競技プログラミングアドベントカレンダー 2017
Day2 : ”花火 ”に対するシンプルな解法

tokoharu

2017.12.02

1 はじめに

この記事では，第 2 回ドワンゴからの挑戦状予選の花火という問題 (以下，花火問題) に対する平易な解法を紹介します．さらにアルゴリズムの根底には最小費用流問題とその双対問題が関わっていることを紹介します．また，花火問題の類似問題はいくつか存在し，これらの問題にもこの方針は適用可能ですので，そのような類題をいくつか紹介します．

2 花火への priority queue 解法

問題概要はリンク先をご覧ください．https://dwango2016-prelims.contest.atcoder.jp/tasks/dwango2016qual_e．コンテストの最終問題ということもあり，難しそうに見えます．

ところが，この問題は次のシンプルなコードで AC を獲得できます．

```
#include <bits/stdc++.h>
using namespace std;

typedef long long LL;
typedef pair<LL,LL> PII;

int main() {
    int N,L;
    cin >> N >> L;

    vector<PII> input;
    LL sum = 0;
    for(int i=0; i<N; i++) {
        LL t,p;
        cin >> t >> p;
        input.push_back(PII(-t,p));
        sum += p;
    }

    sort(input.begin(), input.end());

    priority_queue<LL> qu;
    LL ans = 0;
    for(auto elem: input) {
        qu.push(-elem.second);
        qu.push(-elem.second);
        ans += qu.top();
        qu.pop();
    }
    cout << sum + ans << endl;
    return 0;
}
```

この解法の正当性は何か，という疑問は当然出てきます．これに対する答えは，花火問題を最小費用流の双対問題として捉え，最小費用流の形で解けるようにし，グラフの特殊性を考慮することで平易な解法が現れる，という仕組みになっています．

次の章では，一般に最小費用流の双対問題に対する扱いやすい形を紹介します．

3　背景：最小費用流の双対問題

まず，ここでいう双対問題とは，線形計画問題における双対問題を指します．ここではこの定義は説明しないので，正当性をしっかり確かめたい場合には調べてください．

ここでは最小費用流の代わりに最小費用循環流問題の双対問題を考えます．本来の最小費用流ではなく循環流の方を選ぶ理由は，双対問題が解釈しやすくなるからです．

まずは最小費用循環流問題を線形計画問題に定式化してみます．

$$
\begin{aligned}
(P) \quad & \text{minimize} && \sum_{e\in E} c_e f_e && && (1)\\
& \text{subject to} && f_e \geq l_e && e\in E \quad \text{... 双対変数 } a_e && (2)\\
& && -f_e \geq -u_e && e\in E \quad \text{... 双対変数 } b_e && (3)\\
& && \sum_{e\in\delta_v^-\subset E'} f_e - \sum_{e\in\delta_v^+\subset E'} f_e = 0 && v\in V \quad \text{... 双対変数 } p_v && (4)
\end{aligned}
$$

双対変数を上記のような対応になるように a, b, p とするとき, この問題に対する双対問題は次のようになります. (注: この変数 p は花火問題のパラメータ p とは無関係です)

$$
\begin{aligned}
(D1) \quad & \text{maximize} && \sum_{e\in E} l_e a_e - u_e b_e && && (5)\\
& \text{subject to} && a_e - b_e + p_u - p_v = c_e && e=(v,u)\in E \quad \text{... 主変数 } f && (6)\\
& && a_e \geq 0, b_e \geq 0 && && (7)
\end{aligned}
$$

この双対問題を観察すると, 各頂点に対するポテンシャル変数 p の値を固定してみると他の変数 a, b は一意に定まる性質があることに気づけます．次に目的関数をみると, 総和の形の要素をひとつずつ（つまり各辺 e ごとに）見るとこれは $p_u - p_v$ に関する折れ線関数で $p_u - p_v = c_e$ を境に変化する形です．

このことを図示して確認してみます．まず式変形をしておくと $p_u - p_v = c_e - a_e + b_e$ なので, 基準値 c_e よりポテンシャル差が大きくなれば $-u_e$ ずつ目的関数値が増えます．一方, c_e よりポテンシャル差が小さくなれば l_e ずつ目的関数値が増えます (つまり, 傾き $-l_e$)．これを図 1 に示しています．

この図を見ると傾きが逆になっているのがややこしいため, 目的関数値を正負逆にして最小化問題として考えたほうがわかりやすいです．つまり，次の最適化問題を考えてみます．

$$
\begin{aligned}
(D2) \quad & \text{minimize} && \sum_{e\in E} u_e b_e - l_e a_e && && (8)\\
& \text{subject to} && a_e - b_e + p_u - p_v = c_e && e=(v,u)\in E \quad \text{... 主変数 } f && (9)\\
& && a_e \geq 0, b_e \geq 0 && && (10)
\end{aligned}
$$

