Lecture 25: Dynamic Programming II

2023/10/18

詹博华（中国科学院软件研究所）

Dynamic programming: basic principles

1. Find subproblems of the original problems to be solved.
2. Find recurrence relations between solutions of the subproblems.
3. Compute answers to the subproblems in a certain order.

Review: rod cutting

- Cut a rod of length n for the best total price.
- Subproblems: for any $m \leq n$, cut a rod of length m for the best total price.
- Recurrence relation:

$$r_n = \max_{1 \leq i \leq n} (p_i + r_{n-i}) \ .$$

Location of first cut   Price of first cut   Price of remaining cuts

- Order of computation: in increasing order of m.

Rod cutting: example of computation

- Given the following prices:

i	0	1	2	3	4	5	6	7	8
p_i	0	2	5	6	8	11	12	15	21

- Compute the best price r_m and first cut s_m for each m.

i	0	1	2	3	4	5	6	7	8
r_i	0	2	5	7	10	12	15	17	21
s_i	0	1	2	2	2	2	2	2	8

Review: matrix-chain multiplication

- Find the way to compute the product $A_1 A_2 \cdots A_n$ using the least number of scalar multiplications.
- Subproblems: for any $1 \leq i < j \leq n$, how to compute the product $A_i A_{i+1} \cdots A_j$ using the least number of scalar multiplications.
- Recurrence relation:

$$m[i,j] = \begin{cases} 0 & \text{if } i = j \,, \\ \underbrace{\min_{i \leq k < j}}_{\text{Dividing point } k} \{\underbrace{m[i,k]}_{\text{Cost of computing } A_{ik}} + \underbrace{m[k+1,j]}_{\text{Cost of computing } A_{k+1,j}} + \underbrace{p_{i-1} p_k p_j}_{\text{Cost of multiplying } A_{ik} \text{ and } A_{k+1,j}}\} & \text{if } i < j \,. \end{cases}$$

- Order of computation: in increasing order of $j - i$.

Dynamic programming for counting

- Dynamic programming can be used not only for optimization, but also for counting.
- Subproblems are counts for smaller versions of the problem.
- Recurrence relations compute counts for larger problems from counts for smaller problems.
- Consider the counting versions of the previous problems.

Rod cutting: counting version

- How many ways to cut a rod of length n? (consider $3 = 1 + 2$ and $3 = 2 + 1$ as different).
- Subproblems: for each $m \leq n$, count number of ways to cut the rod (written as $c[m]$).
- Recurrence relation:

$$c[m] = c[m-1] + c[m-2] + \cdots + c[1] + 1 = 1 + \sum_{i=1}^{m-1} c[i]$$

First cut is 1 First cut is 2 $\cdots$ First cut is $m-1$ Take entire rod

Rod cutting: counting results

- $c[1] = 1$
- $c[2] = 1 + c[1] = 2$
- $c[3] = 1 + c[1] + c[2] = 4$
- $c[4] = 1 + c[1] + c[2] + c[3] = 8$
- $c[5] = 1 + c[1] + c[2] + c[3] + c[4] = 16$
- In general $c[i] = 2^{i-1}$.
- The problem of counting the number of partitions, where $3 = 2 + 1$ and $3 = 1 + 2$ are considered the same, is more difficult.

Partition function: https://mathworld.wolfram.com/PartitionFunctionP.html.

Matrix-chain multiplication: counting version

- How many ways to add parenthesis to determine the order of multiplication of n matrices?
- Subproblems: how many ways to multiply m matrices, for each $m \leq n$ (written as $C[n]$).
- Recurrence relation:

$$C[n] = C[1]C[n-1] + C[2]C[n-2] + \cdots + C[n-1][1] = \sum_{i=1}^{n-1} C[i]C[n-i]$$

