電腦數值模擬導論

實習九

B03702030 會計三 吳懿峰

一、 解釋模擬的策略

1. 先把格式設定好。設定出一個 50X50 的正方形區域，再將此正方形區域填入灰色，外邊為土褐色，上邊為藍色(代表水)。
2. 隨機拔取 50x50 區域(灰色)中的格子，判斷只要是拔到非灰色區域，就直接跳出重拔，以避免重複拔取格子。
3. 每拔取一個格子，就判斷周圍是否有水，一旦有水，則進行步驟四，若周圍沒水(藍色)，則回到步驟二繼續拔格子。
4. 一旦判斷有水，就將儲存格填入藍色。
5. 判斷上下左右的空白儲存格，並依據各種不同情況(如:僅左邊為空白格、左邊及上面為空白格、右邊左邊下面為空白格等…)分為不同的路徑模式執行。
6. (A)當周圍僅一個空白格，則將此空白格填入藍色，並將此位置儲存至 a，再判斷 a 位置上下左右的空白儲存格。

   (B)當周圍僅兩個空白格，則將其中一個空白格填入藍色，並將此位置儲存至 a，而另一空白格則儲存至 tunnel 陣列。再判斷 a 位置上下左右的空白儲存格。

   (C) 當周圍有三個空白格，則將其中一個空白格填入藍色，並將此位置儲存至 a，而另兩個空白格則儲存至 tunnel 陣列。再判斷 a 位置上下左右的空白儲存格。
7. 執行第六步驟直到第一個路徑結束為止(即 a 位置判斷周圍皆無空白儲存格時)。路徑結束後，將原先尚未執行的路徑依 tunnel 陣列所儲存之位置存入 a，並重複執行五、六步驟直到所有路徑皆走完為止。
8. 重複步驟二到步驟七，直到最底層出現藍色儲存格(水)為止。
9. 執行 100 次並輸出被拔格子數、孔隙率、跑程式時間等。

二、 流程圖

1. 大綱式

   由於圖片過大，煩請參閱資料夾附帶的”實習九概略.jpg”
2. 詳細版

   由於圖片過大，煩請參閱資料夾附帶的”實習九詳細.jpg”

三、 列出執行結果

模擬序數	移除格子點數	切斷孔隙率	跑程式時間
1	1489	0.5956	1.812988281
2	1394	0.5576	1.672119141
3	1430	0.572	2.014892578
4	1528	0.6112	2.547119141
5	1456	0.5824	1.984863281
6	1441	0.5764	1.953125
7	1599	0.6396	2.745849609
8	1437	0.5748	2.061035156
9	1607	0.6428	2.625
10	1488	0.5952	2.406982422
11	1411	0.5644	1.937011719
12	1457	0.5828	2.187988281
13	1361	0.5444	1.838134766
14	1472	0.5888	2.312011719
15	1568	0.6272	2.519042969
16	1571	0.6284	2.608886719
17	1338	0.5352	1.812988281
18	1469	0.5876	2.218994141
19	1394	0.5576	1.843994141
20	1470	0.588	2.218017578
21	1582	0.6328	2.642089844
22	1391	0.5564	2.062988281
23	1541	0.6164	2.546875
24	1480	0.592	2.187011719
25	1485	0.594	2.391113281
26	1527	0.6108	2.452880859
27	1486	0.5944	2.326171875
28	1515	0.606	2.140869141
29	1537	0.6148	2.328125
30	1468	0.5872	2.218994141
31	1457	0.5828	2.125
32	1452	0.5808	2.202880859
33	1457	0.5828	2.078125
34	1493	0.5972	2.127929688
35	1459	0.5836	2.170898438
36	1540	0.616	2.437988281
37	1361	0.5444	2.109130859
38	1398	0.5592	2.015869141
39	1487	0.5948	2.468017578
40	1503	0.6012	2.454101563
41	1413	0.5652	2.062011719
42	1468	0.5872	2.355957031
43	1473	0.5892	2.264892578
44	1555	0.622	2.687988281
45	1621	0.6484	2.812011719
46	1386	0.5544	1.891113281
47	1433	0.5732	2.140869141
48	1371	0.5484	1.893066406
49	1491	0.5964	2.359130859
50	1599	0.6396	2.656005859
51	1604	0.6416	2.664794922
52	1429	0.5716	1.922119141
53	1554	0.6216	2.312011719
54	1474	0.5896	2.234863281
55	1397	0.5588	1.828125
56	1486	0.5944	2.295898438
57	1539	0.6156	2.781982422
58	1616	0.6464	2.498046875
59	1507	0.6028	2.390136719
60	1443	0.5772	2.187988281
61	1550	0.62	2.548828125
62	1492	0.5968	2.156005859
63	1456	0.5824	2.312011719
64	1522	0.6088	2.156982422
65	1488	0.5952	2
66	1488	0.5952	2.359130859
67	1462	0.5848	2.25
68	1449	0.5796	2.077880859
69	1453	0.5812	2.25
70	1471	0.5884	1.962158203
71	1448	0.5792	2.198974609
72	1475	0.59	2.015869141
73	1271	0.5084	1.718017578
74	1425	0.57	2.204101563
75	1467	0.5868	2.170898438
76	1498	0.5992	2.281982422
77	1446	0.5784	2.328125
78	1486	0.5944	2.202880859
79	1488	0.5952	2.156005859
80	1527	0.6108	2.516113281
81	1429	0.5716	2.108886719
82	1508	0.6032	2.531982422
83	1521	0.6084	2.390136719
84	1558	0.6232	2.562988281
85	1372	0.5488	1.843994141
86	1417	0.5668	2.093017578
87	1462	0.5848	2.25
88	1522	0.6088	2.501953125
89	1525	0.61	2.579101563
90	1516	0.6064	2.389892578
91	1325	0.53	1.875
92	1429	0.5716	2.031982422
93	1488	0.5952	2.171142578
94	1496	0.5984	2.375976563
95	1463	0.5852	2.202880859
96	1513	0.6052	2.390136719
97	1428	0.5712	2.171875
98	1508	0.6032	2.391113281
99	1501	0.6004	2.293945313
100	1392	0.5568	2.009033203