このときの目的関数は，図 1 と同様に，各辺ごとに図 2 のようにかけます．

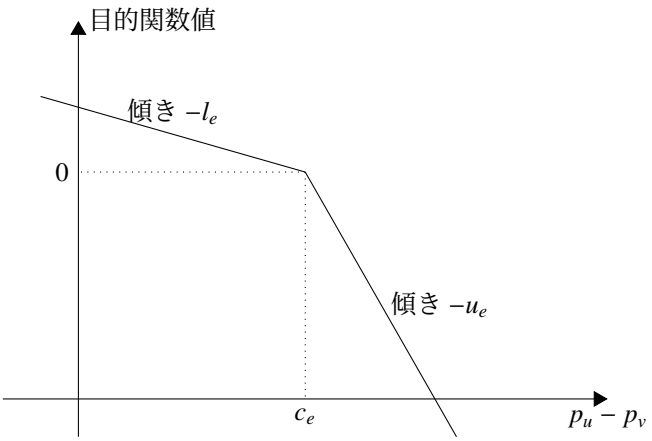

図 1: ある辺 e のポテンシャル差に対する目的関数値 (最大化版)

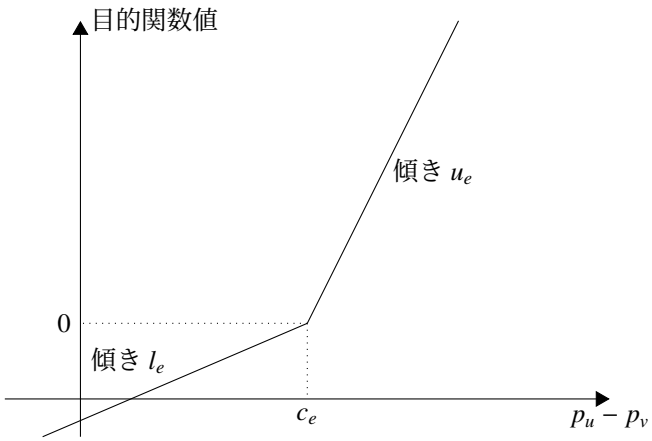

図 2: ある辺 e のポテンシャル差に対する目的関数値 (最小化版)

逆に，目的関数が図 2 のようにポテンシャル差の折れ線関数になるように構成すれば，機械的に (P) の最小費用循環流に変換できます．ただし，この最小化問題の答えは変換した最小費用流問題の解を正負逆にしたものだということに注意が必要です．つまり，[(P) の最適値] = [($D1$) の最適値] = −[($D2$) の最適値] となります．

ここまでで，頂点に定まるポテンシャル差から定まる関数を最小化する問題を導出できました．

4 花火問題を ($D2$) の関数の形に当てはめる

ここで花火問題を ($D2$) の関数の形，つまり図 1,2 の形で表現してみましょう．

花火問題を解くために，時刻 t_i における座標を x_i と定義し，これを調整して最適解を探す問題であるとします．説明のため，t_i はすべて異なるものと仮定します．

このとき，花火問題において最小化すべき関数は $|x_i - p_i|$ の和になります．この p は花火問題の入力の p です．また，他に x が満たすべき条件は $x_i \le x_{i+1}$ と $0 \le x_i \le L$ になります．

まず，目的関数を ($D2$) の関数に直そうと思いますが，ポテンシャルとして定義されている変数が 1 つしかないため，変形できないように見えますが，ポテンシャル値 0 であるような変数 z を新たに定義すれば問題ありません[1]．以上の z を用いれば，次のような図 3 で表される関数 $|x_i - z - p_i|$ で表現できます．したがって，最小流量 −1, 最大流量 1, コスト p_i の辺を z に対応する頂点 から x_i に対応する頂点 へ張ります．

次に，制約についてです．ここでは $0 \le x_1 \le \cdots \le x_N \le L$ を ($D2$) の関数に書きなおしましょう．ここでは $x_i \le x_{i+1}$ の部分について解説します．これは制約ですので，条件を満たさなければコスト INF で条件を満たせばコスト 0 だと考えるとよいです．この結果，図 4 のような関数を定めればよいです．つまり，x_i に対応する頂点から x_{i+1} に対応する頂点へ最低流量-INF, 最大流量 0, コスト 0 の辺を張るとよいです．ただ，これは逆方向に最低流量 0, 最大流量 INF として捉えるほうが扱いやすいので，そうします．

