s_2) = \max\{7a_2 + v_3(s_2 - 3a_2)\}$ where a_2 is an integer satisfying $3a_2 \leq s_2$

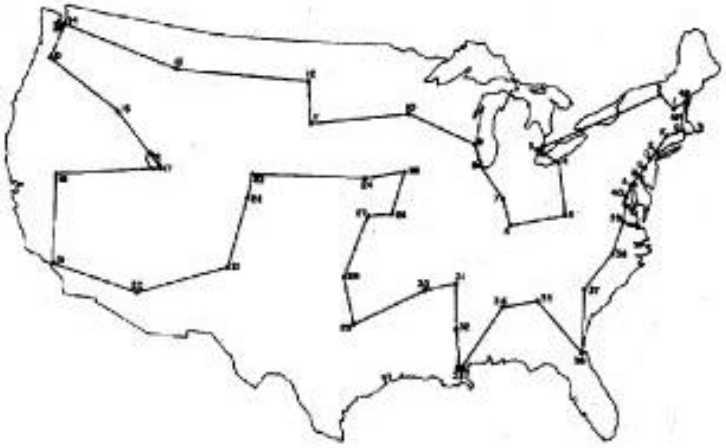
Optimal decisions in stage 1 are governed by:

$v_1(s_1) = \max\{11a_1 + v_2(s_1 - 4a_1)\}$ where a_3 is an integer satisfying $4a_1 \leq s_1$

**See Dreyfus and Law , Chapter 3, for another fully worked example

TSP as a Dynamic Program

1932: First recorded description of the “Traveling Salesman Problem”



Problem Description: Find the shortest route that visits each location once and returns to the starting point.

Learn more about the TSP at: http://www.tsp.gatech.edu/index.html

TSP as a Dynamic Program

DP Formulation

Let $N_j = \{2,3,\dots,j-1,j+1,\dots,N\}$ and let S_i be a subset of N_j with i members

States: defined by current location, j , and subset of nodes already visited, S_i

Action: location to arrive from

Optimality equations:

Distance from a_i to j

Shortest distance from node 1 to a_i given nodes in set $\{S_i - a_i\}$ visited in between.

$$v_i(j,S_i) = \min_{a_i \in S_i}\left\{d_{a_i j} + v_{i-1}(a_i, S_i - a_i)\right\}, \quad \forall j, \forall S_i \subset N_j$$

$$v_0(j,\emptyset) = d_{1j}, \forall j$$

boundary condition

Example: TSP

Solve the following TSP, where location 1 is the starting point.

Distance Matrix:

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

TSP Computational Analysis

At stage i, $v_i(j,S)$ must be evaluated for $(N-1)\binom{N-2}{i}$ different j,S pairs.

Each evaluation requires i additions and $i-1$ comparisons

Total operations is equal to:

$$(N-1)(N-2)2^{N-3} + (N-1)(N-4)2^{N-3} \approx O(N^2 2^n)$$

For a 20 city problem a total of approximately 85 million operations are required!