

主題: BFS

- 基礎
- 應用
- 作業與自我挑戰

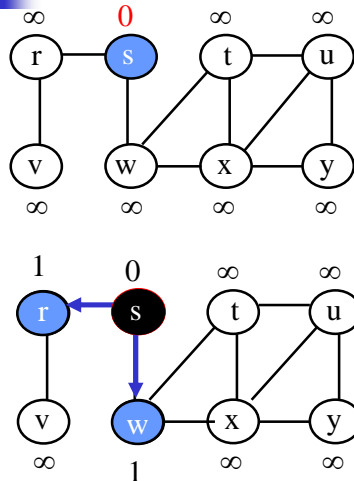
1

基礎

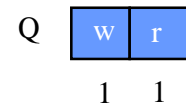
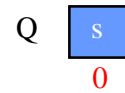
- BFS
- BFS vs. DFS

2

Breadth-first search

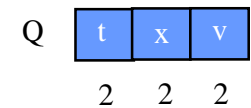
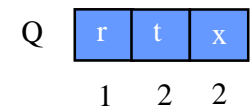
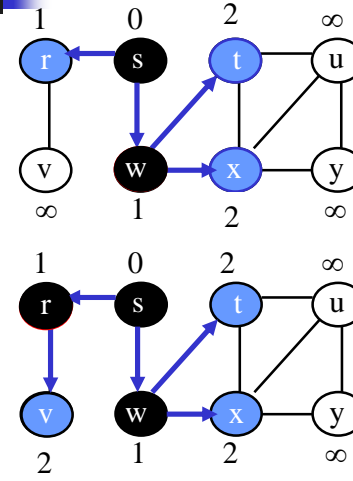


s is the source



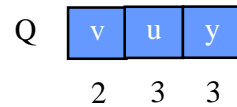
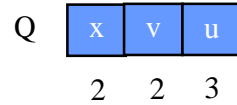
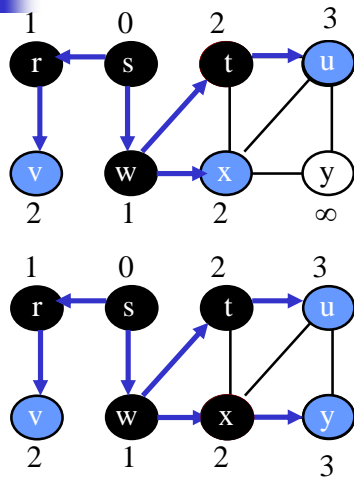
3

Breadth-first search



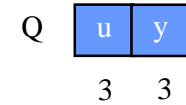
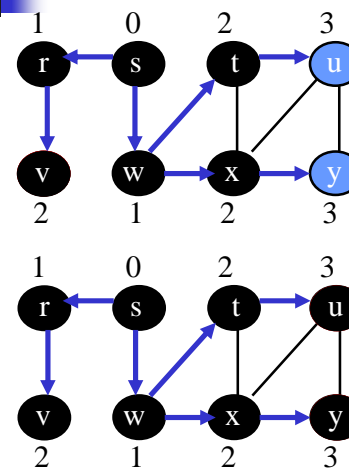
4

Breadth-first search



5

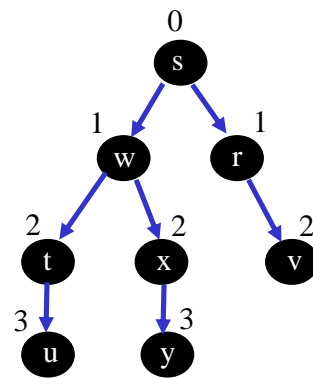
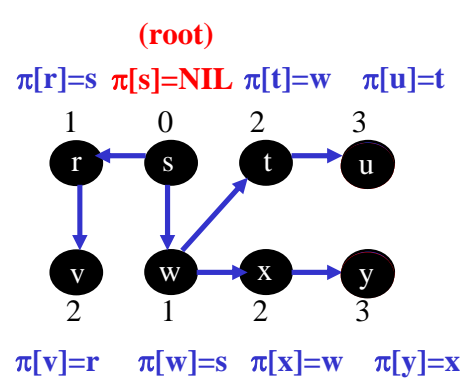
Breadth-first search



