**作業十二挑戰題報告**

**103062224 劉哲宇**

**A. 作法來源**

2-way BFS是上課教的，從尾巴做回來只要做一次的想法是由以下網址啟發的。

http://blog.csdn.net/shuangde800/article/details/7785984

**B. 說明是否有參考它人程式碼?**

無。

**C. 時間分析**

**0. 符號定義**

*S*: 需要走幾步，最高為16。

V: BFS時會拜訪的點，最高為 4^(S/2)

**1(a) 計算量: O( V \* (log\_2(V) \* 2) + V \* ( 2 \* log\_2(V) ) \* log\_2(V) )**

**1(b) 簡單說明**

\* 核心精神是2-way BFS。

\* 一個state 紀錄著移動了幾步、移動的整個過程、現在的轉盤排列方式。『重複的state』代表兩state的轉盤排列方式相同。

\* 由於終點是固定的，因此從終點做完回來的BFS只要做一次就好。

\* 因為同一個方向重複做相同的state很浪費時間，且不能用STL，所以我手刻了一個binary search tree (BST)來存state，可以用log2的時間找到是否有重複，也可以順便用來找答案。

\* 一開始先從終點做回來一次BFS，每次從BFS queue挖state出來後產生新的state，先丟進從終點的BST看看有沒有重複，若有則捨棄，若無則加入BST中，作到步數超過8時停止。最慘 O(**V \* (log\_2(V) \* 2)**)

\* 讀進input之後從起點做也是一樣的流程，不過是放在從起點出發的BST中，並且每次從BFS queue挖出state後，會去從終點出發的BST中尋找重複，若有就代表這個從起點出發的state跟某個從終點出發的state重複了，表示找到答案：**V \* ( log\_2(V) \* 2 ) \* log\_2(V)**

\* 若找到答案，透過這兩個state輸出答案。若整個從起點出發的BFS，一直到搜索的移動步數超過8都沒找到答案，則代表無法在16步內解決，則輸出 “NO SOLUTION WAS FOUND IN 16 STEPS” 。

**2. 驗證計算量**

由程式跑出的結果顯示，扣掉重複的步驟後 *V* 最高不超過 10^4，即以 10^4計算：

**V \* (log\_2(V) \* 2) + V \* ( 2 \* log\_2(V) ) \* log\_2(V) ≒ 3.8 \* 10^6**