AI_11_22

Minimax Decision

最大話與最小化利益

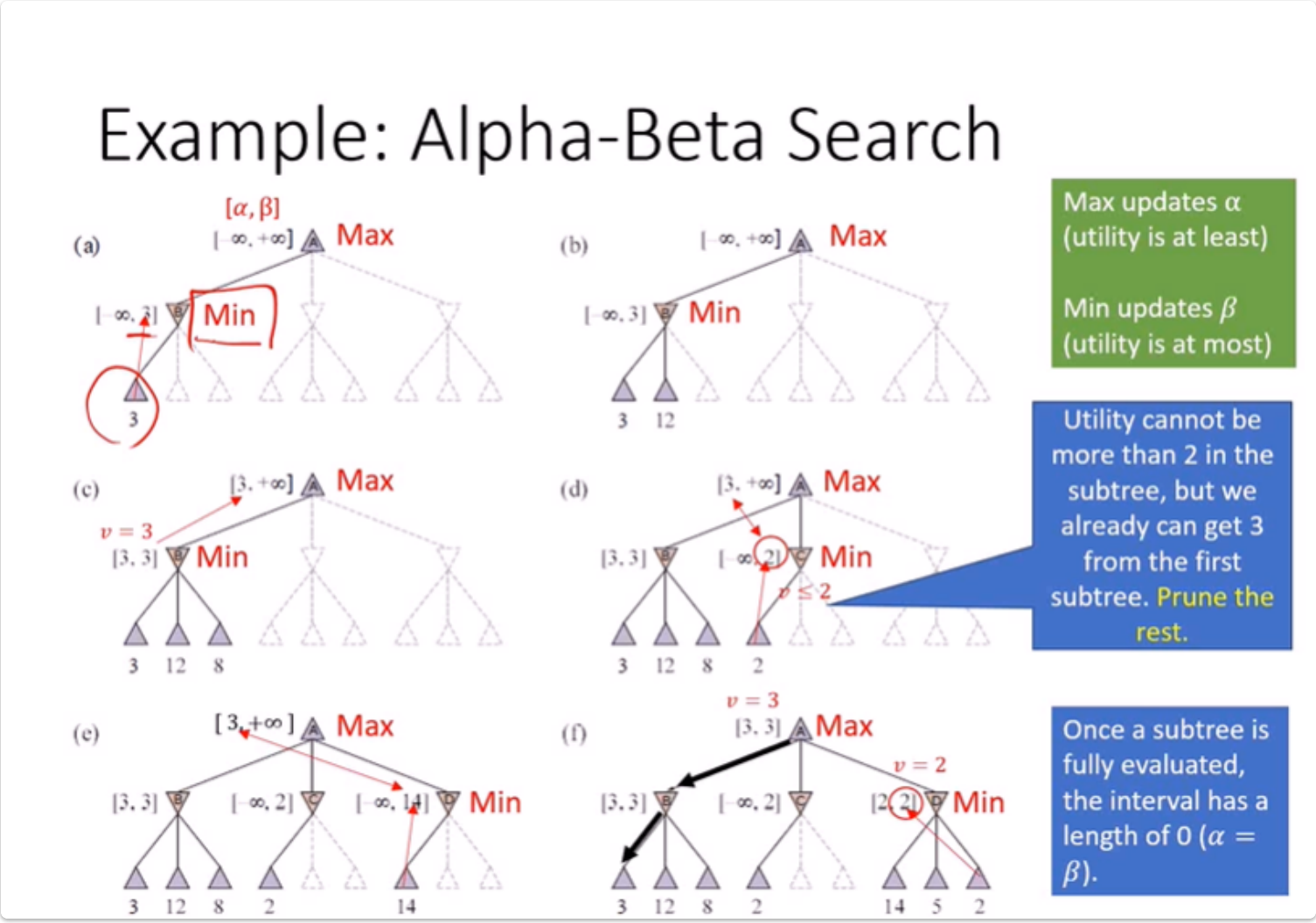
Determining MV Values

定義每個步驟的分數
MAX就選最大的
MIN就選最小的

Alpha-Beta Pruning

$min(3, x, y)$ -> 答案必定 <= 3
$max(5, min(3, x, y, \dots))$ -> 答案必定是5

所以已經知道結果了，就不用繼續搜尋下去



Move Ordering for Alpha-Beta Search

- 搜尋順序重要
- 因為如果很不幸運的，你可以判斷不用判斷的屬性在最後面，這樣就必須多花很多步驟，所以才說順序很重要，順序對了就可以減少很多搜尋
- 要做一個好的搜尋非常不容易

Heuristic Alpha-Beta Search

Cutting off search

Stop search at a non-terminal node 可以停止搜尋在非最終點
Use heuristic evaluation function 用預估函示去評估功能

- Fast to compute
- 要在 win 跟 lose 中間

- 把實際跟勝率去做比對，像是可以加上權重
  可能會不小心把重要的路徑篩掉

Forward pruning

必較爛的步驟，就不要去做
定義的評估

- 預估值小
- 做淺層的搜尋 (cut-off search) 值比較小的
- 過去的經驗
  可能會不小心把重要的路徑篩掉

Monte Carlo Tree Search

會選下面部分節點，看誰效果最好，到終點那種，就選誰
算部分的，預測到底誰的效果好

怎麼挑

- 隨機
- 預估 (由現實專家的經驗去預測)
- 使用學習的方式，建造一個模型去做

問題

- 分支太多
- 沒有一個好評估的方式

從哪個節點都可以，選當下可以選的節點們，然後把他們 run 完，選擇贏面最高的

可以從任何一個節點開始
確認這個節點的評估是否好
那就往下做
但他只是部分 而不是全部

Playout Selection Strategy

探索 vs. 選擇已經做過的

UCB1

UBC1(n) = $\frac{U(n)}{N(n)} + C\sqrt{}$
節點的數值越好 就選擇

Robotic

Locomotion 移動方式 關於硬體移動的部分
Navigation 導航 需要對外在的感知，去預測路徑

- Perception 感知
- Localization 定位
- Planning and motion generation 計畫跟移動的產生

室內 有結構的環境 人為的環境
室外 無結構的環境
自動化路徑機器人
醫院機器人
Khepera IV Robot
管路檢查機器人
農業機器人
昆蟲機器人

Control Scheme

1. 感知真實世界 (感測器)
2. 定位
3. 規劃
4. 移動的 command
5. 移動 (執行指令)
6. 回到 1. 回到 loop

感知 → 定位 → 認知 → 移動控制 → 真實世界