Q

6

Breadth-first search tree



7

Pseudo code

```

BFS(s) {
  for each  $v \in V$ ,  $\text{color}[v] = \text{WHITE}$ ;

   $\text{color}[s] = \text{GRAY}$ ;  $d[s] = 0$ ;  $\pi[s] = \text{NIL}$ 
   $Q = \emptyset$ ; Enqueue( $Q, s$ )
  while  $Q \neq \emptyset$  {
     $u = \text{Dequeue}(Q)$ 
    for each  $v \in \text{Adj}(u)$ 
      if  $\text{color}[v] = \text{WHITE}$  {
         $\text{color}[v] = \text{GRAY}$ 
         $d[v] = d[u] + 1$ ;  $\pi[v] = u$ 
        Enqueue( $Q, v$ )
      }
     $\text{color}[u] = \text{BLACK}$ 
  }
}

```

由 u 走過來
(for backtracking)

8

BFS 的 time complexity 與特色

- Time
 - Adjacency-matrix: $O(n^2)$
 - Adjacency-lists: $O(n+m)$
- BFS 的深度就是 source s 與所有 nodes 的最短步數
- BFS 的最大特色: 可以解決 unweighted graph 的 single-source shortest paths problem
- 如果要找 single-pair (s, x) 的 shortest path, 一旦 x 走到便可以 stop

9

BFS vs. DFS

- DFS
 - 主要用以計算 connectivity
 - 比較簡單好寫
 - recursive 的 stack 由系統提供
- BFS
 - 主要用來計算 unweighted graph 上的“最短距離”
 - 也可以計算 connectivity
 - need to maintain a queue

10

DFS 所需的 stack $> 10^6$ 怎麼辦???

- Method 1.
 - write a non-recursive version
 - maintain a stack by yourself
- Method 2.
 - use BFS (maintain a queue)

11

應用

- 應用一: H.89.2 老鼠找食物
- 應用二: A.808 Bee Breeding
- 應用三: A.571 Jugs
- 應用四: A.652 Eight
- 應用五: 大甲.2000.3 minimum distance in star graphs
- 應用六: A.816 Abbott's Revenge

12

應用一: H.89.2 老鼠找食物

- 給一個 $m \times n$ 的地圖，其中包含老鼠的位置，障礙物的位置及食物的位置，老鼠每次可上下或左右移動一格，依照距離由近到遠，列出牠從目前位置到每一個可以到得了的食物所在點，所需移動的最小步數

- $0 < m, n \leq 80$

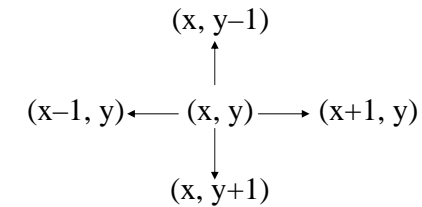
B						
	F ₃					F ₆
F ₄	B		M	F ₁	B	
	B					
		F ₂				
	B		B	B	B	
			F ₅			

M: 老鼠, B: 障礙物, F: 食物

13

地圖表示法

- 只需用一個 2-d array 來表示這個地圖上的 nodes



當每一個 node 的 neighbors 可以被“計算”出來時
 \Rightarrow 只需一個方法表示 nodes，不需表示 edges！

14

Solution

注意: 障礙物 (B) 不要扔進 Q

- M: 老鼠, B: 障礙物, F: 食物

B						
	F ₃					F ₆
F ₄	B		M	F ₁	B	
	B					
		F ₂				
	B		B	B	B	
			F ₅			

1 步: F₁

2 步:

3 步: F₂, F₃

4 步: F₆

5 步: F₄

6 步: F₅

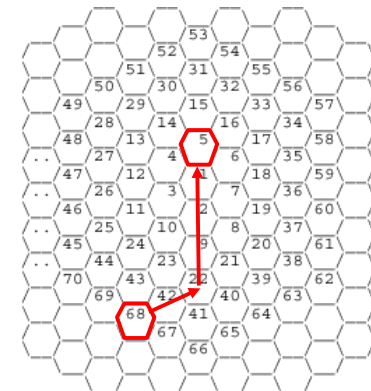
15

應用二: A.808 Bee Breeding

- 給 x, y 兩個 nodes，計算 x 與 y 的最短距離

- $1 \leq x, y \leq 10000$

- Example: $x = 68, y = 5$
 - $d(x, y) = 6$



16

Solutions

- Method 1.
 - 聰明的分析 id 的規律
 - 進一步找出 x, y 最短距離的計算公式
- Method 2.
 - 找出方法去計算一個 node i 的 6 個 neighbors
 - BFS
 - 由 x 出發, 看到 y 便可以停下來

