R07922118 資工碩一 翁子騰

Problem 5

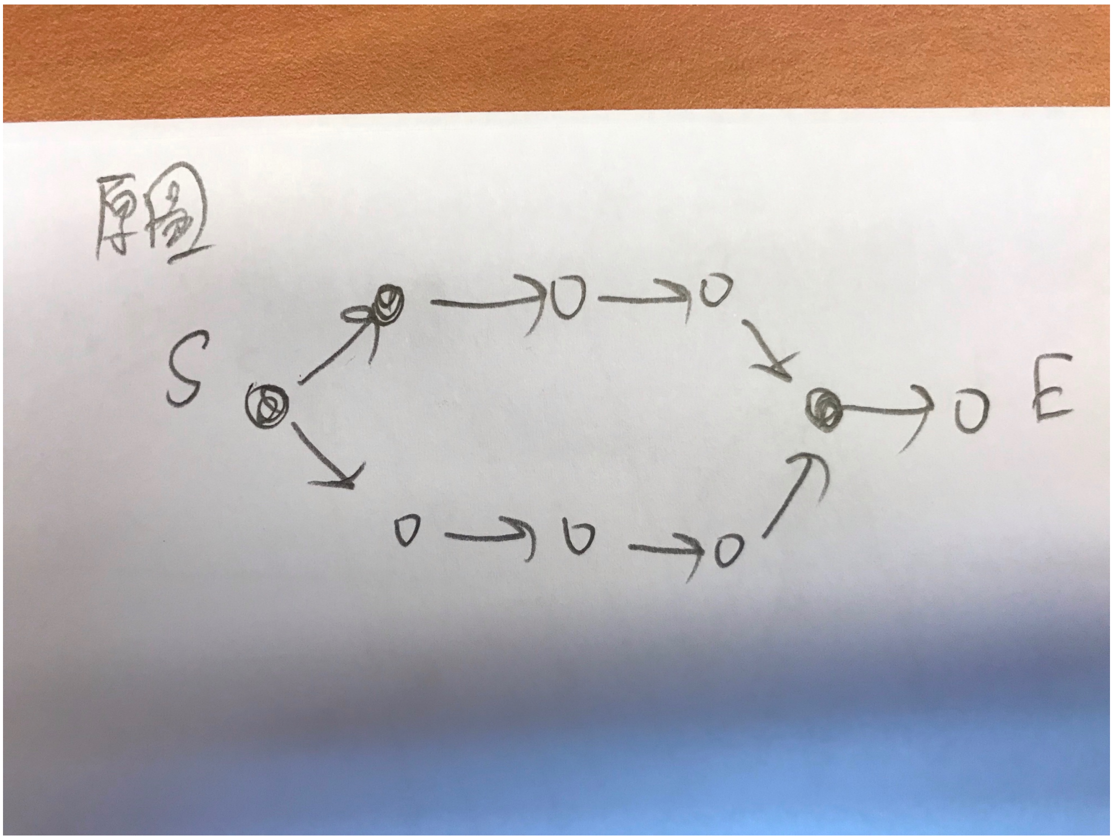
5-(a)

1 設原本有生命值 HP=3(一開始就吃到 pie，只能再多走 2 個城市)

