レギオ講習会

目標:

1. 計算量の感覚を身につける
2. 便利なデータ構造・アルゴリズムを利用してみる
3. かしこいアルゴリズムのアイデアを勉強してみる

進め方

- 1. と 2. は、全員に講義します
- 各種アルゴリズムについては、余裕のある人向けに講義します

計算量のイメージと賢い考え方１（動的計画法）

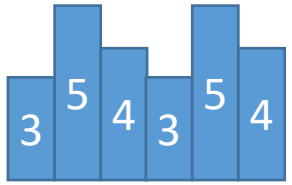
問題

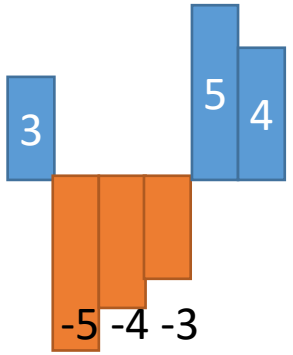
問題AA@レギオ神戸問題セット

問題文＆解答例@github

A_1 .. A_nの部分列の和について、その最大値を答えよ。

- 要素は 10万個ぐらいあることも！

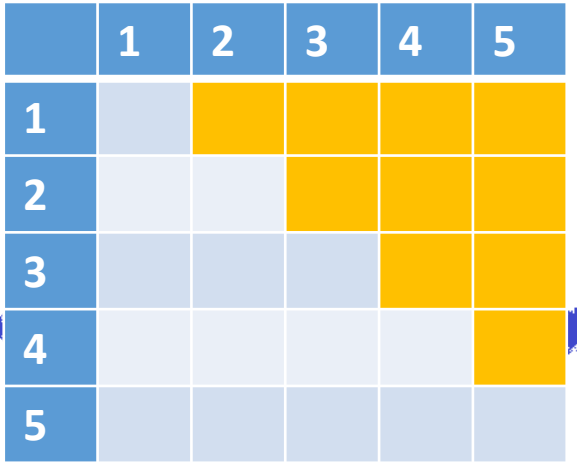




問題

問題AA@レギオ神戸問題セット

問題文＆解答例@github



A_1 .. A_nの部分列の和について、その最大値を答えよ。

- 解1：全部の部分列 A_{from} .. A_{to} について、実際に和を求める！
  - from x to の組み合わせ: $n^2/2$
  - from..to の足し算: n に比例したコスト
  - 併せると n^3に比例したコスト
- どれぐらい厳しいの？？
  - 今回、n は 100000 = 10^5
  - n^3 は 100000 x 100000 x 100000 = 10^{15} = 10^3 兆ってどれぐらい？

```python
for i in range(n):
    for j in range(i+1, n+1):
        #s = 0
        #for k in range(i, j):
        #    s += list[k]
        s = sum(list[i:j])
        result = max(result, s)
```

Python

現実計算機のスペック

- CPU のクロック：数GHz
  - 1秒間に数G回の命令をこなす（1命令1nano sec以下）
  - 1000命令かかる処理でも、数M回こなす
  - つまり「1秒」で解くとは、数G回以内の命令で解くということ
- メモリ容量
  - 主メモリ数GBは当たり前
    - ➢でも、OS の領域も残してね
  - ハードディスクはTBオーダー
    - ➢でも、disk は遅いけどね

