Binomial Coefficient: A Dynamic Programming Approach

Sumanth

June 18, 2025

Abstract

The Binomial Coefficient, denoted $C(n,k)$, represents the number of ways to choose k items from a set of n items without regard to order. This document explores a dynamic programming solution to compute $C(n,k)$, addressing its combinatorial significance, recurrence relation, algorithm, complexity analysis, practical applications, and limitations. Additional insights into space optimization and handling large values are included for a comprehensive understanding.

1 Introduction

The Binomial Coefficient $C(n,k)$, also written as $\binom{n}{k}$, is a fundamental concept in combinatorics, representing the number of ways to select k elements from a set of n elements without considering the order of selection. It appears in various fields, including probability, statistics, and algorithm design, and is a building block of Pascal's Triangle.

A naive recursive approach to compute $C(n,k)$ suffers from redundant calculations, leading to exponential time complexity. **Dynamic programming** offers an efficient solution by storing intermediate results, leveraging the problem's overlapping subproblems and optimal substructure.

2 Problem Formulation

Given two non-negative integers n and k (where $k \leq n$), compute $C(n,k)$, the number of ways to choose k items from n items.

Mathematically:

$$C(n,k) = \binom{n}{k} = \frac{n!}{k!(n-k)!}$$

However, direct factorial computation is inefficient for large n. Instead, we use a recursive relation suitable for dynamic programming.

2.1 Recurrence Relation

The Binomial Coefficient satisfies the following recurrence:

$$C(n,k) = C(n-1,k-1) + C(n-1,k)$$

With base cases:

$$C(n,0) = C(n,n) = 1$$
$$C(n,k) = 0 \quad \text{if } k > n$$

This recurrence reflects the combinatorial interpretation: to choose k items from n, either include the n-th item (requiring $k-1$ more items from $n-1$) or exclude it (requiring k items from $n-1$).

3 Dynamic Programming Algorithm

The bottom-up dynamic programming approach constructs a 2D table dp of size $(n+1)\times(k+1)$, where $dp[i][j]$ stores $C(i,j)$. Pseudocode is as follows:

```
Input: n, k
Output: C(n, k)

Initialize dp[n+1][k+1]
For i from 0 to n:
    dp[i][0] = 1  // Base case: C(i, 0) = 1
    If i <= k:
        dp[i][i] = 1  // Base case: C(i, i) = 1
For i from 1 to n:
    For j from 1 to min(i, k):
        dp[i][j] = dp[i-1][j-1] + dp[i-1][j]
Return dp[n][k]
```

4 Complexity Analysis

- **Time Complexity**:
  - Naive recursion: $O(2^n)$, due to overlapping subproblems.
  - Dynamic programming: $O(n\cdot k)$, as the algorithm fills a table of size $(n+1)\times(k+1)$.
- **Space Complexity**:
  - $O(n\cdot k)$ for the 2D dp table.
  - Can be optimized to $O(k)$ using a 1D rolling array (see Section 5).

5 Space Optimization

Since $dp[i][j]$ depends only on $dp[i-1][j-1]$ and $dp[i-1][j]$, we can use a 1D array $dp[k+1]$ to reduce space complexity to $O(k)$. The array is updated iteratively, computing each row from right to left to avoid overwriting values:

```
Input: n, k
Output: C(n, k)
```

```
Initialize dp[k+1] with dp[0] = 1
For i from 1 to n:
    For j from min(i, k) down to 1:
        dp[j] = dp[j-1] + dp[j]
Return dp[k]
```

This optimization is particularly useful for large n and moderate k.

6 Handling Large Values

For large n, $C(n,k)$ can exceed the range of standard integer types. To prevent overflow:

- Use `long long` or arbitrary-precision arithmetic libraries.
- In competitive programming, apply modular arithmetic (e.g., modulo 10^9+7) to keep intermediate results manageable:

$$C(n,k) \mod m = (C(n-1,k-1) \mod m + C(n-1,k) \mod m) \mod m$$

7 Practical Applications

The Binomial Coefficient is widely used in:

- **Probability and Statistics**: Computing probabilities in binomial distributions.
- **Pascal's Triangle**: Each entry is a binomial coefficient.
- **Combinatorial Algorithms**: Solving problems involving subsets or permutations.
- **Coding Contests**: Common in dynamic programming and math-based challenges.

8 Example

Compute $C(5,2)$.

Using the DP table:

$i \setminus j$	0	1	2
0	1	0	0
1	1	1	0
2	1	2	1
3	1	3	3
4	1	4	6
5	1	5	10

Result: $C(5,2) = 10$, meaning there are 10 ways to choose 2 items from 5.

9 Limitations and Extensions

- **Limitations**:
  - Large n and k may cause overflow without modular arithmetic.
  - Space complexity can be significant for large k.
- **Extensions**:
  - **Lucas' Theorem**: Computes $C(n,k) \mod p$ for prime p.
  - **Multi-choose**: Computes combinations with repetition.
  - **Fast Fourier Transform**: Used for convolution-based binomial calculations in advanced applications.

10 References

- GeeksForGeeks: Binomial Coefficient
- Wikipedia: Binomial Coefficient
- Cormen, T. H., et al., *Introduction to Algorithms*, Chapter on Dynamic Programming.