110062103_李承叡_project 報告

1. Minimax 與 alpha-beta pruning 的實作
   我的五子棋 AI 是採用 Minimax 作為基底，並用 alpha-beta 剪枝法來加速運算。Minimax 會模擬我方與對方的棋路，當模擬我方時，會找出所有可能的棋路中對我方最好的局面，而當模擬對方時，則會算出對我方最壞的局面來作為對敵人下一步的猜測。因此，根據層數的多寡，當要選出本次下棋的步數時，會先假設下了一步棋，然後模擬對方的再下一步棋，依序這樣的互相模擬之後，找出綜合分數最高的一步。
   A. 優化：限縮棋路
      下棋時，15X15 的棋面一定會有某些位置是不可能下出好棋的，尤其是離其他棋很遠的旗子，因此我們可以預先排除這些位置，將可以下得位置限制於必須再 8 個方向中有其他旗子存在，如此一來就可以省去許多不必要的模擬
   B. 優化：alpha-beta-prunig
      若是敵方已經找到一個小值(beta)，就不可能會再往下推演比 beta 更大，由下一層玩家所找到的最大值(alpha)，反之在玩家推演時，也不會選擇 beta 比 alpha 還要小的情況。因此若們我們率先把 alpha 比 beta 大的狀況先刪除，就可以省去不必要的運算時間。
2. 評估函數
   我評估盤面的方法是計算整個盤面上有多少死二、活二、活三等等，若可以下出連五，則將此盤面回傳 INT_MAX，反之若是對方下出連五，則回傳 INT_MIN 來避免下出這步棋，除此之外，將我方所擁有的棋型數量減去對方的棋型數量，並分別乘以一定的權重，最後的總合就是這個盤面的分數。
3. 優化: Zobrist hashing
   在模擬的過程中可能會遇到相同的盤面，為了避免重複的運算，我運用 Zobrist hashing 的技巧，先將一開始讀進來的盤面設定一個隨機 hash 值，每當有新的更動時就會更動 hash，因此若有重複的版面出現，就可以根據他的 hash 值從 map 搜索是否層算過該版面。
4. 優化: 排序節點
   我利用一個評估函數來計算一個空白點的好壞，並只取分數前十高的點進行推演，大幅減少了計算時間。

```
commit 3365de0d6ad7c86cf552e4ba75cf1e42d9bc4877
Author: KappaBarbarosa <ayaba7077@gmail.com>
Date:   Sun Jun 19 16:21:48 2022 +0800

    can_win_baseline1_and_2

commit b58180d4f65af7c7eec377a401890b9f18d56a76
Author: KappaBarbarosa <ayaba7077@gmail.com>
Date:   Fri Jun 17 17:54:09 2022 +0800

    smarter?

commit 0fd0d57e1b2e5852b88ded73452d88bb7850633e
Author: KappaBarbarosa <ayaba7077@gmail.com>
Date:   Thu Jun 16 16:44:34 2022 +0800

    smarter_but_childlike

commit b8ad24c45683070ac5e945eabb2d92db71032624
Author: KappaBarbarosa <ayaba7077@gmail.com>
Date:   Thu Jun 16 16:15:24 2022 +0800

    test
```