但し、主メモリランダムアクセスは、CPUより数十倍遅い

1K = 1000 = 10^3
1M =1000,000 = 10^6
1G = 1000,000,000 = 10^9
1T = 10^{12}

でも、キャッシュは数MB程度以下

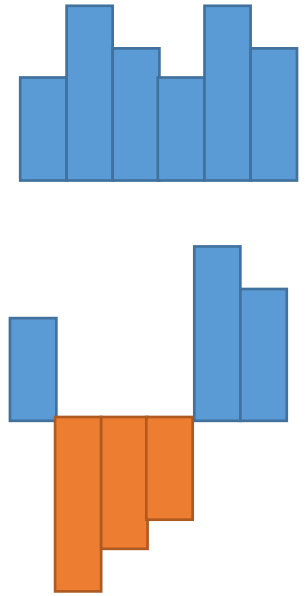
量感覚イメージ

- 累乗: 2^{10}=1024≒K, 2^{20} ≒ M, 2^{30} ≒ G
- 階乗:
  - 1x2x3x4x5=120, 1x2x...x10=3.6Mぐらい
  - ..x13 で4G越える
- 32bit 符号付整数で表現できる値:　-2G ～ +2G
- 時間
  - 1秒に数G回の演算が可能
  - 1分:60秒、1時間:3.6K秒、1日:86.4K秒、
  - 1M秒で11日半ぐらい

もっと賢い方法はないのかね？（考え中）

A_1 .. A_nの部分列の和について、その最大値を答えよ。

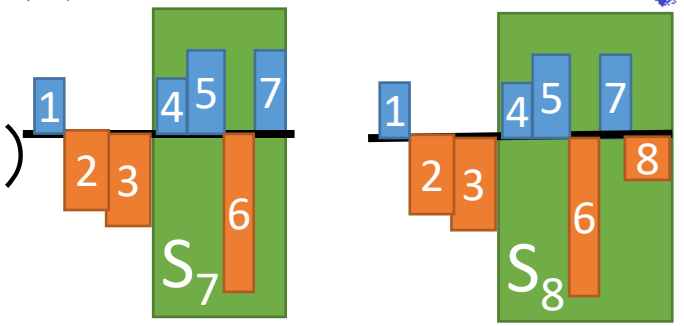
- 観察
  - 全部「正」の値なら、全部足したのが解
  - 途中「負」の領域がある
    - 小さい ⇨ 前後つなげたほうがいい
    - でかい ⇨ 前後個別のほうがいい

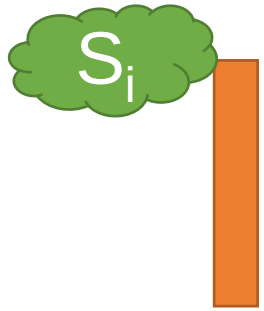


問題文＆解答例@github

もっと賢い方法はないのかね？（解決編：動的計画法）

- 解2：「最後が Ai である部分列の最大和」S_i が分かっていたとする（要素なしの部分列も考慮）
  - S_1 は？　$A_1 > 0$ なら $S_1 = A_1$, $A_1 <= 0$なら $S_1 = 0$
  - S_i がわかっているとして、S_{i+1} は？
    - $S_i + A_{i+1} > 0$ なら　$S_{i+1} = S_i + A_{i+1}$
    - $S_i + A_{i+1} <= 0$ なら　$S_{i+1} = 0$
  - そんな S_i (i = 1 .. n) たちの最大値を求めればOK





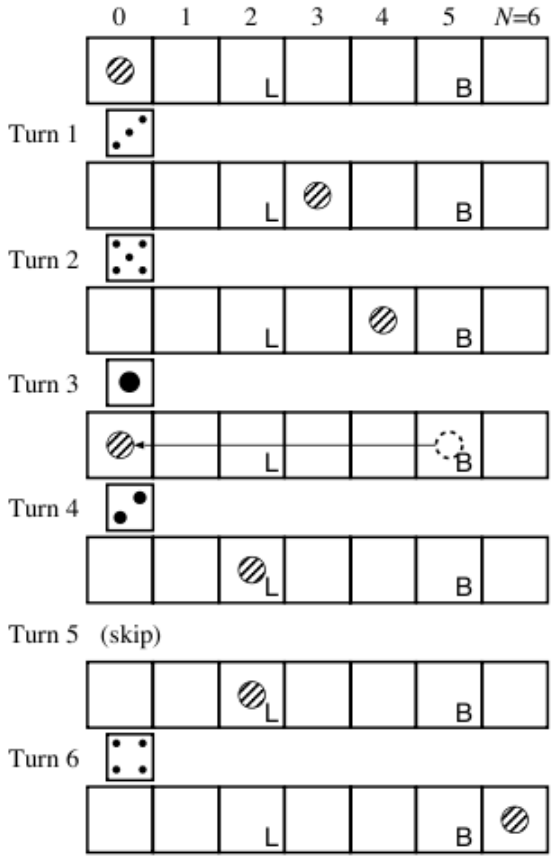
動的計画法 (Dynamic Programming: DP)
部分問題から順に解き、解き終わった部分問題の結果を用いて、より大規模な問題を解く方法

