Introduction to AI　Project #1 Report

0516072 資工 09 洪立宇

- Thinking

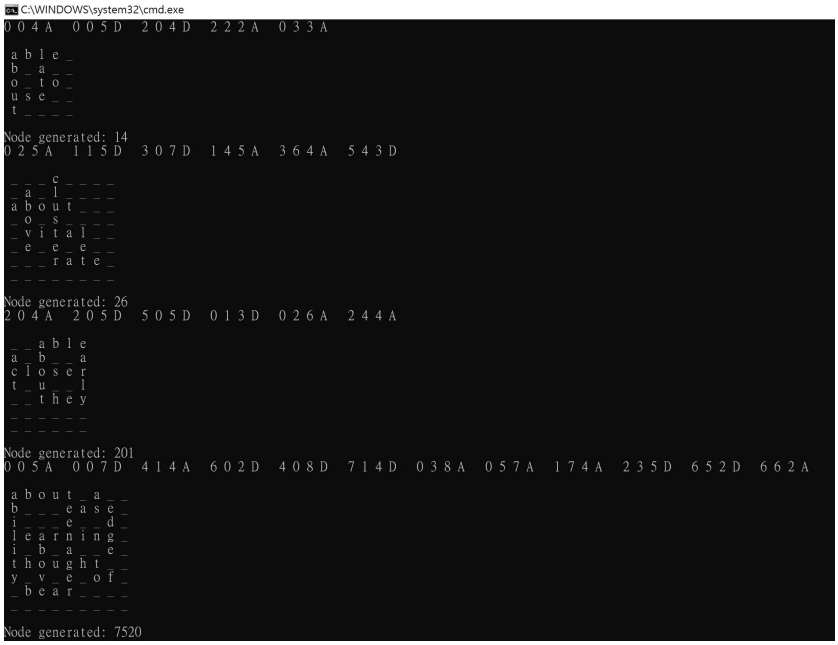
  在一拿到 project 看到題目時，首先會最先想到的方法便是使用 dfs，並且用 stack 下去實作。然而因為一開始拿到的 3000 單字長度不一，並且是以 a~z 的單字編排的，所以在思考演算法時，就有想法應該要如何 sort 這些單字來讓我們的 dfs 更快、用更少 node 來找到我們所需要的解。

- 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)

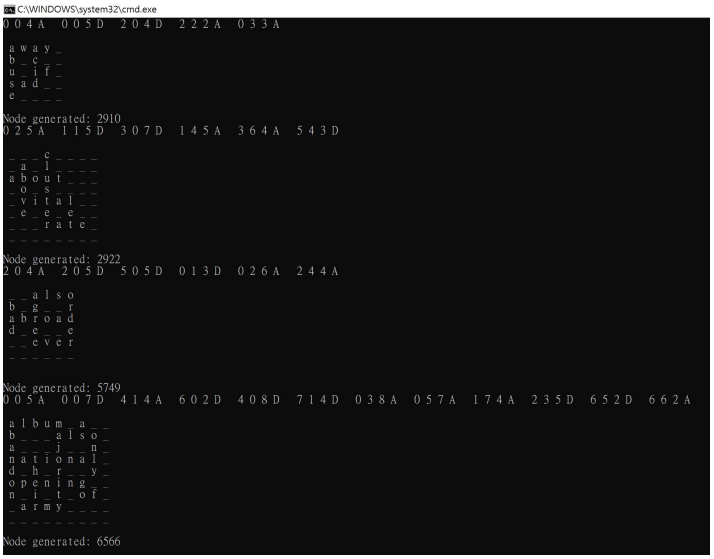
  從以下的圖片可以看到我們在沒有 sorting 情況下所做的 dfs，也就是沒有使用任何 heuristic function 來幫助減少我們的 node 使用量，我們可以觀察到當輸入條件變多的時候，因所受的條件增加，也會增加搜尋的深度所以會使得搜尋很沒有效率，並導致需要很多 node 產生的結果。