2 對原圖複製兩份 verteces，成為一個有 3*|V|的大圖，大圖可以分成 3 個區塊，HP=3, HP=2, HP=1。

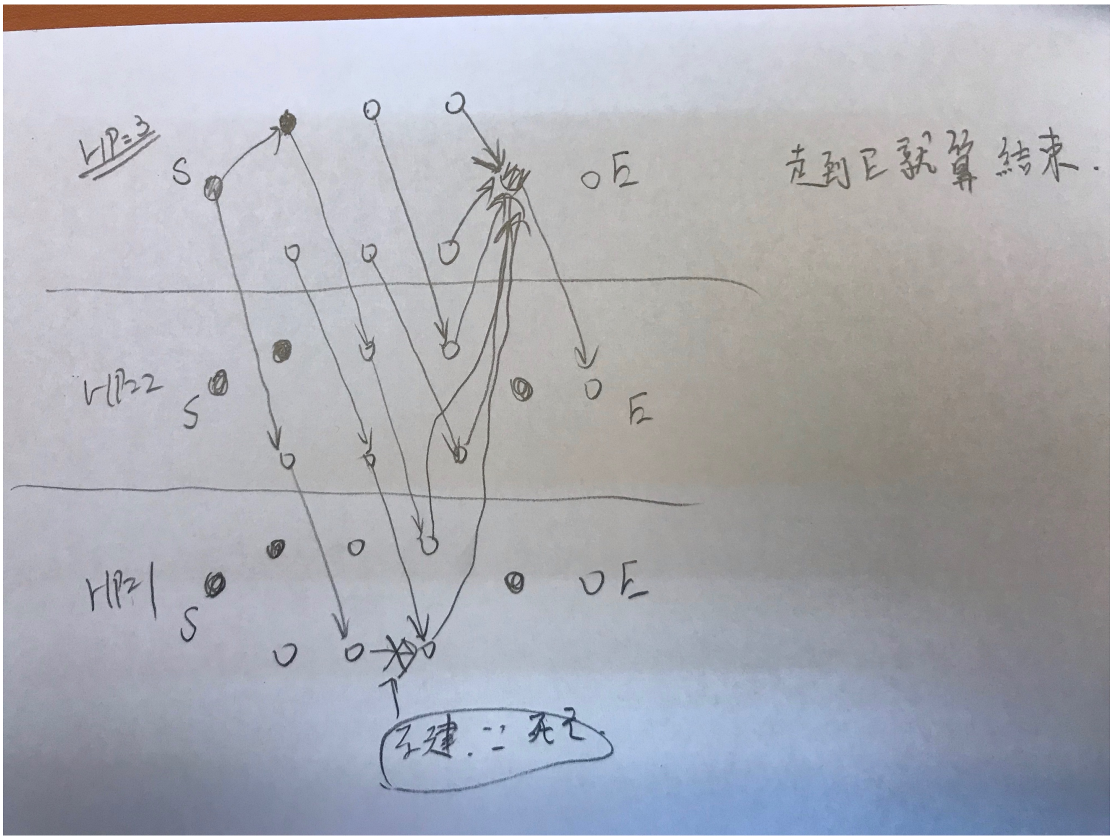
3 開始建 edge，建 edge 的方法是走訪所有的 vertices。如果吃到 pie，是從 HP=3 的那部分開始走。如果原圖的下一步是有 pie 的城市，就建一條邊回到 HP=3 的部分，如果原圖下一步是沒派的城市，就建一條邊到 HP 少 1 的那一部分圖

4 針對大圖的 empire city 做 dijkstra



大圖中會有 3*|V|個 vertex 和 O(3*|E|)條邊。

第3步花3*O(V+E)，第4步花O(Dijkstra(V’=3*V,E’= 3*E))，一共花 3*O(V+E)+O(3*Dijkstra(V,E))=O(Dijkstra(V,E))。

