Essay Search

110062239 侯茹文

1. Implement

我是以 trie 來完成這次的 project。一個 trie 可以完成 prefix search 以及 exact search 的部分。

因此可以以 trie 直接執行 prefix search 以及 exact search。因為 trie 會設立一個 tree 存一篇論文會有的字，因此我在 trienode 中加入一個布林值，來記錄這個 node 是否為一個字的結尾。Prefix search 在執行時就直接深入 trie，若可以跑到 prefix search 的最後一個字母，則回傳 true。Exact search 與 prefix search 相似，不同的地方是 exact search 是回傳剛剛提到記錄這個字是否為結尾的布林值，這樣才可以確定是否為剛好一個字。

而 suffix search 的部分我是在一個 trie 中建立另一個 root，使他有兩個 root，一個存正常順序的字，一個存順序相反的字(在 insert 時加另一個 for 迴圈跑相反的方向)，suffix search 時就跑相反的那個 root。

而建完許多 trie 後，我以 vector 存全部論文的 trie，再 open query 的 file。然後先輸入第一行的 query 後開始判斷第一篇論文使否符合要求，若符合就加入 answer 的 vector，而後再判斷下一篇論文，直到結束所有論文即開始判斷下一個 query。

2. Challenges

我認為我做最久的是 suffix search 的部分，想了一陣子後才想到要用 inverse root 去做，因為還另外加上一些東西，而我一直忘東忘西，debug 就花了一些時間，算是我自己太笨。

還有一個處理很久的問題是我在處理 " " 和 * * 時，一開始我先將 " " 和 * * 刪掉，忘了它會被處理許多次（在 n 篇論文中，判斷一個相同的 query），所以在第一次處理 query 中被切的字元時就把** 和 “” 刪掉了，因此在其他篇論文中判斷時** 和 “” 就會消失，這個 bug 我也找很久。

3. Reference

https://haogroot.com/2021/01/07/trie-leetcode/

https://chat.openai.com/chat

我參考了這個網址以及問了 openai 許多問題，它也幫助我許多。