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

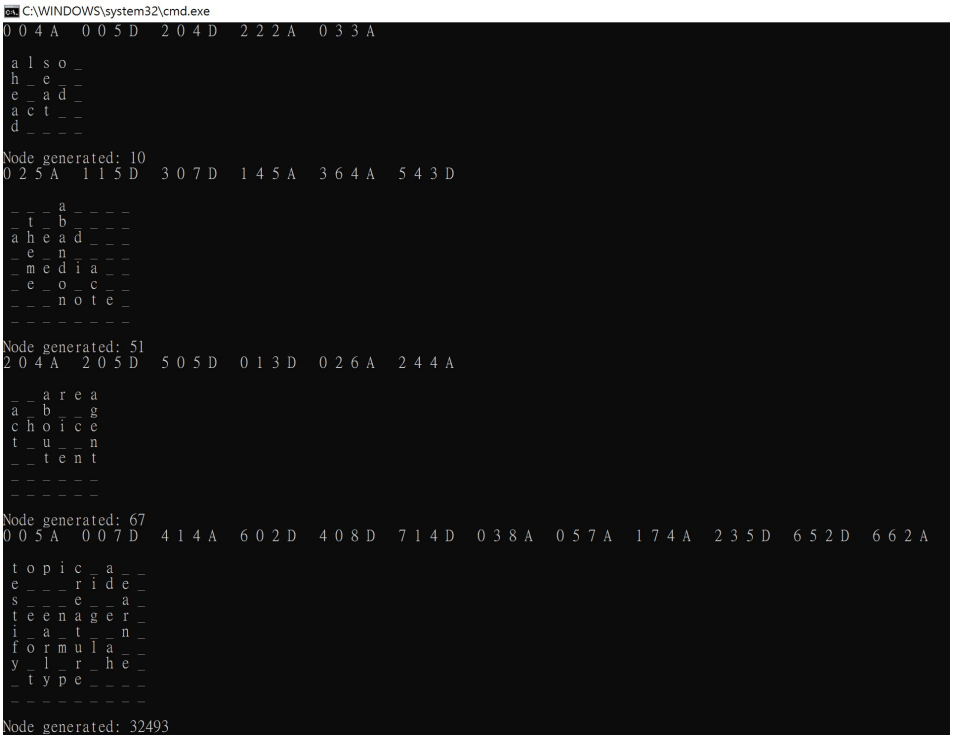
  此實驗是透過計算和目前的字有剛好疊在同一個格子(交集)並進行排列，我們可以發現對於條件少的問題時，我們需要比正常沒有 sorting 還要花更多 node，而當條件變多的時候反而是相對需要比較少 node 的。可以發現，在小運算量時，經過 heuristic 不一定會需要比較少 node，但是在平均或者是大運算量的時候，可能 heuristic 後表現會更好一些。

```
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++;
        }
    }
}
```



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

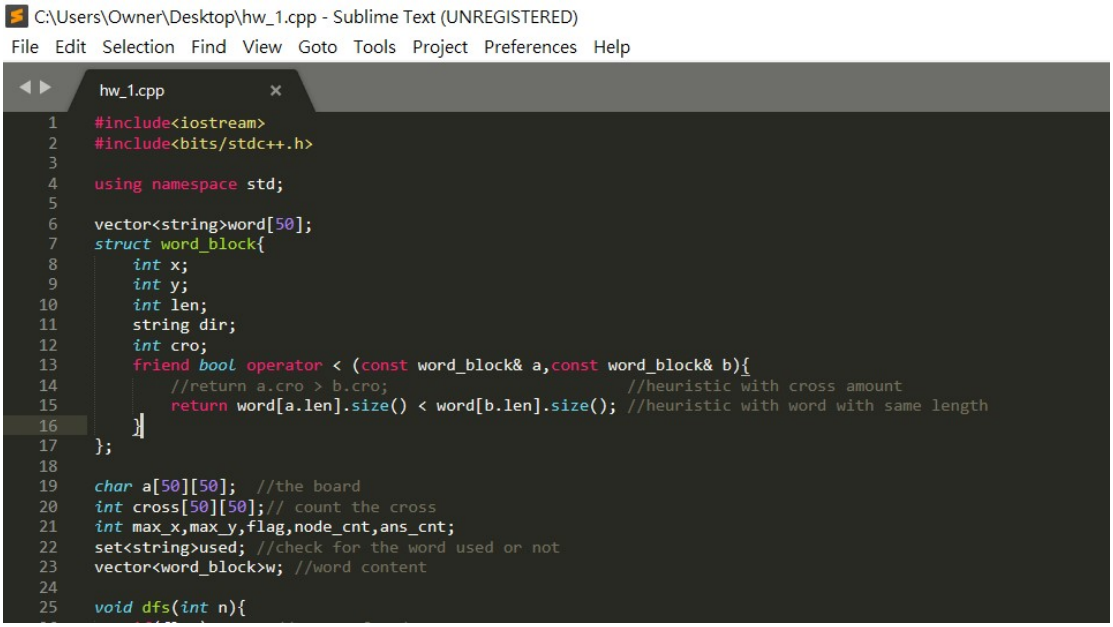
  此實驗是用相同長度的字去做排列，我們可以發現在條件少的情況下，效果比沒有做 sorting 的還要好，但是一但條件變多了，所需要的運算量會變大很多，可能是因為後面的字字數都比較多，所以會需要花比較多的計算才能夠找到我們的答案。



- 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

```
49      }
50      else{
51          ys = 1;
52      }
53      for(auto i:word[len]){// find throught 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){
```