残りの $0, L$ を含む制約については詳細な説明は割愛しますが，結果として構成されたグラフは図 5 のようになります．

[1] このように固定値である変数をおいても問題がない理由は，($D2$) の関数はポテンシャル差のみで定義されているため，自由度が 1 あるからです

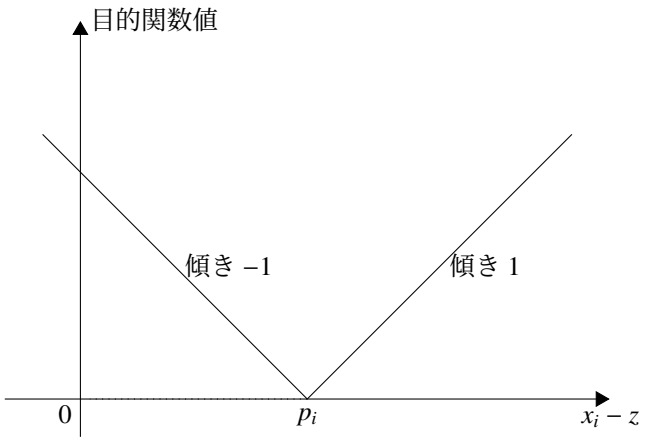

図 3: 花火問題の目的関数部

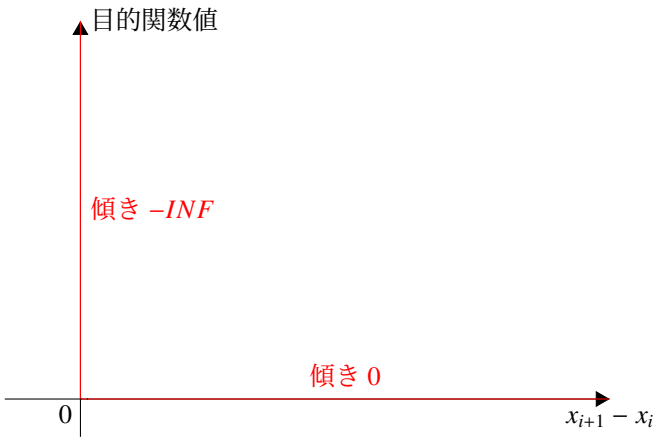

図 4: 花火問題のメイン制約

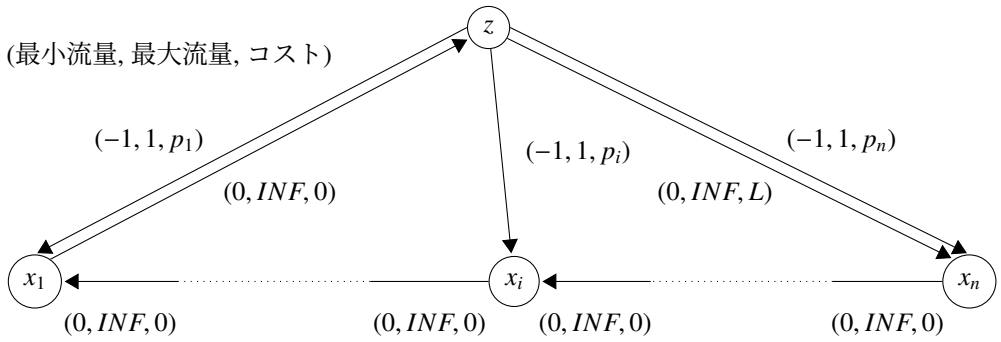

図 5: ($D2$) に当てはめた結果構成されるグラフ

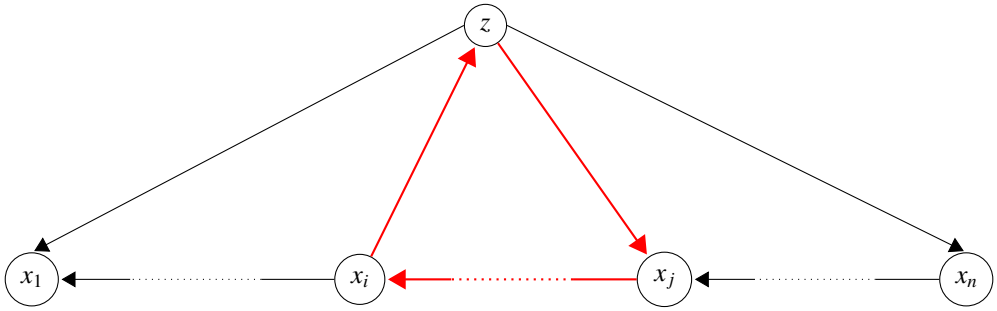

図 6: 循環流の形状

5 高速化

さて，ここからは図 5 の最小費用循環流問題を高速に解く方法を考えます．これ以降，変数に対応する頂点の名前を，単にその変数の名前で呼ぶことにします．例えば以降 x_i は頂点名であるとします．

