

Introduction to AI Project #1 Report

0516072 資工 09 洪立宇

- Thinking

在一拿到 project 看到題目時，首先會最先想到的方法便是使用 dfs，並且用 stack 下去實作。然而因為一開始拿到的 3000 單字長度不一，並且是以 a~z 的單字編排的，所以在思考演算法時，就有想法應該要如何 sort 這些單字來讓我們的 dfs 更快、用更少 node 來找到我們所需要的解。

- Observation and To Do

所以透過觀察我發現我們可以用有 cross 的字和相同長度的規則去 sort 我們的單字表，這兩種方式可以當作兩種不同的 heuristic function 來幫助我們更接近、更快找到解答。於是我們決定做實驗，(一)不 sort 的結果下去跑 dfs (二)用 cross 的字母數量下去跑 dfs(三)用同樣長度的單字來跑 dfs。

- Experiment I (Do dfs without sorting)

從以下的圖片可以看到我們在沒有 sorting 情況下所做的 dfs，也就是沒有使用任何 heuristic function 來幫助減少我們的 node 使用量，我們可以觀察到當輸入條件變多的時候，因所受的條件增加，也會增加搜尋的深度所以會使得搜尋很沒有效率，並導致需要很多 node 產生的結果。

```
C:\WINDOWS\system32\cmd.exe
0 0 4 A 0 0 5 D 2 0 4 D 2 2 2 A 0 3 3 A
a b l e _
b _ a _ _
o _ t o _
u s e _ _
t _ _ _ _
Node generated: 14
0 2 5 A 1 1 5 D 3 0 7 D 1 4 5 A 3 6 4 A 5 4 3 D
_ _ c _ _ _ _
_ a l _ _ _ _
a b o u t _ _ _
o _ s _ _ _ _
v i t a l _ _
_ e _ e _ e _
_ _ r a t e _
_ _ _ _ _
Node generated: 26
2 0 4 A 2 0 5 D 5 0 5 D 0 1 3 D 0 2 6 A 2 4 4 A
_ _ a b l e
a _ b _ _ a
c l o s e r
t _ u _ l
_ t h e y
_ _ _ _ _
Node generated: 201
0 0 5 A 0 0 7 D 4 1 4 A 6 0 2 D 4 0 8 D 7 1 4 D 0 3 8 A 0 5 7 A 1 7 4 A 2 3 5 D 6 5 2 D 6 6 2 A
a b o u t a _
b _ _ e a s e _
i _ _ e _ d _
l e a r n i n g _
i _ b _ a _ t e _
t h o u g h t _
y _ v _ e _ o f _
_ b e a r _ _ _ _
Node generated: 7520
```

● Experiment II (Do dfs with cross-count-heuristic)

此實驗是透過計算和目前的字有剛好疊在同一個格子(交集)並進行排列，我們可以發現對於條件少的問題時，我們需要比正常沒有 sorting 還要花更多 node，而當條件變多的時候反而是相對需要比較少 node 的。可以發現，在小運算量時，經過 heuristic 不一定會需要比較少 node，但是在平均或者是大運算量的時候，可能 heuristic 後表現會更好一些。

```
for(auto &i:w){
    int tmp_x=i.x,tmp_y=i.y;
    for(int j=0;j<i.len;j++){
        i.cro += cross[tmp_x][tmp_y] - 1; // count cross heuristic
        if(i.dir=="A"){
            tmp_x++;
        }
        else {
            tmp_y++;
        }
    }
}
```

```

C:\WINDOWS\system32\cmd.exe
0 0 4 A 0 0 5 D 2 0 4 D 2 2 2 A 0 3 3 A

a w a y _
b _ e _
w _ i _
s _ a _
e _ _ _

Node generated: 2910
0 2 5 A 1 1 5 D 3 0 7 D 1 4 5 A 3 6 4 A 5 4 3 D

_ _ c _ _ _
a b o u t _ _ _
o _ s _ _ _
v _ i _ t _ a _ l _ _ _
e _ e _ e _ _ _
_ _ r _ a _ t _ e _ _ _
_ _ _ _ _ _

Node generated: 2922
2 0 4 A 2 0 5 D 5 0 5 D 0 1 3 D 0 2 6 A 2 4 4 A

_ _ a l s o _
b _ g _ _ r _
a b r o a d _
d _ e _ _ e _
_ _ e _ v _ e _ r _
_ _ _ _ _ _

Node generated: 5749
0 0 5 A 0 0 7 D 4 1 4 A 6 0 2 D 4 0 8 D 7 1 4 D 0 3 8 A 0 5 7 A 1 7 4 A 2 3 5 D 6 5 2 D 6 6 2 A

a l b u m _ a _ _
b _ _ _ a l s o _
a _ _ _ j _ _ n _
n a t i o n a l _
d _ h _ r _ _ y _
o p e n i n g _
n _ i _ t _ o _ f _ _
_ _ a _ r _ m _ y _ _ _ _
_ _ _ _ _ _

Node generated: 6566

```

- Experiment III (Do dfs with same-length-word-heuristic)