類題

ACM ICPC 2007 アジア予選 東京大会

https://judge.u-aizu.ac.jp/onlinejudge/description.jsp?id=1277

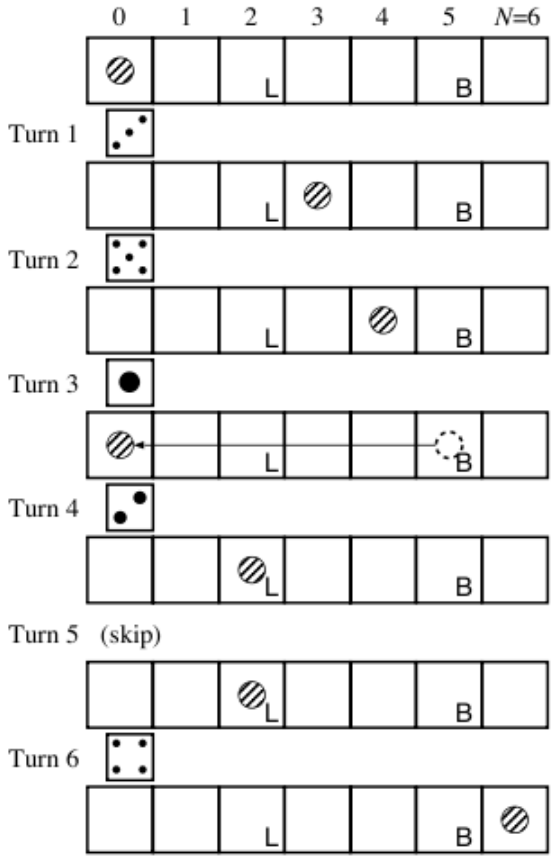
- ある種の双六ゲーム
  - 左端からスタート、右端がゴール
  - サイコロを振って、目の数だけ進む
    - ゴールを超えたら、その分戻る
  - L: 1回休み
  - B: スタートに戻る
- 質問： T 回以内にゴールにたどり着く確率は？(1≦T≦100)



