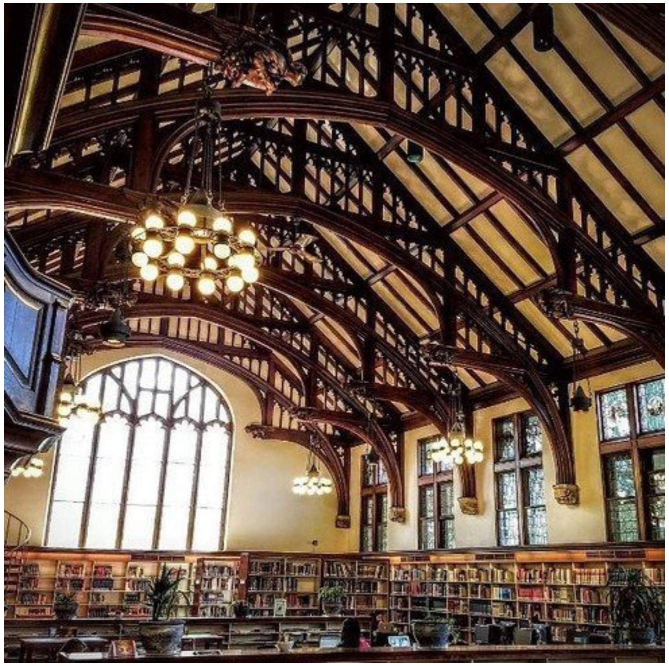
Dynamic programming

Reading: Kleinberg & Tardos
Ch. 6
CLRS Ch. 15

Books for thought

Suppose you found yourself with time to read books…



Given n books in the library,

how many ways are there to choose k to read?

Binomial coefficients

- # ways to choose k elements from a set of size n
- Pick one element *x*
- Count sets that have *x* and sets that don't
  - # sets that have *x*: # ways to choose k-1 elements from set of size n-1
  - # sets that don't: # ways to choose k elements from set of size n-1
- Feels recursive…

Binomial coefficients

- # ways to choose k elements from a set of size n
- Pick one element *x*
- Count sets that have *x* and sets that don't
  - # sets that have *x*: # ways to choose k-1 elements from set of size n-1
  - # sets that don't: # ways to choose k elements from set of size n-1
- Feels recursive…

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

Binomial coefficients

- # ways to choose k elements from a set of size n
- Pick one element *x*
- Count sets that have *x* and sets that don't
  - # sets that have *x*: # ways to choose k-1 elements from set of size n-1
  - # sets that don't: # ways to choose k elements from set of size n-1
- Feels recursive…

$$\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$$

- Base cases: k = 0 or n = k
  - Exactly one way to choose k elements from a set of size k

Binomial coefficients

```
binomialCoefficient( n, k ):
    if ( k == 0 || k == n ):
        return 1
    else:
        return binomialCoefficient( n - 1, k - 1 ) +
               binomialCoefficient( n-1, k )
```

Binomial coefficients

```
binomialCoefficient( n, k ):
    if ( k == 0 || k == n ):
        return 1
    else:
        return binomialCoefficient( n - 1, k - 1 ) +
               binomialCoefficient( n-1, k )
```

Exactly one way to choose nothing OR everything

Binomial coefficients

```
binomialCoefficient( n, k ):
    if ( k == 0 || k == n ):
        return 1
    else:
        return binomialCoefficient( n - 1, k - 1 ) +
                binomialCoefficient( n-1, k )
```

Exactly one way to choose nothing OR everything

Fix a single book -- you can read it

Binomial coefficients

```
binomialCoefficient( n, k ):
    if ( k == 0 || k == n ):
        return 1
    else:
        return binomialCoefficient( n - 1, k - 1 ) +
               binomialCoefficient( n-1, k )
```

Exactly one way to choose nothing OR everything

Fix a single book -- you can read it

Or not

Binomial coefficients

```
binomialCoefficient( n, k ):
    if ( k == 0 || k == n ):
        return 1
    else:
        return binomialCoefficient( n - 1, k - 1 ) +
               binomialCoefficient( n-1, k )
```

Exactly one way to choose nothing OR everything

read it: you have one less book to choose from and one less book slot to fill

don't read it: you have one less book to choose from and still need to fill all slots

Binomial coefficients

```
binomialCoefficient( n, k ):
    if ( k == 0 || k == n ):
        return 1
    else:
        return binomialCoefficient( n - 1, k - 1 ) +
               binomialCoefficient( n-1, k )
```

Binomial coefficients

```
binomialCoefficient( n, k ):
    if ( k == 0 || k == n ):
        return 1
    else:
        return binomialCoefficient( n - 1, k - 1 ) +
               binomialCoefficient( n - 1, k )
```

- What does the recursion tree look like?
  - Start with b(5,3)...

Can we improve this?

- Notice **optimal substructure**
  - Answers to subproblems give answers without needing refinement
- Notice **overlapping subproblems**
  - Redoing work is expensive!
- How can we solve this?
  - “memoization”

Adding in memoization

- Recursive approach in python tutor
- Recursive approach infused with memoization
- Bottom up iterative approach for memoization