循環流を考えますので，このフローは閉路の流れの和に分解が可能です．その閉路の形状に着目すると，ここでのフローは必ず頂点 z を通り，下部では x の添字が大きい方から小さい方に流れるような循環流となることがわかります．図 6 の赤線が閉路の例になります．

したがって，次のように添字の大きい順に調べる気持ちのアルゴリズムが構成できます．ただし，x_n, x_1 は特殊辺があるので，適切に扱ってください．

1. x_n から添字の降順で頂点を見ていく．
2. x_i において,$-p_i + p_j < 0$ でかつ流量に余裕がある j $(i < j)$ が存在すれば，そのうち $-p_i + p_j$ が最小となる j を用いて循環流を作る．
3. フェーズ 2 で 1 流れれば 2 つ，フェーズ 2 で流れなければ x_i には 1 つ余裕があるとしておく

ここまでくれば，priority queue によって $O(n \log n)$ 時間のアルゴリズムを構築できることになります．

以上，ここまでがキーアイデアになり，これを用いれば花火の実装もできるでしょう．しかし，これはそのまま冒頭のソースコードに変換できません．これについては，図 5 のグラフを少し変形することで実現しました．これは次のことを考慮すれば到達できます．

- 実は x_n, x_1 にある特殊辺は必要ありません．これは，入力の p が 0 以上 L 以下であることが保証されているため，これらの辺を用いても負閉路を作れないため無駄だからです．
- 負の最小流量を消して最小費用流に変換します．これによって，super source から頂点 z に流量 n の辺が張られ，各頂点 x_i から super sink へ流量 1 の辺が張られ，z から x_i の辺は最小流量 0, 最大流量 2 の辺になり，結局両方向の流れが起きうる状況が解消されます．

変形後のグラフは図 7 になります．ソースコード中の sum は 0 – (負の流量を消した分を調整する量) を表し，ans は 0 – (図 7 での最小費用流のコスト) になります．求めたい量はこれを正負反転したものなので，sum+ans が答えになります．

6 演習問題，類題集

次の 3 題が私の知っている類題になります．他にご存知であればぜひ教えてください．

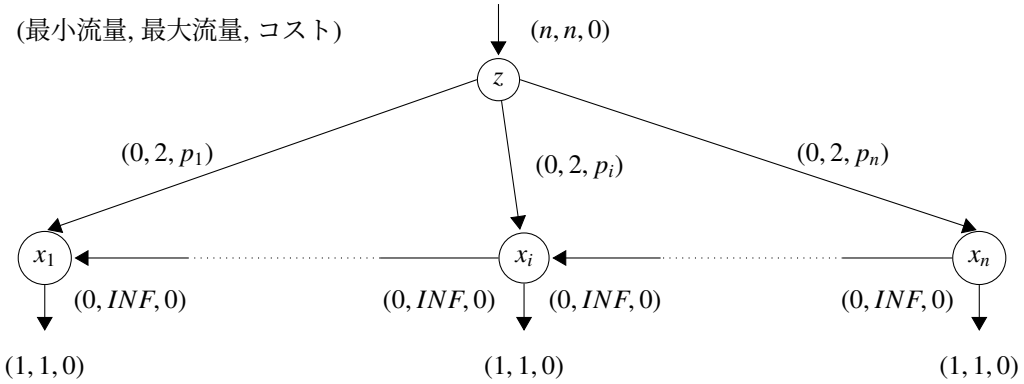

図 7: ソースコードに対応するグラフ

問題	コメント
壁壁壁壁壁壁壁 / WAAAAAAAAAAAAALL	花火問題に帰着するのはわかりづらいかも
じょうしょうツリー	花火問題の自然な拡張．だからシンプルにかける． (Meldable Heap は必要ない!!!)
Codechef Manufacturing Goods	元問題は $N \leq 400$ ですが，$N \leq 10^5$ で解いてください．

7 余談

花火はその解説をみたり，他の人の解法を見る限り，多くの解法が考えられる非常に性質の良い問題です．さらに ”じょうしょうツリー ”の部分問題であることを思えばこのことはより納得できると思います．では，このような様々な解法の方針を制限しつつ，今までこの文書で書いた priority queue 解法を際立たせる問題は作れないか，という疑問が生まれました．JAG Asia 2017 の J 問題はこのモチベーションから生まれました．

8 さいごに

ここまで読んでいただきありがとうございます．
これを読んだ皆さん，ぜひこの技術を使う問題を作ってみてください．