アルゴリズムとデータ構造
動的計画法

Photo by https://www.pakutaso.com/

目次

動的計画法とは？

動的計画法（dynamic prigramming,DP）はアルゴリズム設計技法の一つです。

一言で言えば、**「与えられた問題全体を一連の部分問題に上手に分解し、各部分問題に対する解をメモ化しながら、小さな部分問題からより大きな部分問題へと順に解を求めていく手法」**です。

動的計画法は非常に汎用性の強い手法であり、コンピュータサイエンス上の重要な問題から、世の中のさまざまな現場における最適化問題まで、広範囲の問題を解くのに役立ちます！

- ナップサック問題
- スケジューリング問題
- 発電計画問題
- 編集距離を求める問題 (diff コマンドの仕組みです)
- 音声認識パターンマッチング問題
- 文章の分かち書きをする問題
- 隠れマルコフ問題

解決できる問題の幅が広い＝手法を適用するバリエーションが多彩で習得が難しい

ただ、設計パターン自体はそれほど多くないので、「習うより慣れろ！」精神で経験すれば習得できる。

・・・らしいです。

ABC261_D_Flipping and Bonus

高橋君が N 回コイントスを行います。 また、高橋君はカウンタを持っており、最初カウンタの数値は 0 です。 i 回目のコイントスで表裏のどちらが出たかによって、次のことが起こります。

- 表が出たとき：高橋君はカウンタの数値を 1 増やし、X_i円もらう。
- 裏が出たとき：高橋君はカウンタの数値を 0 に戻す。お金をもらうことは出来ない。

また、M 種類の連続ボーナスがあり、i 種類目の連続ボーナスではカウンタの数値がCiになるたびにYi円もらうことができます。
高橋君は最大で何円もらうことができるかを求めてください。

参考：iwasa提出コード

貰えるお金は試行回数とカウンタの数値で決まる。

「試行回数 i 回」*「カウンタ j」ような条件の下で 1 回目から i 回目までで得られる金額の最大値を更新していく。

メモ(2次元配列dp)を作成。
0回目、カウント0からスタートするため、[0][0]のみ初期値を入れ、ほかは外れ値を入れておく。

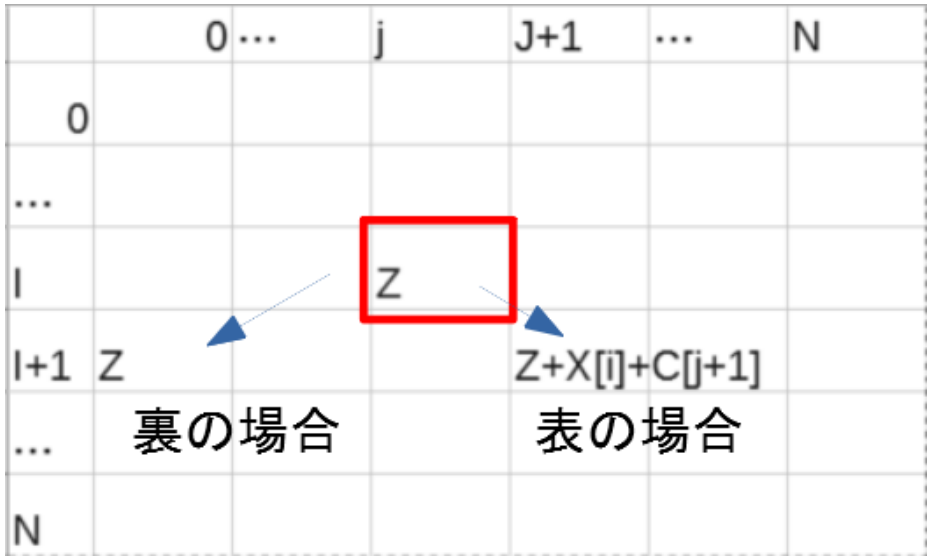
```
// 初期化
vector<vector<ll>> dp(N+1,vector<ll>(N+1,-1));
dp[0][0] = 0;
```

カウンタの値 j

試行回数 i	0	1	2	3	4	5	6
0	0	-1	-1	-1	-1	-1	-1
1	-1	-1	-1	-1	-1	-1	-1
2	-1	-1	-1	-1	-1	-1	-1
3	-1	-1	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1

あるi,jに着目すると、裏になるパターンと表になるパターンの2通りの動作があることが分かる。

```
// 配るDP
rep(i,N){
    rep(j,N){
        if(dp[i][j]==-1) continue;
        // 表の場合
        dp[i+1][j+1] = dp[i][j]+X[i]+CY[j+1];
        // 裏の場合
        dp[i+1][0] = max(dp[i+1][0],dp[i][j]);
    }
}
```



入力例1

```
6 3
2 7 1 8 2 8
2 10
3 1
5 5
```

出力例1

```
48
```

カウンタの値 j

試行回数 i	0	1	2	3	4	5	6
0	0	-1	-1	-1	-1	-1	-1
1	0	2	-1	-1	-1	-1	-1
2	2	7	19	-1	-1	-1	-1
3	19	3	18	21	-1	-1	-1
4	21	27	21	27	29	-1	-1
5	29	23	39	24	29	36	-1
6	39	37	41	48	32	42	44

6回終わった後の最大値

Flog問題

by iwasa.kenta

配るDPと貰うDP

by iwasa.kenta

ナップサック問題

by iwasa.kenta

編集距離を求める問題

by iwasa.kenta

アルゴリズムとデータ構造輪講 2022 技術チャレンジ部

by iwasa.kenta

まとめ

複雑な問題をシンプルな部分問題に上手に分解することがポイントです。
設計手法に慣れましょう。

- アルゴ式：この本の著者の管理する学習コンテンツ
- EPDC：解き進めましょう