此實驗是用相同長度的字去做排列，我們可以發現在條件少的情況下，效果比沒有做 **sorting** 的還要好，但是一但條件變多了，所需要的運算量會變大很多，可能是因為後面的字字數都比較多，所以會需要花比較多的計算才能夠找到我們的答案。

```

C:\WINDOWS\system32\cmd.exe
0 0 4 A 0 0 5 D 2 0 4 D 2 2 2 A 0 3 3 A

a l s o _
h _ e _ _
e _ a _ d _
a _ c _ t _ _
d _ _ _ _ _

Node generated: 10
0 2 5 A 1 1 5 D 3 0 7 D 1 4 5 A 3 6 4 A 5 4 3 D

_ _ _ a _ _ _ _
_ _ t _ b _ _ _ _
a h e a d _ _ _ _
_ _ e _ n _ _ _ _
_ _ m e d i a _ _ _
_ _ e _ o _ c _ _ _
_ _ _ n o t e _ _ _
_ _ _ _ _ _

Node generated: 51
2 0 4 A 2 0 5 D 5 0 5 D 0 1 3 D 0 2 6 A 2 4 4 A

_ _ a r e a _
a _ b _ _ g _
c h o i c e _
t _ u _ _ n _
_ _ t e n t _
_ _ _ _ _ _

Node generated: 67
0 0 5 A 0 0 7 D 4 1 4 A 6 0 2 D 4 0 8 D 7 1 4 D 0 3 8 A 0 5 7 A 1 7 4 A 2 3 5 D 6 5 2 D 6 6 2 A

t o p i c _ a _ _
e _ _ _ r i d e _
s _ _ _ e _ _ a _
t e e n a g e r _
i _ a _ t _ _ n _
f o r m u l a _
y _ l _ r _ h e _
_ _ t y p e _ _ _ _
_ _ _ _ _ _

Node generated: 32493

```

● Conclusion

透過以上的實驗我們可以發現，在 **sorting** 之後雖然表現不一定會比在沒 **sorting** 之前好，可是在某些情況下，透過不同 **heuristic function**，我們可以更有效率的找到我們的解答。**cross-count-heuristic** 就比較適合多條件的問題，**same-length-word-heuristic** 則是對少條件的問題比較有利，所以在遇到不同的問題時我們需要多嘗試一下不同的 **heuristic** 方法，也許我們會有機會找到更適合本問題的方法。所以花時間去測試、實驗是有必要的。

● Code

C:\Users\Owner\Desktop\hw_1.cpp - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

```
hw_1.cpp
1  #include<iostream>
2  #include<bits/stdc++.h>
3
4  using namespace std;
5
6  vector<string>word[50];
7  struct word_block{
8      int x;
9      int y;
10     int len;
11     string dir;
12     int cro;
13     friend bool operator < (const word_block& a,const word_block& b){
14         //return a.cro > b.cro; //heuristic with cross amount
15         return word[a.len].size() < word[b.len].size(); //heuristic with word with same length
16     }
17 };
18
19 char a[50][50]; //the board
20 int cross[50][50]; // count the cross
21 int max_x,max_y,flag,node_cnt,ans_cnt;
22 set<string>used; //check for the word used or not
23 vector<word_block>w; //word content
24
25 void dfs(int n){
```

C:\Users\Owner\Desktop\hw_1.cpp - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

```
hw_1.cpp
49 }
50 else{
51     ys = 1;
52 }
53 for(auto i:word[len]){ // find through all words consist current needed length
54     if(used.find(i)!=used.end())continue; // find unused words
55     int found=0;
56     int nx=x,ny=y;
57     for(int j=0;j<len;j++){
58         if(a[nx][ny]!='_'&&a[nx][ny]!=i[j]){ // check if there is a space for the word
59             found=1;
60             break;
61         }
62         nx = xs+nx;
63         ny = ys+ny;
64     }
65     nx=x,ny=y;
66     if(found==0){
67         vector<pair<int,int> >sp;//assigned the letters into the node
68         nx=x;
69         ny=y;
70         for(int j=0;j<len;j++){
71             if(a[nx][ny]=='_'){
72                 sp.push_back({nx,ny});
73                 a[nx][ny]=i[j];
```

C:\Users\Owner\Desktop\hw_1.cpp - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

```
hw_1.cpp
97 }
98 while(getline(cin,s)){
99     stringstream ss(s);
100     int x,y,len;
101     string dir;
102     max_x=0;
103     max_y=0;
104     w.clear();
105     while(ss >> x >> y >>len >>dir){ // get the input x, y, length, direction required
106         w.push_back({x,y,len,dir,0});
107         max_x=max(max_x,x+(dir=="A")*len);
108         max_y=max(max_y,y+(dir=="D")*len);
109         int tmp_x=x,tmp_y=y;
110         while(len>0){ //check the cross from the words
111             cross[tmp_x][tmp_y]++;
112             if(dir=="A"){
113                 tmp_x++;
114             }
115             else{
116                 tmp_y++;
117             }
118             len--;
119         }
120     }
121     for(auto &i:w){
122         int tmp_x=i.x,tmp_y=i.y;
```