どうやったら解けそう？

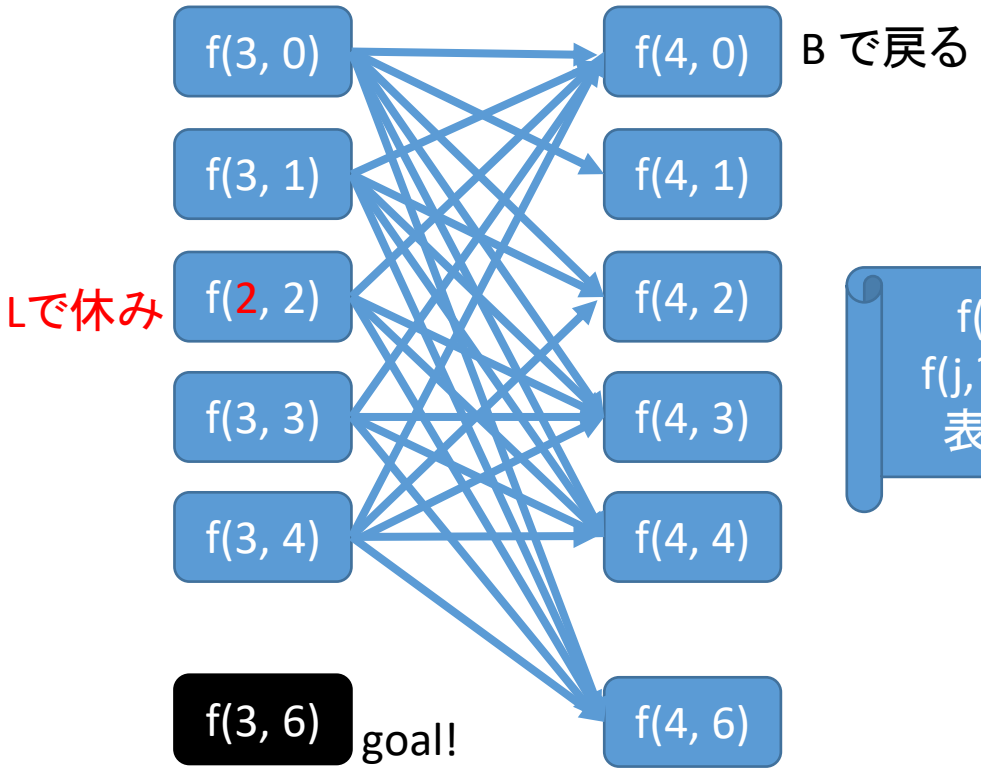
試行錯誤

- 最初は必ずスタート
- 1/6 の確率で 1 から 6 に移動
  - 但し、L についたら1回休み
  - B についたら、スタートに戻る
- サイコロのパターンを全部試す？
  - 10回振るパターン: 6^{10} ～60M
  - 20回だと、6^{20} ～ 3600ペタ？？
  - 100回なんて、、、、

