**斯坦納樹**

斯坦納樹屬於NP-Hard問題

定義：

在平面上求一點，使得此點與平面上原有點之間的距離相加為最小。[1]

假設是NP-C問題，如何解釋？

[2]如何計算斯坦納樹是一個NP問題，即你無法在多項式時間內求解。但是如果已知K非常小，我們可以在關連於3k的一個時間複雜度內利用動態規劃解決這個問題。

首先，我們需要意識到斯坦納樹，也是一棵樹，樹必然有根，我們可以用枚舉根來進行計算。並且由於我們僅關心關鍵點是否包含在樹中，因此我們對關鍵點進行狀態壓縮，共2k種狀態，我們可以利用二進位來表示。

我們記dp(i,s)表示以i為根，i可以是普通點也可以是關鍵點，滿足s代表狀態的子樹。我們記結點u的遮罩為u.mask，如果u是關鍵點，則u.mask表示u在二進位中對應的比特位，否則u.mask為0。

考慮到對於某一株樹的根結點，根結點下有三種情況：

1. 根結點沒有子結點
2. 根結點有一個子結點
3. 根結點有多個子結點

因此我們也可以根據三種情況建立狀態轉移公式。

1. dp(i,s) = s == i.mask ? 0：inf
2. dp(i,s | i.mask) = min(dp(i,s | i.mask) ,dp(j,s)+(i,j).w)
3. dp(i,s)=min(dp(i,s),dp(i,s′ | i.mask)+dp(i,(s − s′) | i.mask))

其中(i,j).w表示從i到j邊的權重。

這裡稍微說明一下複雜度。總共有V⋅2K個dp狀態，而每個狀態根據狀態轉移公式，時間複雜度如下：

1. O(1)
2. 可以通過在擁有相同狀態的dp點上跑SPFA。時間複雜度攤還為O(E)，如果邊權非負，你還可以利用Dijkstra來將攤還時間複雜度降低到O(V)。
3. s的子集數目為2B(s)，其中B(s)表示s中1的數目。由於2i=(1+2)K，因此攤還時間複雜度為O(1.5K)。

因此對於任意一個dp狀態所需的時間複雜度為O(V+1.5K)，總共有V⋅2K個dp狀態，因此總的時間複雜度為O(V(V⋅2K+3K))。

目前已有的解法：

[3]定義：給一個連通無向圖G=(V(G),E(G))，和邊權w：E(G)→R>0。現在有一個點集K⊆V(H)，求一個邊權和最小的連通子圖H，使得K⊆V(H)。其中H就被稱為斯坦納樹。

暴力解法[3]：在一個放棄思考的情況下，不難得出一個暴力的解法：直接枚舉點子集，而後跑個最小生成樹。這樣做的複雜度是O(2n˙nO(1))，其中n=|V(G)|。

動態規劃-狀態壓縮:[4]

參考資料：

[1] 費馬點，維基百科，<https://zh.wikipedia.org/wiki/%E8%B2%BB%E9%A6%AC%E9%BB%9E>

[2] 斯坦納樹，Daltao，

<https://taodaling.github.io/blog/2019/04/09/%E6%96%AF%E5%9D%A6%E7%BA%B3%E6%A0%91/>

[3]知乎-算法笔记-容斥原理与子集卷积（三），好地方bug，<https://zhuanlan.zhihu.com/p/33293771>

[4] 动态规划：状压DP-斯坦纳树，osc\_61rodd1h <https://my.oschina.net/u/4409491/blog/3868610>