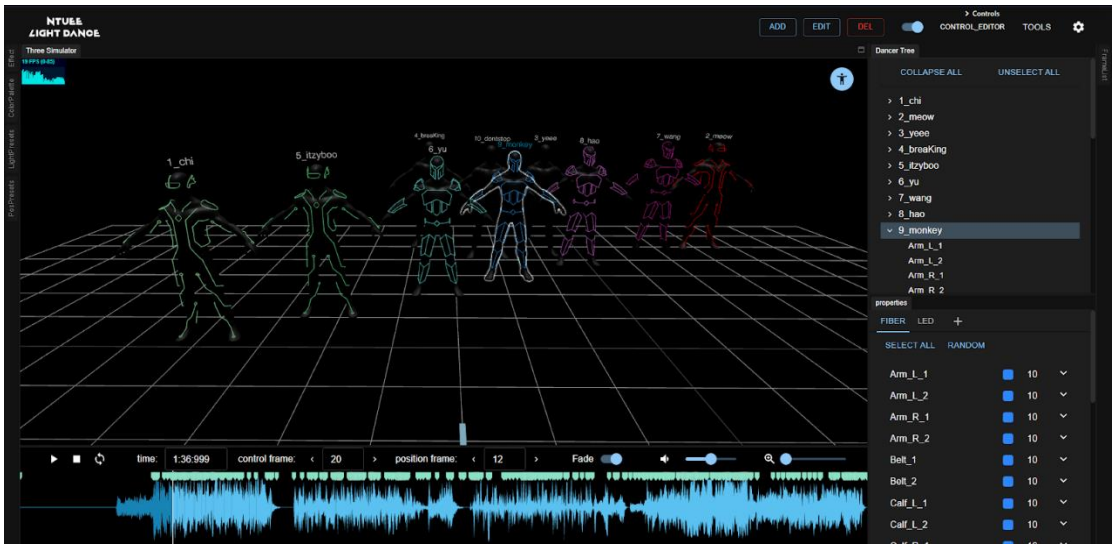
Alg22_DIY 光舞

B08901073 王維芯

「你有看到今年的光舞嗎？超帥的欸！」
「沒有欸，我忙著寫演算法 DIY…」
「蛤寫什麼 DIY 啦？還不趕快上 YouTube 搜尋 2022 台大電機光舞！」

光舞表演當中絢麗的特效是由幕後團隊用程式設計出來的，這個過程我們稱為「編光」。



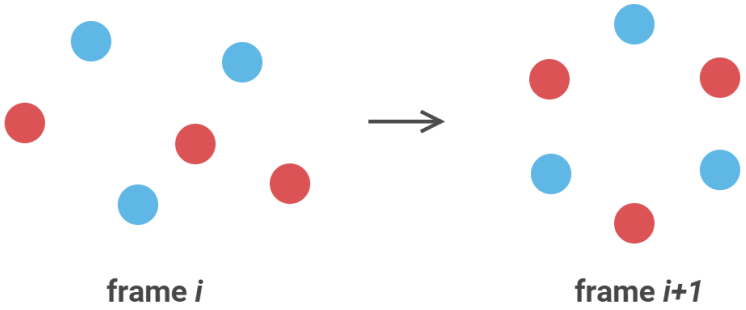
軟體組開發的強大編輯器 ── Lightdance Editor

在編輯器當中，我們可以在選定的時間點插入 keyframe，在該影格調整隊形和光的特效，例如每個部位的顏色和亮度。編光牽涉到許多複雜的走位以及各種特效的排序問題，這些可以從演算法的角度來解決。

一、隊型問題

問題：

按照劇情，下一首歌要進行打鬥的場面，舞者要從現在的站位移動到下一個隊型。下一個隊形中，舞者要排成圓圈，兩兩相鄰的人會打架（影片 5:37）。已知劇情設定是兩個敵對國家的人互相挑釁，A 國以紅色表示，B 國以藍色表示，因此隊形可以視為一個顏色交錯排列的圓圈。另外，考量到舞台上走位需要花的時間，舞者可以移動的距離有最小值。我想找出下一個隊形可能的排列方式，並且盡量用到最多的舞者。



