Aliens 優化

Yihda Yol

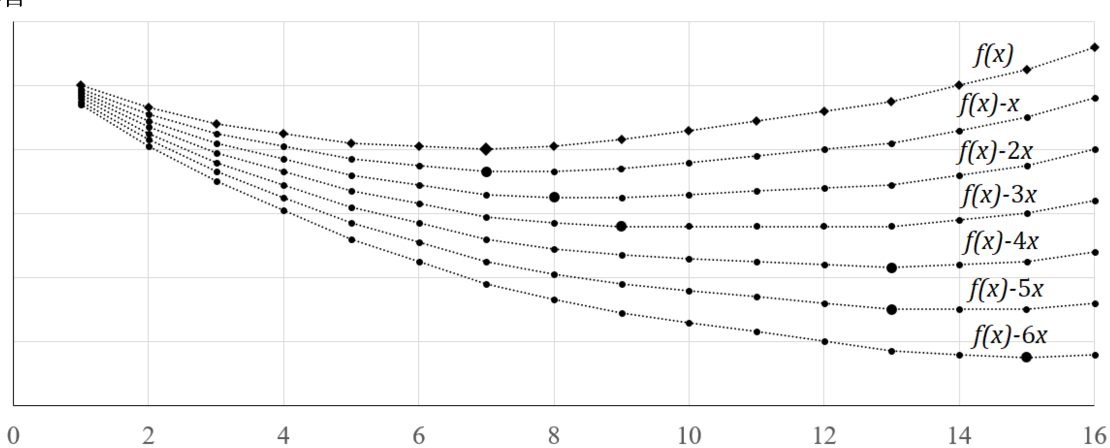
2018 年 3 月 11 日

1 核心概念

Aliens 優化可以歸類為一種 DP 優化。顧名思義，就是 IOI 2016 Aliens 中所用到的技巧。這個技巧近期在臺灣競程界普及了，所以就在這裡介紹一下吧。

首先，我們來考慮一個問題：現在有一個函數 $f(x)$，定義域在 1 到 K 的整數上，已知這個函數是凸的（也就是它的差分遞增）。不幸的是，沒有任何足夠有效率的演算法可以直接計算出 f 的函數值。然而有這麼一個演算法 A，任意給定一個 p，它可以算出最小值 $M(p) = \min_x(f(x) + px)$ 和最小值發生的位置 $V(p) = \min \arg\min_x(f(x) + px)$（如果有很多個最小值，回傳最小的 x）。給定一個 k，要如何求出 $f(k)$ 呢？

首先有個非常顯然的性質：如果戳了一個 p 的值，然後跑完演算法 A 後它告訴我們 $V(p) = k$ 上，那答案就是 $M(p) - pk$ 了。接下來，我們可以試著把 $f(x) + px$ 的圖形畫出來看看：



其中每條線上特別大的點代表最小值的位置。從這個圖可以明顯看出當 p 愈小的時候，最小值會發生在 x 愈大的地方。其實這個性質對於所有的函數 f 都是對的，不必然要是

凸函數。至此答案呼之欲出：只要對 p 二分搜，搜到 $V(p) = k$ 的那點，然後開心的回傳 $M(p) - pk$ 就好了！

然而還有一些問題沒有解決。因為 f 雖然是凸的，但不是嚴格凸的，這代表我們有時候會根本沒辦法找到一個 p 使得 $V(p) = k$。例如上圖中就不可能找到任何一個 p 使得 $V(p) = 11$，因為 $f(9)$ 到 $f(13)$ 是等差的。但是可以注意到，因為 f 是凸的，如果這種情形發生，代表 k 一定位於 f 中一個等差段落的中間。也就是說，如果找不到任何 $V(p) = k$，就找到最小的 p_0 使得 $V(p_0) < k$，以及最大的 p_1 使得 $V(p_1) > k$，算出 $f(V(p_0))$ 和 $f(V(p_1))$，就可以用等差的性質得出 $f(k) = f(V(p_0)) + (f(V(p_1)) - f(V(p_0)))\frac{k-V(p_0)}{V(p_1)-V(p_0)}$。