Divide into $A_1(A_2 \cdots A_n)$

Divide into $(A_1A_2)(A_3 \cdots A_n)$

…

Divide into $(A_1 \cdots A_{n-1})A_n$

Matrix-chain multiplication: counting results

- $C[1] = 1$
- $C[2] = C[1]C[1] = 1$
- $C[3] = C[1]C[2] + C[2]C[1] = 2$
- $C[4] = C[1]C[3] + C[2]C[2] + C[3]C[1] = 5$
- $C[5] = C[1]C[4] + C[2]C[3] + C[3]C[2] + C[4]C[1] = 14$
- $C[6] = C[1]C[5] + C[2]C[4] + C[3]C[3] + \cdots = 4 + 2 \cdot (14 + 5) = 42$
- The general formula is:

$$C[n] = \frac{1}{n}\binom{2(n-1)}{n-1}$$

Catalan number: https://mathworld.wolfram.com/CatalanNumber.html

Coin-changing problem

- Given a set of coin values $p_1, p_2, \dots, p_n$, and amount n, find the way to convert n into the least number of coins.
- For example: the coin values are $1, 4$ and 9.
- For $n = 15$, it is best to use the largest possible value at each time: $15 = 9 + 4 + 1 + 1$, changing into 4 coins.
- For $n = 12$, it is incorrect to use the largest possible value at each time: $12 = 9 + 1 + 1 + 1$, but also $12 = 4 + 4 + 4$. This shows the *greedy method* does not always work.

Dynamic programming solution

- Find the least number of coins to change a value n.
- Subproblems: for each $m \leq n$, find the least number of coins to change m. Write as $a[m]$.
- Recurrence relation:

$$a[m] = 1 + \min(a[m - p_1], a[m - p_2], \dots, a[m - p_k])$$

First coin

Remainder after choosing p_1.

Remainder after choosing p_2.

…

Remainder after choosing p_k.

- Order of computation: in increasing order of m.

Example

- Consider the coins $p_1 = 1, p_2 = 4, p_3 = 9$.
- Compute for increasing values of n:

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$a[n]$	1	2	3	1	2	3	4	2	1	2	3	3	2	3	4	4	3	2	3	4

Coin-changing problem: counting version

- Given a set of coin values $p_1, p_2, \ldots, p_n$, and amount n, find the number ways to change n into coins.
- For example: given $p_1 = 1, p_2 = 4, p_3 = 9$.
- Let $n = 9$. Then there are 4 ways to change n:
  - $9 = 9$
  - $9 = 4 + 4 + 1$
  - $9 = 4 + 1 + 1 + 1 + 1 + 1$
  - $9 = 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1$

Counting problem: first try

- Subproblems: for any $m \leq n$, find number of ways to change m into coins. Write as $C[m]$.
- Recurrence relation:

$$C[0] = 0$$

$$C[m] = C[m - p_1] + C[m - p_2] + \cdots + C[m - p_k]$$

Use p_1 Use p_2 $\cdots$ Use p_k

- Result:

m	0	1	2	3	4	5	6	7	8	9
$C[m]$	1	1	1	1	2	3	4	5	7	10

- **What went wrong?** Double counted different orders, e.g. $9 = 4 + 4 + 1$ and $9 = 4 + 1 + 4$.

Counting problem: correct solution

- Subproblems: for each $m \leq n$ and $i \leq k$, count the number of ways to change m into coins, using only coins of values $p_1, \ldots, p_i$.
- Write this as $C[m, i]$.
- Recurrence relation:

$$C[0,0] = 1, \qquad C[m, 0] = 0 \ (m > 0)$$

$$C[m, i] = C[m, i-1] + C[m - p_i, i-1] + C[m - 2p_i, i-1] + \cdots$$

Use p_i zero times Use p_i once Use p_i twice

- Order of computation: in increasing order of i, then in increasing order of m.

Counting problem: results

- Computation for $p_1 = 1, p_2 = 4, p_3 = 9$ and $m \leq 10$.

m	0	1	2	3	4	5	6	7	8	9	10
$C[m,0]$	1	0	0	0	0	0	0	0	0	0	0
$C[m,1]$	1	1	1	1	1	1	1	1	1	1	1
$C[m,2]$	1	1	1	1	2	2	2	2	3	3	3
$C[m,3]$	1	1	1	1	2	2	2	2	3	4	4