2048 AI要做…

(1)

在一個2048的棋盤狀態，輸出一個好的移動方向

(2)

在多次移動後可以得到一個好的盤面

(3)

一個好的棋盤狀態(by網友經驗):

1順序排列：

不同大小的數字需要順序排列，由左至右，由上至下，或反過來，或一條蛇型都可，只是數字是順序連在一起

2三角排列：

將方塊歸在左面和下面，不同大小數字散射出去

3明確路徑：

有1024之後，你要努力合成一個512；有512之後，要努力會合成一個256

(4)

遵循原則(by網友經驗)

1集中火力：

同一時間只拼一個數。例如你要拼256，就集中火力不斷合成把2和4，然合成8,16,32，然後8,16,32再合成64,128...。換句話說，不要左合一個32，右合一個64，上合一個16

2堅持順序：

盡量別讓不同大小的數字「不順序排例」，例如出現16,64,32這種情況

3堅守角落：

數字最大的方塊都應該保持在角落位置(Corner)。若果不慎移出，必須盡快歸回角落

4層層分解：

要合成2048，最理想的狀態是左下角一個1024，然後右邊512,256,128...

(5)

可以利用的資料結構和演算法

Heap, recursive, devide and conquer

TO SUM UP,

2048\_AI\_proposal

Idea:

4x4 --->混亂的情況

Max不再corner

不是由大到小照順序排

| | |

| | |

| | | {AI要做的}

| | |

V V V

4x4-🡪理想的情況

三角排列, like…

2 0 0 0

4 2 0 0

8 4 2 0

16 8 4 2

三角排列:

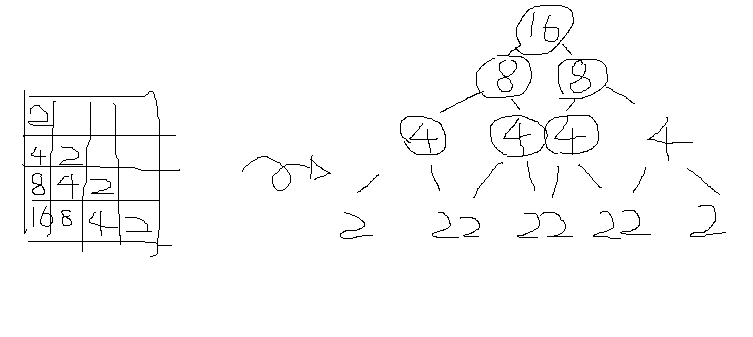
好處

移動不會被大數字卡住

有明確路徑

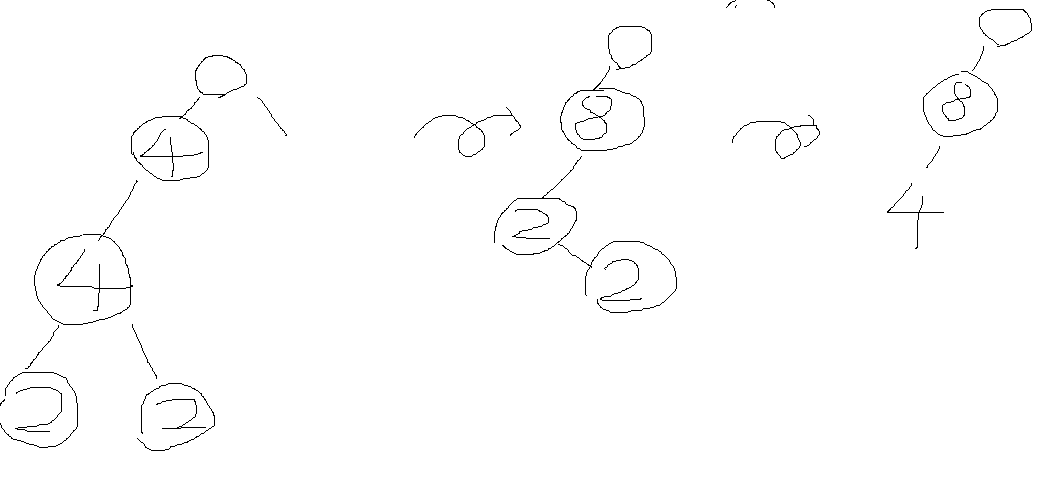
Datastructor

2048\_Binary\_Heap

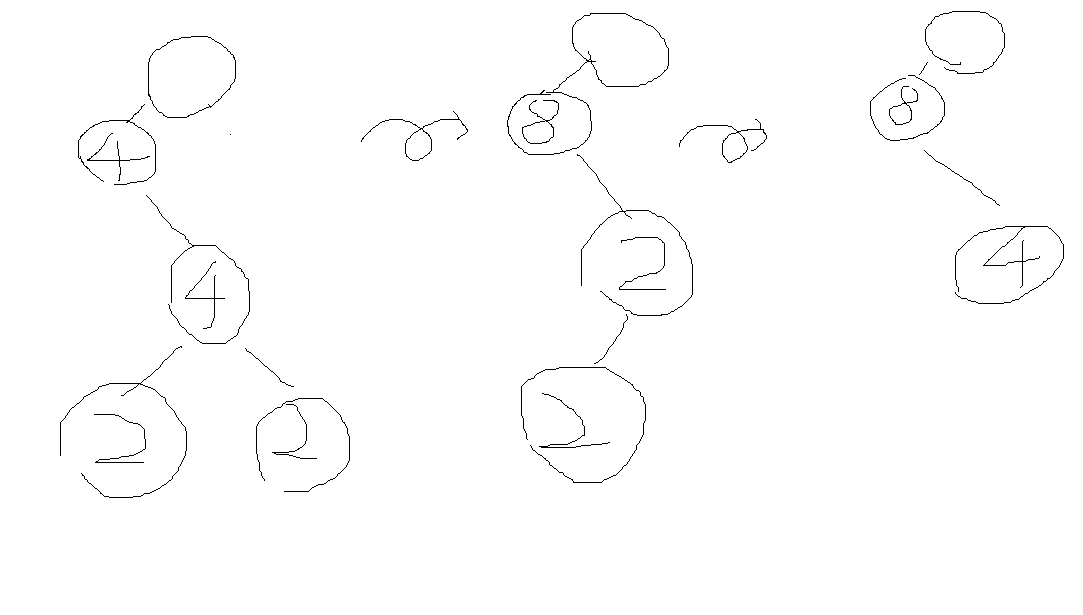


Merge\_And\_BubbleUp(node)

Case1.



Case2.



Function

利用recursive讓root(max)長大

MaxHeapUp(root){

//Base case

If(root.value == children.value)

Merge\_And\_BubbleUp(root) //root變兩倍大

//Other case

Else

MaxHeapUp(children) //讓root下面的children長大,直到等於root為止

}

混亂->理想

(1)max不在corner

讓max靠邊、靠角:

移max到邊角

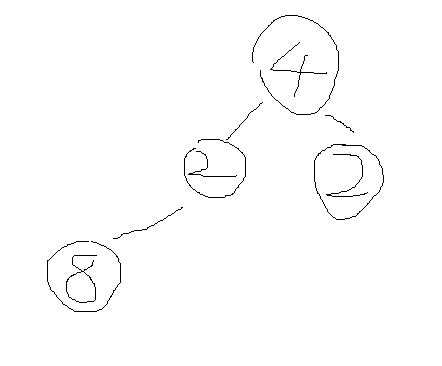
或讓在角落的數比current max還大

(2)not in order

讓較大的數向max方向推

像是heap的bubble up

小數字被較大的數卡住



要把8向上推

2048技巧參考http://wordstohealbyheart.blogspot.com/2014/04/2048\_6.html