實作上如果小心，其實可以避免內插的使用。由於演算法 A 在有多個最小值的時候會回傳 x 最小的一個，因此 $f(V(p_0)) + p_0V(p_0) = f(k) + p_0k$（因為我們要求 p_0 最小，因此 $f(x) + p_0x$ 是「平底」的），也就是說 $f(k) = M(p_0) - p_0k$！如果演算法 A 在有多個最小值的時候是回傳 x 最大的，就得改用 p_1 來算。當然，如果是浮點數的話，因為 p_0 和 p_1 只差了一個 eps，所以不管用哪個基本上都會對。

二分搜的部分，易知如果 f 的值域是整數，那麼用整數二分搜就可以了。至於二分搜範圍的決定，易知 $V(f(K-1) - f(K) - 1) = K$、$V(f(1) - f(2)) = 1$，因此理論上左右界分別是 $f(K-1) - f(K) - 1$ 和 $f(1) - f(2)$。當然實際上我們無法算出這兩個值（因為沒辦法算 $f(x)$），因此需要根據題目決定合理的範圍。複雜度則是 $O(\log($ 二分搜值域 $) \times$ 演算法 A 複雜度 $)$，實際寫出來的話大概長這樣：

Algorithm 1: Outline of Aliens Optimization

```
struct Result { int M, V; };
Result getMandV(int p);
int Aliens(int K) {
  int L = 0, R = 1000000; //二分搜左右界
  while (L + 1 < R) {
    int M = (L + R) / 2;
    int p = M - 1;
    Result res = getMandV(p); //回傳 M(p) 與 V(p)
    if (res.V == K) return res.M - p * K;
    if (res.V < K) R = M;
    else L = M;
  }
  Result res = getMandV(L); //p0 = L, p1 = L - 1
  return res.M - L * K;
}
```

要特別注意一件事情：在上述的推導當中，只有最後一步（內插）用到了 f 的凸性。所以如果硬把這個演算法套在不是凸的 f 上，二分搜的部分仍然會照常運作，只是最後內插出來的答案會是錯的而已。

1.1 題外話：二分搜小技巧

如果 f 的值域是浮點數的話，就會要對浮點數二分搜。但是如果用一般的二分搜方法去二分搜 $[1, 10^{100}]$ 之類的範圍就會把常數噴得很大，而且精度也很難處理。解決的方法是先搜指數再搜尾數，這樣可以完美的搜到所需的精度，也不必擔心太多的浮點數誤差。實作方法當然可以寫兩次二分搜；但是有一件邪惡的事情是 IEEE 754 的浮點數恰好指數在高位、尾數在低位，所以：

Algorithm 2: Floating Point Binary Search

```
double BinarySearch(double L, double R, int eps) {
//二分搜範圍 [L,R) (L,R>0)，相對誤差 2^{-eps}
  while (R - L > (1ll << (52 - eps))) {
    unsigned long long l, r;
    double M;
    memcpy(&l, &L, 8); memcpy(&r, &R, 8);
    l = (l + r) / 2;
    memcpy(&M, &l, 8); //Magic!!
    if (check(M)) R = M;
    else L = M;
  }
  return L;
}
```

2 應用

看到這裡你可能會覺得奇怪，一開始不是說這是一種 DP 優化，為甚麼看起來只是在解說一個凸函數求值問題呢？既然是講解 Aliens 優化的講義，當然要拿 Aliens 出來說明囉！

Aliens 的題目大意如下：有個 $M \times M$ 的網格，其中有 N 個重要格子，現在要用 K 個正方形把所有的重要格子，每個正方形都要覆蓋完整的格子，且對角線需與網格的主對角線重合。求這 K 個正方形的總覆蓋面積最小值。

如果對 DP 夠熟的話，可以一眼看出這是個經典的 DP 問題。把一些不必要的重要格子移除、算出 l, r 並排序後，以 $dp[i][j]$ 代表用 i 個正方形蓋住前 j 個重要格子的最小覆蓋面積，可以列出 DP 式：

$$dp[i][j] = \min_{k<j} \left(dp[i-1][k] + (r_{j-1} - l_k)^2 - \max(0, r_{k-1} - l_k)^2 \right)$$