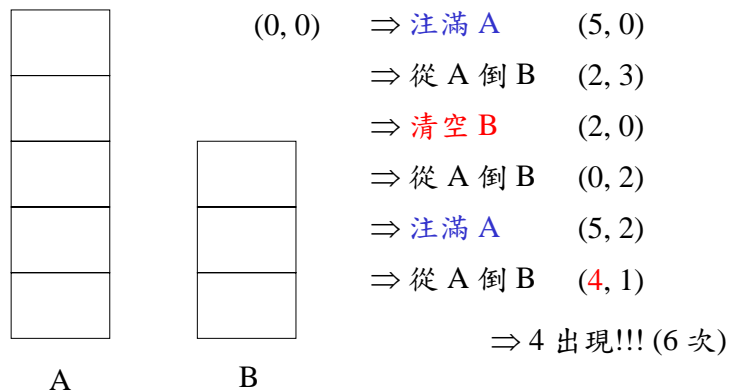
17

應用三: A.571 Jugs

- 給兩個罐子及其容量 A 與 B (A, B 互質), 允許的動作有
 - (1) 把某罐子 **清空**
 - (2) 把某罐子 **注滿**水
 - (3) 把某個罐子的水 **倒到** 另一個罐子裡直到一邊空了或滿了為止
- 給予一目標 n , 請找出最少的動作次數使 n 出現
- $0 < B < A \leq 1000, n \leq A$

18

Example: $A = 5, B = 3, n = 4$



19

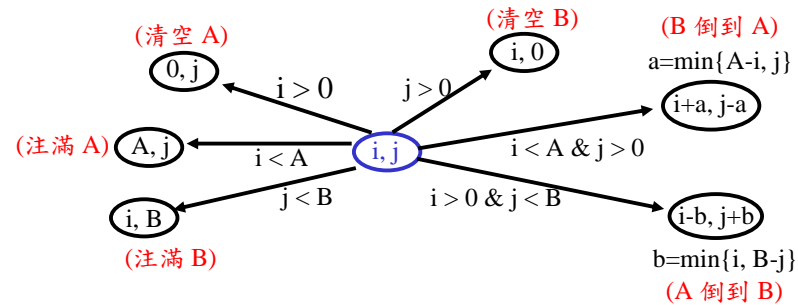
Solution 1

- 巧妙的利用下列性質 (**只適用兩個罐子**)
 - 存在一組最佳解, 不是 A 一直注滿 B 一直倒空, 就是 B 一直注滿 A 一直倒空
- Solution: (假設是 A 一直注滿 B 一直倒空)
 - 把 A 注滿
 - 從 A 倒 B
 - 若 B 滿, 清空 B
 - 若 A 沒了, 注滿 A
 - 檢查是否出現 N , 若無則重複以上步驟
- 把 A 與 B 的角色對調並重複上述步驟, 取動作次數較少的即為答案

20

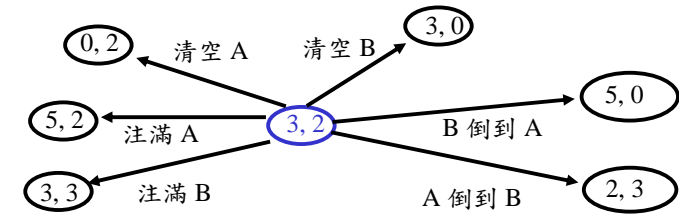
Solution 2

- 以 node $s(i, j)$ 表示 A, B 中的水分別為 i 與 j 的 state
- 決定 $s(i, j)$ 可以一步到達的 neighbors



21

Example: $A = 5, B = 3, n = 4$

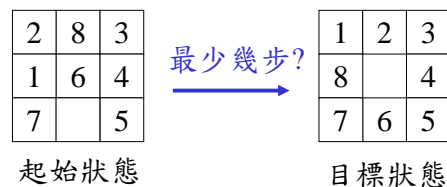


- 用一個 2-d array 表示 $s(i, j)$, $0 \leq i, j \leq 1000$
- 求 $s(0, 0)$ 與所有 $s(n, j)$ 與 $s(i, n)$ 的最短距離
- $O(A \times B) \approx 10^6$

