Algorithm - Strongly Connected Components

Def (Strongly Connected Component)

$G = (V, E)$ 是個有向圖，$C \subseteq V$ 是個 G 的 connected component。若:

$$\forall u, v \in G.\ u \rightsquigarrow v \text{ and } v \rightsquigarrow u$$

且:

$$\forall w \in V \setminus C. V \cup \{w\} \text{ is not a connected component}$$

則稱 C 是個 Strongly Connected Component.

更精簡的訂法:

$$\text{SCC is the equivalent class of ”mutually reachable”}$$

Observation (轉置後 SCC 不變)

$G = (V, E)$ 是一張有向圖，則:

$$U \text{ is a SCC of } G \iff U \text{ is a SCC of } G^T$$

假定 G 中 $u \overset{p_1}{\rightsquigarrow} v$ 且 $v \overset{p_2}{\rightsquigarrow} u$。則顯然 G^{T} 中 $v \overset{p_1^{\mathrm{T}}}{\rightsquigarrow} u$ 且 $u \overset{p_2^{\mathrm{T}}}{\rightsquigarrow} v$，因此在 G 中連通的各點，在 G^{T} 中仍然連通。

因為 $G = (G^{\mathrm{T}})^{\mathrm{T}}$，所以 G^{T} 中連通的各點，在 G 中也保持連通。

所以知道:

$$u, v \text{ mutually reachable in } G \iff u, v \text{ mutually reachable in } G^{\mathrm{T}}$$

由此得證。

Lemma (SCC 們是個 DAG)

$G = (V, E)$ 是一張有向圖，C', C 是 G 相異的 SCC，$u, v \in C$，$u', v' \in C'$，則:

$$u \rightsquigarrow u' \Rightarrow v' \not\rightsquigarrow v$$

若 $v \rightsquigarrow v'$，則對於 C 中的任意點 w 及 C' 中任一點 w' ：

$$\left.\begin{aligned} & w \rightsquigarrow u \\ & u \rightsquigarrow u' \\ & u' \rightsquigarrow w' \end{aligned}\right. \Rightarrow w \rightsquigarrow w'$$

及:

$$\left.\begin{aligned} & w' \rightsquigarrow v' \\ & v' \rightsquigarrow v \\ & v \rightsquigarrow w \end{aligned}\right. \Rightarrow w' \rightsquigarrow w$$

故 C'，C 都不是 Strongly Connected Component，矛盾。

Def (Discovery and Finish Time for Sets of Vertices)

$G = (V, E)$ 是一張有向圖，$U \subseteq V$，則定義 DFS 的起始與結束時間:

$$\begin{cases} d(U) = \min\left(\{u.d \mid u \in U\}\right) \\ \\ d(U) = \max\left(\{u.d \mid u \in U\}\right) \end{cases}$$

Lemma (邊的指向就是遍歷順序)

$G = (V, E)$ 是一張有向圖，C', C 是 G 相異的 SCC。假定 $v \in C$，$v' \in C'$，且 $(v, v') \in E$，則:

$$f(C) > f(C')$$

假定 $d(C) < d(C')$，令 $x \in C$ 是 C 中第一個被發現的點。在 $x.d$ 時間時，C, C' 全白。對於任意 $w' \in C'$：

$$x \rightsquigarrow v \rightarrow v' \rightsquigarrow w$$

是一條全白路徑。因此 C 中所有點都是 x 的子節點。由 Nestings 得證。

假定 $d(C) > d(C')$，假定 x' 是 C' 中第一個發現的點。因 $d(C) > d(C')$，故在 $x'.d$ 時，C 為全白。由 Lemma 知：

$$\neg \exists u \in C, u' \in C'. (u', u) \in E$$

所以在 $x'.d$ 時:

$$\forall w \in C. x' \cancel{\xrightarrow{\texttt{WHITE}}} w$$

因此 C 中任意點，都不是 x' 的子節點。由 Nestings 的狀況 1. 知 C 中每一點的 f 值都比 C' 中每一點的 f 值大。由此得證。

如果把 SCC 收縮之後的圖想成 DAG，再用 DAG 的性質下去做感覺也可以?

Corollary

$G = (V, E)$ 是一張有向圖，C', C 是 G 相異的 SCC。假定 $v \in C$，$v' \in C'$，且 $(v', v) \in E^T$，則:

$$f(C) > f(C')$$

$$(v', v) \in E^T \iff (v, v') \in E$$

因為 G 和 G^T 的 SCC 相同，因此套用 Lemma 即得證。

Kosaraju's Algorithm

假定 $G = (V, E)$ 是一張有向圖，則以下的演算法可以找出 SCC：

1. `DFS(G)`，並在過程中紀錄每一點的 f 值。
2. 計算 G^{T}
3. 對所有 G^{T} 中的點，依照第一次 DFS 的 f 由大到小的順序，進行 `DFS_VISIT`。至於要怎麼做到，可以第一次 DFS 時每塗黑一個點就塞進 `stack` 裡，第再依序 `pop` 出來就可以了。
4. 結束之後，DFS Forest 中的每棵樹，就是一個 SCC。

直觀的理解是：

1. 第一次 `DFS` 時，出發點所屬的那個 SCC （C_1）可以到達剩下所有的 SCC。
2. 所有邊反過來後，C_1 就變成「沒辦法到任何一個 SCC 的 SCC」。因此第二次 `DFS` 從他開始時，因為到不了任何地方，DFS 時自己變成一棵樹。
3. C_1 遍歷完之後，因為到不了其他 SCC，所以就會準備生下一棵 DFS Tree。假定下一個棵樹的起點屬於 C_2 。 C_2 的點在 G^{T} 中，只能走到 C_1 跟 C_2 的點，但 C_2 的點都走過了，所以狀況就變得跟 C_1 一樣。
4. 以此遞推下去。C_k 中的點，除了自己內部的點以外，都只能走到 $C_1, C_2 \ldots C_{k-1}$ 中的點，但這些 SCC 必定都會在遍歷 C_k 時全部被遍歷過。所以第 k 棵 DFS Tree 就只能包含 C_k 中的所有點。
5. 因此可以用歸納法證出第k 個 SCC ，剛好是第二次 DFS 的第 k 棵 DFS Tree，

所以證法就是歸納法：

假定第一次遍歷時， $f(C_1) > f(C_2) > \ldots > f(C_k) > \ldots > f(C_n)$。

$(k=0)$：顯然成立。

$(k>1)$：假定 $\forall C_i.\, f(C_i) > f(C_k)$ 都已在遍歷 G^{T} 時被遍歷過，且他們的 DFS Tree 點 = SCC 點。則：

1. 若 C_k 在 G^{T} 中有通往其他 SCC C' 的邊，由 Corollary 知：

   $$f(C_k) < f(C')$$

   但由歸納法將假設知：所有滿足 $f(C_k) < f(C')$ 的 SCC C' 都已被遍歷過，因此第 k 個 DFS Tree 不可能經過這些點。

2. 又因為 C_k 內部各點連通，假定第 k 棵 DFS Tree 的根節點是 u。根據 White-Path Theorem，C_k 中的所有點，u 的子節點。但上面的推論知這棵 DFS Tree 不可能包含 C_k 以外的點了，由此得證這棵 DFS Tree 的點，恰好就是 C_k 所有的點。