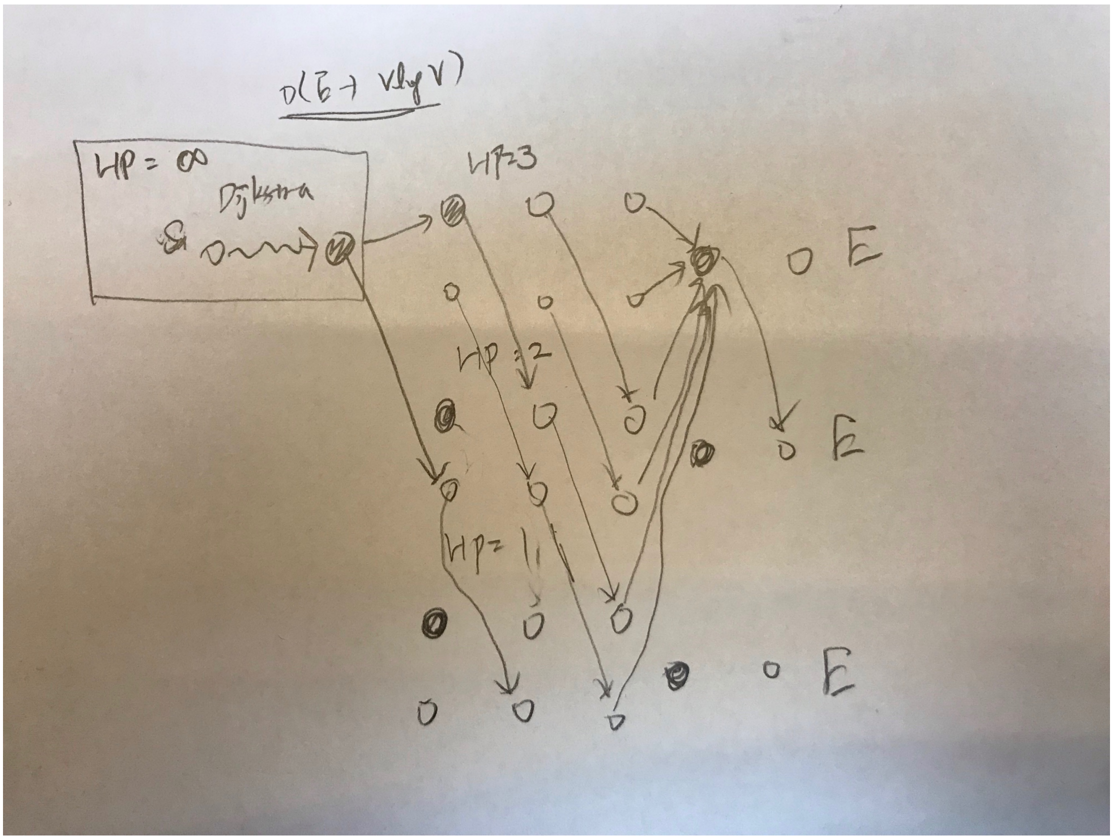


5-(b)

假設一開始沒吃到 pie，一開始就對原圖做 dijkstra，若一路上都沒遇到有 pie 的城市，直接 return。(直接得到正解)

若走到有 pie 的城市，重複 5-(a)，複雜度仍為 O(Dijkstra(V,E))

R07922118 資工碩一 翁子騰



5-(c)

1 設原本有生命值 HP=k+1

2 針對原圖的 empire city 做 dijkstra，若從頭到尾都沒吃到 pie 就直接 return 正解。

3 若一開始還沒吃到 pie，但中間吃到 pie，就走到吃到 pie 為止，對吃到 pie 之後的原圖複製 k 份 verteces，成為一個有 O(k*|V|)的大圖，大圖可以分成 k 個區塊，HP=k+1, HP=k,…, HP=1。

4 開始建 edge，建 edge 的方法是走訪所有的 vertices。如果吃到 pie，是從 HP=k+1 的那部分開始走。如果原圖的下一步是有 pie 的城市，就建一條邊回到 HP=k+1 的部分，如果原圖下一步是沒派的城市，就建一條邊到 HP 少 1 的那一部分圖

5 對大圖的 HP=k+1 部分的原點做 dijkstra，得到解答。

大圖中會有(k+1)*|V|個 vertex，O((k+1)*|E|)條邊。

第3,4步建圖花(k+1)*O(V+E)，第5步花O(dijksra(V'=(k+1)*V, E'=O(k+1)*E)