22

應用四: A.652 Eight

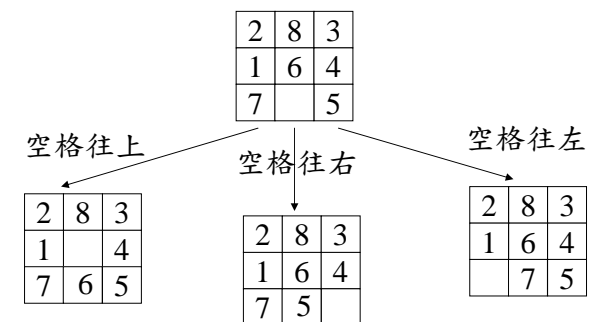
- 一個 3×3 的魔術盤，上面有 1 到 8 的數字與一個空格，空格可以和上下左右的數字交換。給定起始狀態與目標狀態，請問最少要移動幾次才能從起始狀態變成目標狀態



23

Solution

- 僅有 $9! (\approx 10^5)$ 種 states，可用 BFS



- 困難: 如何用 array 來儲存這 $9!$ 個 nodes? (state 和 id 如何 mapping?)

24

狀態編碼

- 將每一種不同的狀態看成一個數字
 - 將空格視為 0

2	8	3
1	6	4
7		5

→ 283164705

- 問題: 不能直接用這個數字作為 id!
 - 雖然只有 $9!$ 種狀態，但是數字範圍從 012345678 到 876543210 ($\approx 10^9$)，無法開一個大小為 876543210 的陣列

25

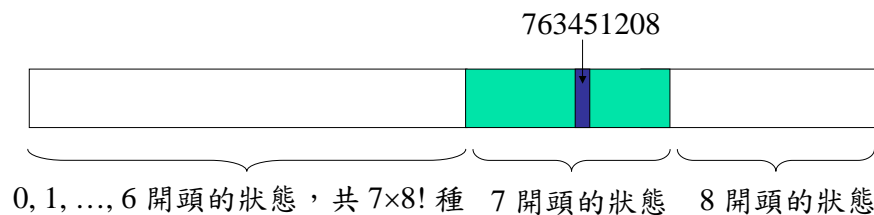
Method 1: Construct a mapping

- 用 state 的字典排列(大小)順序作為 id
- 先暴力展開把所有 $9!$ 種 states 按字典順序存在一陣列中
 - string 陣列大小: $9!$
 - 012345678 → array[0]
 - 012345687 → array[1]
 - 012345768 → array[2]
 - ...
- 要知道某一 state 的 id 時，就在這個陣列中作 binary search

26

Method 2: Computing

- 一樣是用 state 的字典排列順序作為 id
- 用 recursive program 來計算某個 state 是排列中的第幾個

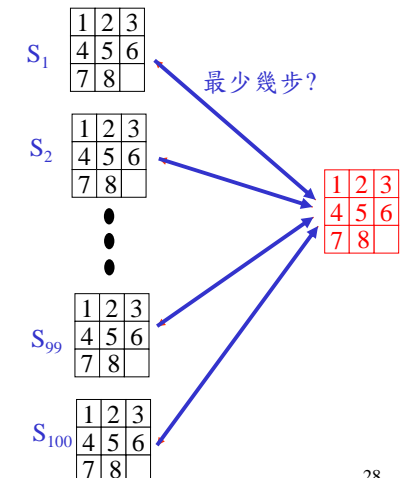


27

Remark: A.652 Eight

- 原題是給 n 組不同起始狀態
- 目標狀態相同，都是

1	2	3
4	5	6
7	8	

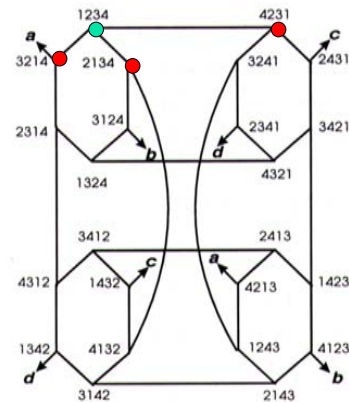


- 只需由目標狀態，作一次 BFS
- 也可以偷跑!