四、　　列出平均

移除格子點數	切斷孔隙率	跑程式時間
1475.93	0.590372	2.24078125

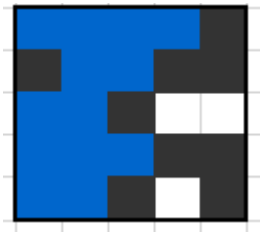
五、　　是否會三維的模擬

三維的模擬就無法用顏色來判斷了，取代而之應該用數字代替，如石頭區

域改為代號 0，空白格為 1，水則是 2，接著在位置上就必須變成 50*50*50 的區域，可以用 1 到 12500 代表所有方格的位置。但同樣地，如果能設定方格為 52*52*51 (51 為高，最上面一層是水)，在三維的立體方格外圍包上一層非空白格區域(即假設為非 1 之任何數)，會使我們更加方便計算。在流水的方向判斷上，除了上下左右還須加上「頂」、「底」兩個方位，這將會使得判斷路徑變得十分複雜(會產生 30 種不同的路徑，少了 2 種分別是全部方位皆為空白格以及全部方位皆非空白格的情況)，然而邏輯方法仍與二維的相同，最後判斷水(2)流到最底層即結束程式。

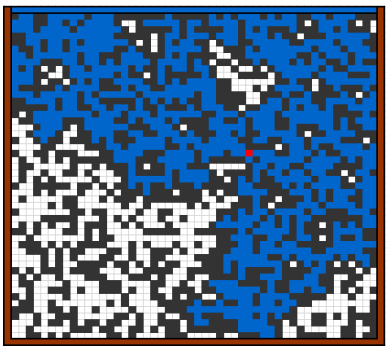
六、 心得

這題是迄今為止花了最久的時間完成的作業吧…，首先一開始做 5x5 的模擬時就快崩潰了，因為那時候傻傻的用了一維的陣列又用顏色判斷，卻沒有加上邊框，導致必須分別做最左邊、中間、最右邊，以免最旁邊的格子會影響到下一行的格子。而填水時又是先一行一行判斷，最終使得程式碼又臭又長，還要做好幾次的 Do-Loop 迴圈才能將水填滿。當時覺得這樣下去如果用同樣的邏輯做 50x50 的話會爆炸甚至可能做不出來，因此就換了另一個方式。



圖為 5x5 的執行結果…

經過老師跟助教的指導才有了更清晰的想法，雖然本來就有想過如果能像走迷宮一樣，將每一條的路走完並記下尚未走過的路，等到下次再將沒走過的路走完，以這樣的思維應該可以更有效率以及更快完成流水的程式，但我卻遲遲無法動工(因為不知道怎麼寫)，直到懂了如何運用陣列儲存位置後，才開始進行這最麻煩的部分。在設計程式的過程中，最困難的就是陣列的讀取了，大概讓我除了兩三次的蟲，最後甚至是把情況畫在書桌上，一步一步推敲，最終才完成了這個煩人的作業，總計花了約 12 個小時的時間才完成。但不得不說看到程式一步一步把水填滿，且僅在 2 秒的時間就完成真的是有很大的成就感。剩下最後一個作業就解脫了!



圖為 50x50 的執行結果