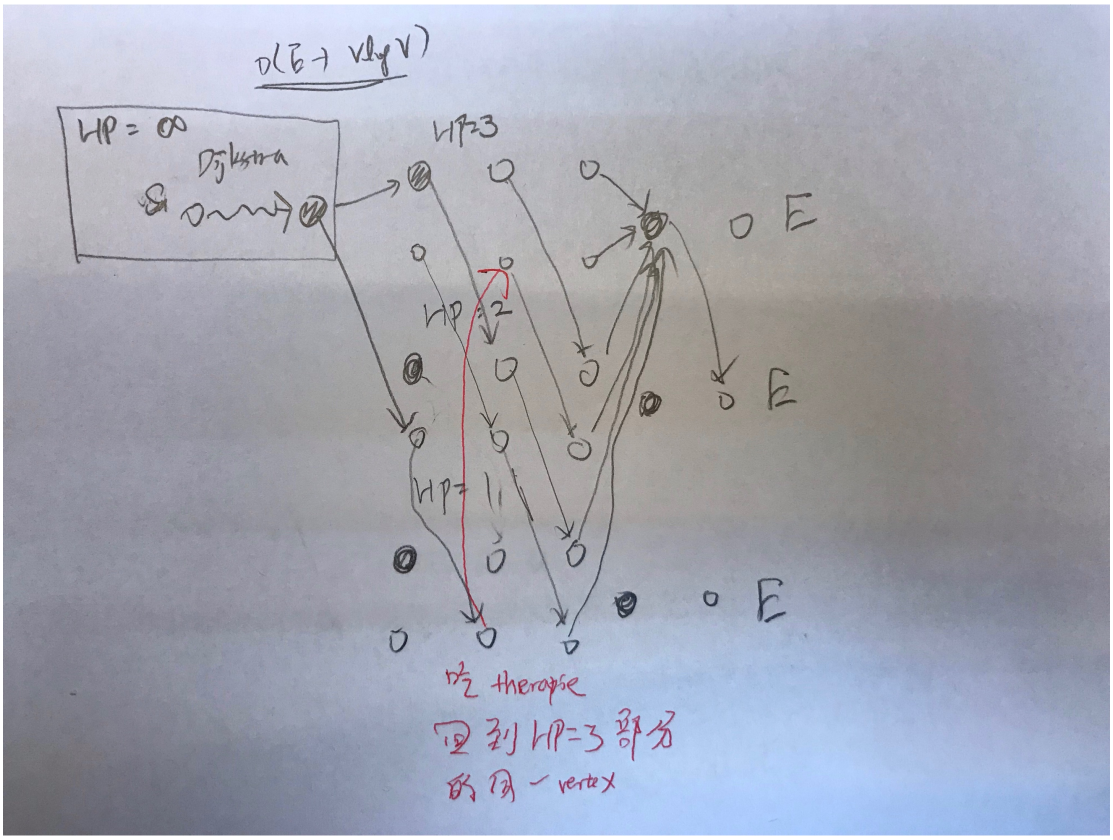
=O(k*dijkstra(V,E))，總共花O(k*dijkstra(V,E))。

5-(d)

設 theraPIE=True

前面作法和 5-(c)一樣，但是在 HP=1 部分且沒有 *edge 可走*的情形，吃 theraPIE(theraPIE=False)使得 HP 補滿，回到 HP=k+1 部分的同一個 vertex 繼續走。若可以走到終點則發現 shortest-path，但若還是走不到終點，只好回到吃 theraPIE 的當時的前一個 vertex，同時 theraPIE=True，重找一條路做 dijkstra，不斷重複到找到終點為止。

以上只比 5-(c)增加了一個變數 theraPIE，吃了 theraPIE 可能讓本來不能走到的路變得走得到，也可能讓本來沒辦法走到的路花更多的時間才發現不能走。但是整體而言，對於演算法的時間複雜度影響為 O(1)，仍為 O(k*Dijkstra(V,E))。



5-(e)

5-(f)

選擇的策略是：一遇到有 pie 的城市就淨化。

總共可以淨化 k2 個城市。

除了複製 k 份原本的 vertex 外，在另外複製 k2 份原本的 vertex，做一張超級大的 graph。多複製 k2 的那部分，代表所剩下的淨化額度。

也就是現在除了 HP=k+1 之外還有一個是 MP=k2。所以大圖上會有 HP=k, k-

1, …,1 的部分還有 MP=k2, k2-1, …,1 的部分。

一開始從 MP=k2 的部分開始做 dijkstra，走到有 pie 的城市發動淨化，就會到 MP=k2-1 的部分，直到 MP=0 就會進入 HP=k+1 的部分。之後就會像 5-(c)。

