動的計画法

d-hacks B3 mioto

動的計画法

- Dynamic Programming
- 同じ計算を繰り返し行わない
- 計算結果をメモリに記憶
- メモリの情報を再利用

動的計画法と全探索

Program 11.1: 動的計画法の例

```
solve(i, m)
  if dp[i][m]が計算済み
    return dp[i][m]

  if m == 0
    dp[i][m] = true
  else if i >= n
    dp[i][m] = false
  else if solve(i+1, m)
    dp[i][m] = true
  else if solve(i+1, m - A[i])
    dp[i][m] = true
  else
    dp[i][m] = false

  return dp[i][m]
```

Program 6.4: 整数が作れるかどうかを判定する再帰関数

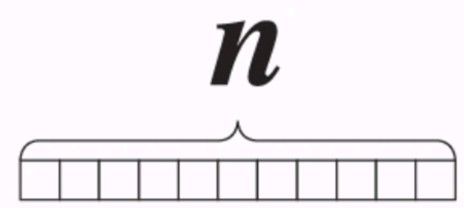
```
solve(i, m)
  if m == 0
    return true
  if i >= n
    return false
  res = solve(i + 1, m) || solve(i + 1, m - A[i])
  return res
```

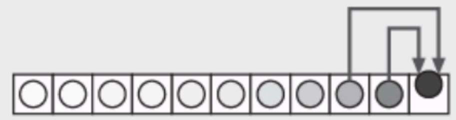
毎回計算

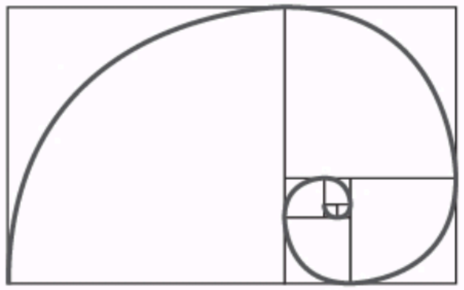
動的計画法と全探索

- Time Complexity
  - 全探索：O(2^n)
  - 動的計画法：O(nm)
- Space Complexity
  - 動的計画法：O(nm)

フィボナッチ数列







思考★　　実装★

フィボナッチ数列の第n項目を出力するプログラムを作成してください。ここではフィボナッチ数列を以下の再帰的な式で定義します：

$$
fib(n) = \begin{cases} 1 & (n = 0) \\ 1 & (n = 1) \\ fib(n-1) + fib(n-2) & \end{cases}
$$

入力　1つの整数nが与えられます。

出力　フィボナッチ数列の第n項目を1行に出力してください。

制約　$0 \le n \le 44$

入力例

```
3
```

出力例

```
3
```

動的計画法を使わない場合

Program 11.2: 再帰によるフィボナッチ数列

```
fibonacci(n)
  if n == 0 || n == 1
    return 1
  return fibonacci(i - 2) + fibonacci(i - 1)
```

無駄

動的計画法

No Loops

Program 11.3: メモ化再帰によるフィボナッチ数列

```
fibonacci(n)
  if n == 0 || n == 1
    return F[n] = 1 // F[n] に 1 をメモしてそれを返す
  if F[n] が計算済み
    return F[n]
  return F[n] = fibonacci(i - 2) + fibonacci(i - 1)
```

Loops

Program 11.4: 動的計画法によるフィボナッチ数列

```
makeFibonacci()
  F[0] = 1
  F[1] = 1
  for i が 2 から n まで
    F[i] = F[i - 2] + F[i - 1]
```

最長共通部分列

- Longest Common Sequence
- 二つの行の間の全ての共通の解

$$X = \{a, b, c, b, d, a, b\} \qquad Y = \{b, d, c, a, b, a\}$$

$$LCS = \{b, c, b, a\}$$

動的計画法

$$
c[i][j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ c[i-1][j-1] + 1 & \text{if } i, j > 0 \text{ and } x_i = y_j \\ \max(c[i][j-1], c[i-1][j]) & \text{if } i, j > 0 \text{ and } x_i \neq y_j \end{cases}
$$

連鎖行列積

- Matrix Chain Multiplication Problem
- n個の行列の連鎖が与えられた時スカラー乗の回数が最小となる行列の組み合わせ
- 全探索ではO(n!)（time complexity）

動的計画法

$m[n+1][n+1]$	$m[i][j]$ を $(M_i M_{i+1} \ldots M_j)$ を計算するための最小の掛け算の回数とする 2 次元配列
$p[n+1]$	M_i が $p[i-1] \times p[i]$ の行列となるような 1 次元配列

これらの変数を用いて、$m[i][j]$ は次の式で得られます。

$$m[i][j] = \begin{cases} 0 & \text{if } i = j \\ \min_{i \leq k < j} \{m[i][k] + m[k+1][j] + p[i-1] \times p[k] \times p[j]\} & \text{if } i < j \end{cases}$$

動的計画法

Program 11.6: 動的計画法による連鎖行列積の最適解

```
matrixChainMultiplication()
  for i = 1 to n
    m[i][i] = 0

  for l = 2 to n
    for i = 1 to n - l + 1
      j = i + l - 1
      m[i][j] = INFTY
      for k = i to j - 1
        m[i][j] = min(m[i][j], m[i][k] + m[k + 1][j] + p[i - 1] * p[k] * p[j])
```

- O(n^3)