複雜度是個很明顯的 $O(N^2K)$。但是多看了它一眼，就發現可以斜率優化：

$$dp[i][j] = r_{j-1}^2 + \min_{k<j}\left(-2l_k r_{j-1} + dp[i-1][k] + l_k^2 - \max(0, r_{k-1} - l_k)^2\right)$$

斜率有單調性，可以單調隊列，$O(NK)$，應該可以了吧！看了一下測資範圍，結果發現 $N, K \leq 10^5$。

這時候我們試著改寫一下：令 $f(x) = dp[x][N]$。給定 p，我們有沒有辦法求出 $f(x) + px$ 的最小值呢？答案是可以的！注意到這相當於不限制正方形的數量，但是給每個正方形都加上 p 的 penalty，求總分的最小值。這個 DP 和原本的 DP 幾乎一模一樣，只是多加了個常數並少了一個維度而已：

$$dp'[j] = p + r_{j-1}^2 + \min_{k<j}\left(-2l_k r_{j-1} + dp'[k] + l_k^2 - \max(0, r_{k-1} - l_k)^2\right)$$

這個 DP 可以在 $O(N)$ 求出，而 $dp'[N]$ 就是 $f(x) + px$ 的最小值，也就是 $M(p)$；在 DP 的同時記錄轉移次數，就可以得到 $V(p)$。最後可以證明 f 的確是凸的，因此套用前一節提到的演算法，就可以算出 $f(K)$，也就是 $dp[K][N]$ 了！

注意把 DP 看成「對每個正方形加 p 的 penalty」，可以明顯地看出 $V(0) = N$（沒有 penalty 就對每個重要格子蓋一個最小蓋得到它的正方形就好了）、$V(M^2) = 1$（直接用一個超大正方形把全部蓋住），於是就有一個很棒的二分搜左右界。因為值域是整數，所以直接用整數二分搜，總複雜度是 $O(N \log M)$。

可以發現 Aliens 優化是把 DP 的一個維度用 penalty 加二分搜取代，所以它會有一個內部的 DP，而這個內部 DP 也很有可能會用到其他的 DP 優化。而二分搜的左右界則通常要從原本 DP 的性質去推斷。另外，Aliens 優化是非構造性的：如果答案是內插出來的話，就沒有 DP 的過程可以回溯，也就沒辦法輸出可行解。

通常 Aliens 優化的凸性都很難證（Aliens 的官方解也沒有給證明），但是可以用直覺去感受 (?)，或乾脆實作看看。但是實作如果 WA 的話可能就要好好判斷一下到底是自己寫爛還是根本沒有凸性。

2.1 習題

以下幾題除了用 Aliens 優化解題以外，也請試著推導出合理的二分搜範圍。

1. (TIOJ 1986) 數線上有 N 個點 x_i，現在要選定 K 個特殊點，問每個點到距離它最近的特殊點的距離總和最小可以到多少。$N \leq 3 \times 10^5, 0 \leq x_i \leq 10^9$。

2. (TIOJ 2001) 題目有點複雜，請自行閱讀。

3. (TIOJ 2039)(2017 全國賽) 給定一個商品的價格序列 P_i，長度為 N。你可以進行最多 K 次買賣（只能先買後賣，且在買入到賣出之間不能再買入），如果買入的時候價格是 P_i、賣出的時候價格是 P_j，那你會賺到 $P_j - P_i$ 元。求你最多可以賺到多少錢。$N \le 10^6, P_i \le 10^7$。
（註：這題有 $O(N)$ 或 $O(N \log N)$ 的 greedy 解，但可以嘗試寫 $O(N \log C)$ 的 Aliens 優化。）

4. (https://goo.gl/1pQb7D，CF 要加入 tw-icpc-blog 才能看) 類似上一題，但是一次買賣賺到的錢變成 $(P_j - P_i)^2$。
（註：這題雖然長得很像上一題，但是它沒有凸性，所以不能 Aliens 優化。請嘗試找出反例。）