28

應用五: 大甲.2000.3 minimum distance in star graphs

- 給一個 $n (\leq 10)$ 維的 star graph, 以及 $m = 5$ 組圖上的兩點, 求兩點間的最短距離
- star graph
node (x_1, x_2, \dots, x_n) 的鄰居有
 (x_2, x_1, \dots, x_n) ,
 $(x_3, x_2, x_1, \dots, x_n)$,
 \dots ,
 (x_n, x_2, \dots, x_1)
- 大甲題庫:
 - <http://www2.nsysu.edu.tw/contest2004/>



29

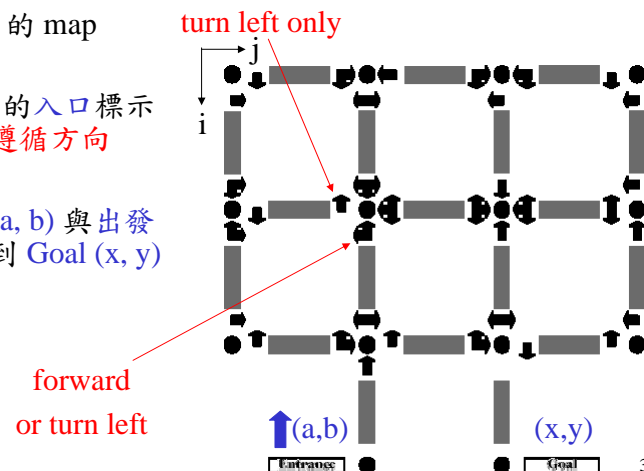
Solution

- $n \leq 10$
 - at most $n! = 10! \cong 10^6$ nodes
 - at most $n! \times (n-1) \cong 10^7$ edges
- BFS
 - $O(m \times (n! \times (n-1))) \cong 10^8$
 - need a mapping for the ID
(from 10^{10} to 10^6)

30

應用六: A.816 Abbott's Revenge

- 給一個 $n \times n$ 的 map
- 每一個 node 的入口標示有離開時的遵循方向
- 給 Entrance (a, b) 與出發方向, 找走到 Goal (x, y) 的最短距離
- $n \leq 9$



31

Difficulty

- 由不同的方向到達 node (i, j) 有不同的意義
- 當一個 node 被 visited, 不可以被 mark 成 GRAY
- 當一個 GRAY node 展開後, 也不可以被 mark 成 BLACK

32

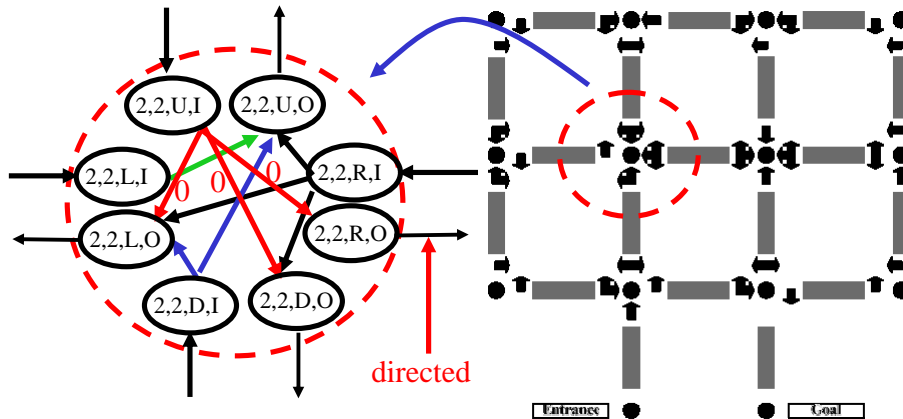
Solution

node duplication (還記得這個技巧嗎?)

disadvantages:

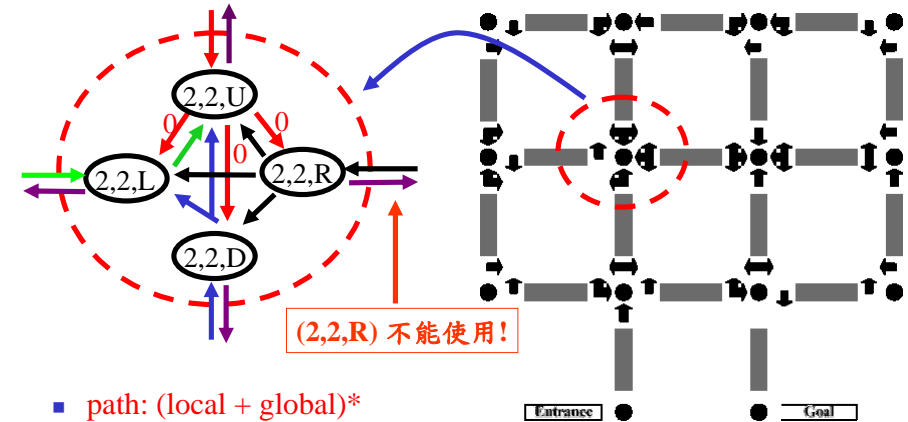
- (1) construct G'
- (2) handle 0 edges

~~BFS~~



33

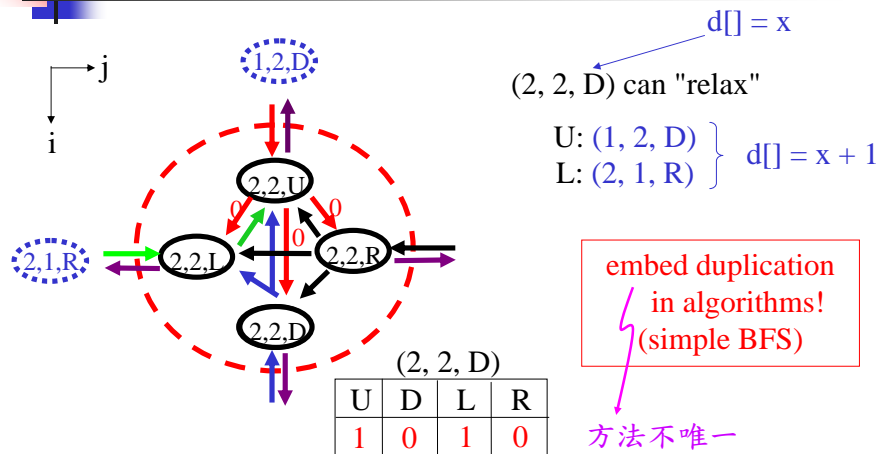
Another way to duplicate



path: (local + global)*

34

Implementation



35

Another way to duplicate (cont.)


- node duplication
 - $s(i, j, k)$: 目前由入口 k 進入 node (i, j) , $k \in \{U, D, L, R\}$
 - 一共有 $9 \times 9 \times 4$ nodes
- two-level relaxation: (local + global)*
 - 不必真的造圖, edges 在程式中用"算的"或"查表"
- 作 BFS
 - 依出發點 (a, b) 與出發方向, 找到一步可以到達的入口 $s(c, d, e)$
 - 由 $s(c, d, e)$ 開始作 BFS, 求 $s(c, d, e)$ 與所有 $s(x, y, k)$ 的最短距離

36



作業與自我挑戰

- 作業
 - 練習題
 - A.816 Abbott's Revenge
<http://uva.onlinejudge.org/external/8/816.html>
 - 挑戰題
 - A.10603 FILL
<http://uva.onlinejudge.org/external/106/10603.html>
 - A.810 A Dicey Problem
<http://uva.onlinejudge.org/external/8/810.html>
- 自我挑戰
 - A.589 Pushing Boxes
<http://uva.onlinejudge.org/external/5/589.html>
 - AT2000F A Version of Nim

- 
- 其它有趣的題目
 - A.439 Knights Moves
 - <http://uva.onlinejudge.org/external/4/439.html>
 - H. 90.1 影像之結構化特徵
 - http://www.cc.nccu.edu.tw/info_race90/finalprogram.pdf
 - A.298 Race Tracks
 - <http://uva.onlinejudge.org/external/2/298.html>
 - A.336 A Node Too Far
 - <http://uva.onlinejudge.org/external/3/336.html>
 - A.10097 The Color Game
 - <http://uva.onlinejudge.org/external/100/10097.html>