Pacman Homework Report

R12922A09
Yung-Hsiang Yang

1. Depth First Search

是一種用來遍歷樹(tree)或圖(graph)的演算法。其概念可以概括為一直沿著路徑向前只到無法在訪問，並返回之前的分岔路，繼續此一過程。最終持續到整個過程無法在繼續，或是到達要搜尋的目標。由於 Stack 的性質 LIFO (Last In First Out)，DFS 也可以用遞迴的形式進行實現，但是 DFS 並不能保證當前找到的路徑為最短路徑。

步驟如下：

1. 選擇起始節點並將其標記為已訪問。
2. 將起始節點放入 Stack。
3. 當 Stack 不為空時，執行下列步驟：
   a. 從 Stack 取出一個節點，將其標記為已訪問
   b. 檢查該節點的相鄰節點中是否有未造訪過的節點
   c. 如果有未訪問過的節點，選擇一個未訪問過的相鄰節點，將其標記為已訪問，並將其放入 Stack
   d. 如果所有相鄰節點都已經造訪過，繼續從 Stack 取出下一個節點
4. 重複步驟 3，直到 Stack 為空

2. Breath First Search

與 DFS 一樣，也是一種用來遍歷樹或圖的演算法，與 DFS 不同的是，BFS 優先訪問鄰近的節點，然後是與這些節點相鄰的所有節點，以此類推。BFS 相較於 DFS 的深度，頃向於廣度，由於使用 Queue 來儲存節點，BFS 保證了在同一層級的節點被先訪問，因此它可以用於找到最短路徑。

步驟如下：

1. 選擇起始節點並將其標記為已訪問
2. 將起始節點加入 Queue
3. 當 Queue 不為空時，執行下列步驟：
   a. 從 Queue 取出一個節點
   b. 對於從 Queue 取出的節點的每個未存取的相鄰節點，將其標記為已存取並將其放入 Queue
4. 重複步驟 3，直到 Queue 為空。

3. Uniform Cost Search

UCS 是一種用於在加權圖中尋找最低成本路徑的圖搜尋演算法。與 BFS 同，UCS 考慮了邊的權重，而不是簡單地考慮節點之間的連結。實作上來說與 BFS 無異，如果能保證 BSF 的通用性，只要將 Queue 改成 Priority Queue，並考慮節點之間的 cost 即可。UCS 給出了到達目標節點的最佳解或路徑。

步驟如下：

1. 將起始節點放入 Priority Queue（按 cost 排序），並將起始節點的 cost 設為 0。
2. 從 Priority Queue 中取出具有最低成本的節點。
3. 對於被取出的節點，檢查其所有未訪問的相鄰節點，並計算到每個相鄰節點的路徑成本。如果該路徑 cost 比已知的路徑 cost 低，則更新路徑成本並將相鄰節點插入 Priority Queue。
4. 重複步驟 2 和 3，直到找到目標節點或 Priority Queue 為空。

4. A star Search

A star 的評估表示為 $f(n) = g(n) + h(n)$

$g(n)$ 表示從起點到任意頂點 n 的實際距離，$h(n)$ 表示表示任意頂點 n 到目標頂點的估算距離，這個估算距離也可以視為兩點之間的 cost。以老鼠走迷宮為例，老鼠需要找尋在迷宮中的起司，$g(n)$ 表示為老鼠與起點間的距離，而 $h(n)$ 為起司的氣味，在各個節點間有不同的數值用來表示氣味的強烈程度。

實作方面也是以 UCS 作一些修改，在計算 cost 也把 $h(n)$ 考慮進去得到新的 cost 並以此作為優先度將節點放入 Rripority Queue 即可。

7. Grade