Version Control 截圖

一張含有 文字, 螢幕擷取畫面, 膝上型電腦, 電腦 的圖片

自動產生的描述一張含有 文字 的圖片

自動產生的描述

把main.cpp的Othello Class 複製到我的.cpp中

Tree Search (Minimax) and Alpha-beta Pruning

設最多計算到六層就要回傳board value

都要用minimax找尋下一步棋後的board value

如果是奇數層(代表己方下棋)選max board value，再把自己的children中最大的board value傳給 alpha(走對自己最有利的棋) (如果比alpha大才傳，否則不變)

如果是偶數層(對方下棋) 選 min board value，再把自己的children中最小的board value 傳給 beta (因為敵人預設走對我們最不利的棋)(如果比beta小才傳，否則不變)

若alpha>=beta => prune

一張含有 文字 的圖片

自動產生的描述

Alpha-beta Pruning

一張含有 文字 的圖片

自動產生的描述

Write valid spot包含第一層alpha beta pruning。

用it遍歷所有下一步產生的board 。 如果board已經結束，直接回傳board value，不然就丟到minimax找到這個board value。

最後把最大的board value 的下棋點存起來後輸出。

Alpha也要在第一層進行prunining，把回傳值m傳給alpha存放。

State Value Function Design

算每一步產生的棋盤對己方有利的程度

Black棋子在棋盤得到的value – white棋子在棋盤得到的value

計算方式:

下一步可移動的位子 \* 3 + 此顏色棋子在棋盤的數量\*5 + 若有棋子在角，一個角加50點 + 若有此顏色棋子在初始棋周圍，加四點 + 若有旗子在邊邊(不包括角周圍點)，加八或六點 – 如果有棋子在corner周圍，卻不在那個corner，剪20 點 (因為很容易被敵人佔到角)

最後自己顏色棋子的value – 敵人顏色棋子的value = board對自己有利程度。

一張含有 文字 的圖片

自動產生的描述

一張含有 桌 的圖片

自動產生的描述

一張含有 桌 的圖片

自動產生的描述

一張含有 桌 的圖片

自動產生的描述