現在的 graph V'=(k+1)*V+ k2*V E'= O((k+1)*E+k2*E)，所以總共需要 O(Dijkstra(V', E'))=O((k+k2)*Dijkstra(V,E))

5-(g)

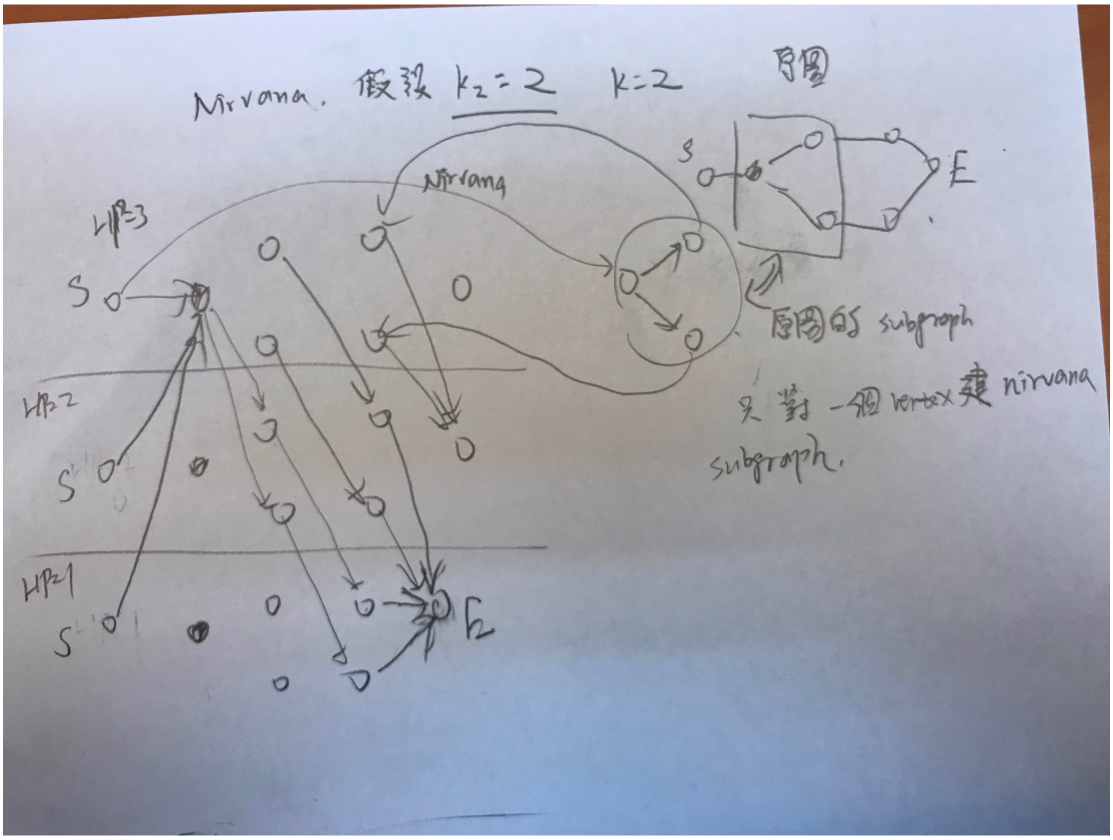
每一個時間點都有可能可以發動 Nirvana，所以要針對每一個 vertex，另外再多 k2 個 vertex。

同樣使用 5-(c)的建圖方法，只是現在針對大圖上面每一個 vertex，多畫 k2 個 vertex，這 k2 個 vertex 是原圖的 subgraph。建立一個更大的 graph。

如此會有(k+1)*|V|*(k2)的 vertex，邊會有 O((k+1)|*|E|*k2)條。

對超大圖的原點做 dijkstra，所花的時間複雜度為 O(dijksra(V'=(k+1)*V*k2, E'=O((k+1)*E*k2)))=O((k*k2*dijkstra(V, E)))。

以下的圖**只有針對一個 vertex 做 Nirvana**，沒有對每一個 vertex 做 Nirvana。



R07922118 資工碩一 翁子騰