Monge DP

ヘクト

平成 27 年 11 月 24 日

1 はじめに

Monge DP に必要な事項を忘れないようにするためのメモである.

2 Monge 性

二変数の関数 $f(i,j)$ が「$Monge$」であるとは，任意の $i \leq j$ と $k \leq l$ について

$$f(i,l) + f(j,k) \geq f(i,k) + f(j,l) \tag{1}$$

が成立することをいいます．Monge 性は min-max 束の劣モジュラ不等式だとも言える.

$$f(i,l) + f(j,k) \geq f(\min(i,j), \min(k,l)) + f(\max(i,j), \max(k,l)) = f(i,k) + f(j,l) \tag{2}$$

DP のコスト関数が Monge 性を満たす時には,DP テーブルも Monge 性を満たすことが知られている．この Monge 性を利用することにより，DP を高速化できることが知られている.

3 Divide and Conquer Optimization

この問題は, 式 (3) の形の漸化式で表されて,dp[n][k] を求めたい.

$$\mathrm{dp}[i][j] = \min_{0 \leq k < j} \mathrm{dp}[i-1][k] + \mathrm{cost}[k][j] \tag{3}$$

この式は普通の DP であるため, 計算量 $O(kn^2)$ で計算できる. ここで Monge 性を利用する. 式 (4) を満たす C[i][j] を dp[i][j] が最小値を取るインデックスと定義する.

$$\mathrm{C}[i][j] = \min_{0 \leq k < j} \mathrm{dp}[i-1][k] + \mathrm{cost}[k][j] \tag{4}$$

Monge 性より,C[i][j] は式 (5) の関係を満たす.

$$\mathrm{C}[i][j] \leq \mathrm{C}[i][j+1] \tag{5}$$

式 (5) の関係から,C[i][j] を分割統治法を用いることで計算量 $O(kn \log n)$ で計算できる．また, 行最大値発見問題のアルゴリズムを用いることで計算量 $O(kn)$ で計算できる.

ソースコード 1: Divide and Conquer

```
int dp[kmax][nmax];
void solve(int i,int L,int R, int optL,int optR){
      if(L>R) return;
      int M=(L+R)/2;
      int optM=-1;
      for(int k=optL;k<=optR;++k)
            if(dp[i+1][M]>dp[i][k]+cost[k][M])
                  dp[i+1][M]=dp[i][k]+cost[k][M] ,optM=k;
      solve(i,L,M,optL,optM);
      solve(i,M,R,optM,optR);
      return;
}

int main(void){
      for(int i=0;i<=k;++i)
            for(int j=0;j<=n;++j)
                  dp[i][j]=inf;
      dp[0][0]=0;
      for(int i=0;i<k;++i) solve(i,0,n,0,n);
      cout << dp[k][n] << endl;
      return 0;
}
```

4 Knuth Optimization

最適二分探索木問題を解くために,Knuth が開発した手法らしい．この問題は, 式 (6) の形の漸化式で表される．

$$\mathrm{dp}[i][j] = \mathrm{cost}[i][j] + \min_{i<k<j} \mathrm{dp}[i][k] + \mathrm{dp}[k][j] \tag{6}$$

この DP は区間 DP であるため, 計算量 $O(n^3)$ で計算できる．ここで Monge 性を利用する．式 (7) を満たす C[i][j] を dp[i][j] が最小値を取るインデックスと定義する．

$$\mathrm{C}[i][j] = \arg\min_{i<k<j} \mathrm{dp}[i][k] + \mathrm{dp}[k][j] \tag{7}$$

Monge 性より,C[i][j] は式 (8) の関係を満たす．

$$\mathrm{C}[i][j-1] \leq \mathrm{C}[i][j] \leq \mathrm{C}[i+1][j] \tag{8}$$

式 (8) の関係から, 幅が小さい方向から DP テーブルを埋めることにより, 計算量 $O(n^2)$ で計算できる．

ソースコード 2: Divide and Conquer

```
int dp[nmax][nmax];
int C[nmax][nmax];

int main(void){
      for(int i=0;i<=k;++i)
            for(int j=0;j<=n;++j)
                  dp[i][j]=inf;
      for(int i=0;i<n;++i) dp[i][i]=0,C[i][i]=i;

      for(int d=2;d<=n;++d){
            for(int i=0;i+d-1<n;++i){
                  int L=i,R=i+d-1;
                  int idx=C[L][R-1];
                  for(int j=C[L][R-1];j<=C[L+1][R];++j){
                        if(dp[L][R]>dp[L][j]+dp[j+1][R]+cost[L][R])
                              dp[L][R]=dp[L][j]+dp[j+1][R]+cost[L][R],idx=j;
```

```
                    }
                    C[L][R]=idx;
                }
        }
        cout << dp[0][n-1] << endl;
        return 0;
}
```

5 参考問題

KUPC 2012 J Sashimi
JAG Summer 2013 Day2 D
JAG Autumn 2015 K

参考文献

[1] Letterfrequency(English),ALGORITMY.NET
http://en.algoritmy.net/article/40379/Letter-frequency-English

[2] デジタル情報の処理と認識 ('12) 15 章・鈴木一史
http://compsci.world.coocan.jp/OUJ/2012PR/pr_15_a.pdf

[3] Alice's Adventures in Wonderland(1866),Lewis Carroll
https://en.wikisource.org/wiki/Alice%27s_Adventures_in_Wonderland_(1866)
/Chapter_1