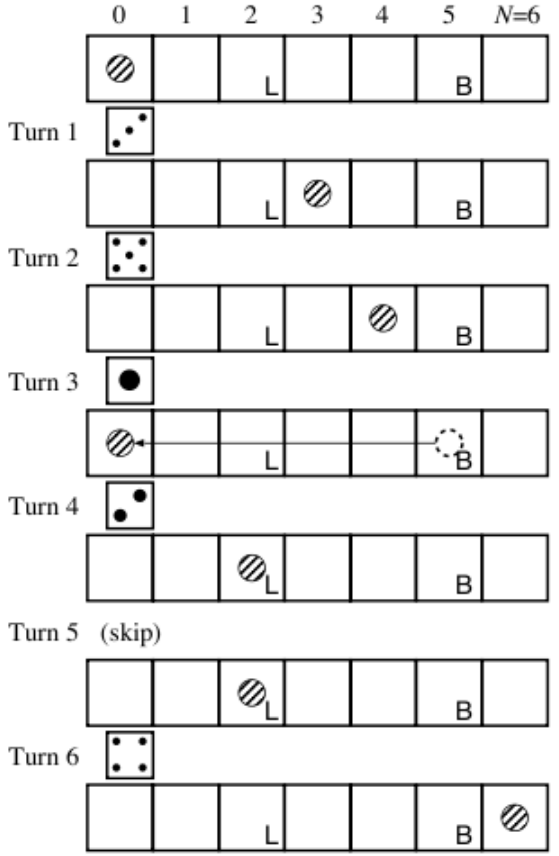


解法

- f(k, p): k回目に位置 p に到着する確率とする



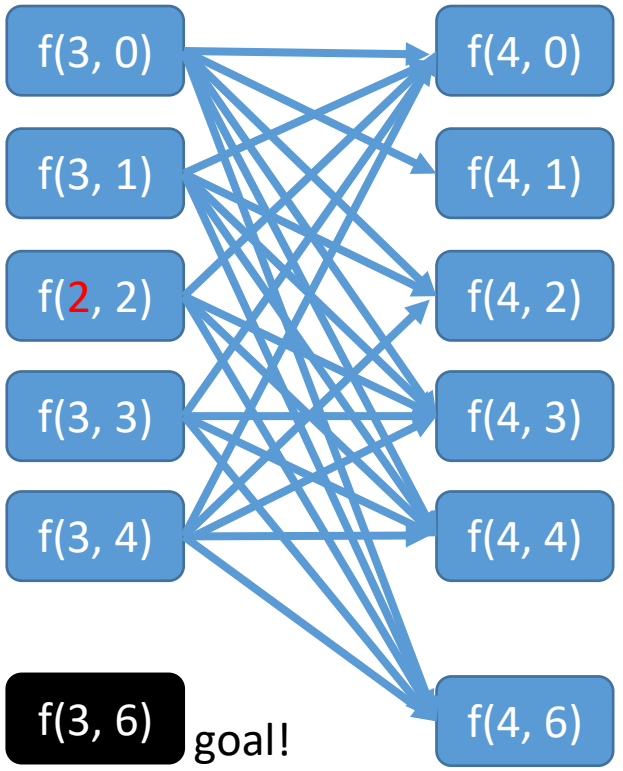
f(k, p)は
f(j,?) (j<k)で
表現可能



動的計画法 (Dynamic Programming, DP)

- 部分問題から順に解き、
- 部分問題の結果を用いて、
  より大規模な問題を解く

```
def solve(n, t):
    prob = [ [ [ 0 ] * (n+1)] * (t+1) ]
    for now in range(t):
        now 以前の結果を使って prob[now][?] を求める
        もしくは
        prob[now][?] の結果を以後の確率に反映
    print(str(prob))
```



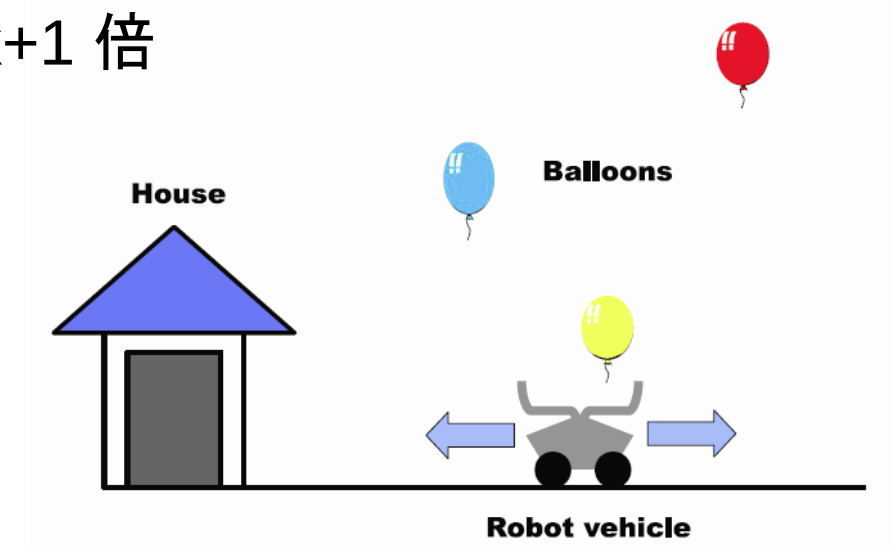
類題

ACM ICPC 2010 アジア予選 東京大会

https://judge.u-aizu.ac.jp/onlinejudge/description.jsp?id=1306

Balloon Collecting

- 上から落ちてくる風船をロボットで集める
  - 与えられるもの：各風船が落ちる場所と時刻
  - ロボットには風船を3個まで載せられる
  - 風船は左端の家で回収
  - 風船を k 個載せると、移動時間は k+1 倍
- 質問
  - 風船をすべて回収できる場合は、最小移動距離
  - 回収できない場合は、何個目で回収不能か？



考え方

- 素直に動的計画法
  - f(k, b): k 番目の風船をキャッチして風船 b 個になるまでの最小移動距離（or 到達不能か）