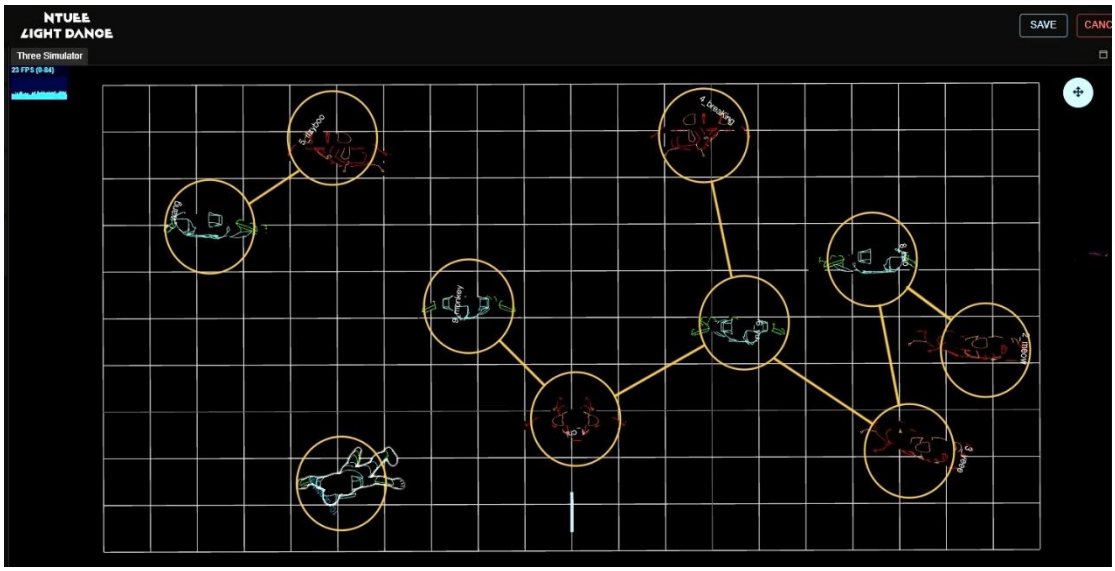
問題示意圖

解法：

已知下一個隊形中可以站在一起的條件：

1. 方圓距離 x 以內的人
2. 不同國家的人

因為目前的站位可以看成是一個拓樸結構，所以直覺地想用 graph 的方式表示，而相連的部分是指接下來可以站在一起的人。如此一來可以形成一個 undirected graph。

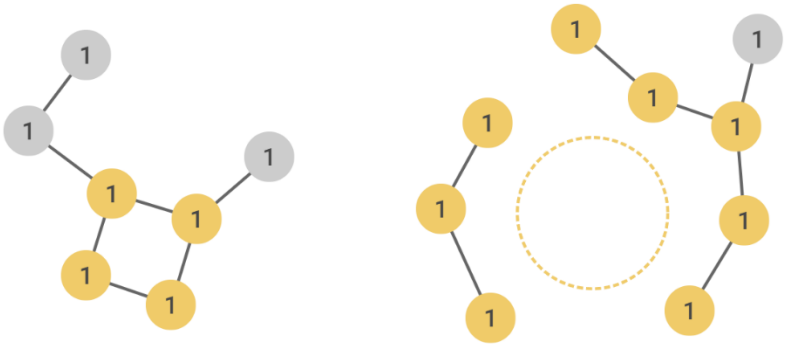


t_0 隊形示意圖

接著，可以根據這個 graph 產生一個 adjacency list，每個 node 代表一個舞者，每個 list 代表接下來可以選擇的鄰居。

要排成下個隊形中的圓圈需要考慮兩個 case：

1. cycle：最大的 cycle 本身就可以排成下個隊形。
2. multiple chains：很多個 chains 連接在一起也能形成一個圓圈。這個情況要額外考慮相鄰端點的人是否屬於不同國家。



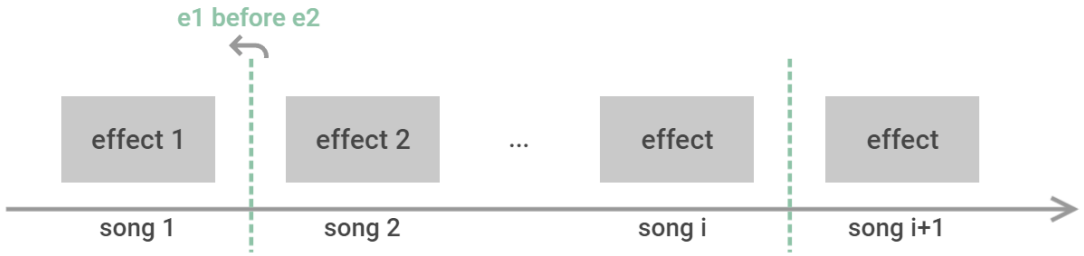
case 1: cycle　　case 2: multiple chains

再比較這兩個 cases，找出最大值。
操作上，可以用 DFS 找出 undirected graph 中的 connected components。記錄每個 connected components 的最長路徑以及它的性質是否為 cyclic。若為 cyclic 則屬於 case 1，若為 acyclic，則歸類為 case 2。再將各個 chain 拼起來，與最大的 cycle 做比較取最大值。演算法的時間複雜度是 O(V+E)。

二、特效排序問題

問題：

經過多年的經驗我們發現，先想好一些想呈現的效果，例如彩虹漸層、武器碰撞等等，再針對這些特效進行編舞是比較有效率的做法。集思廣益以後，我們設計了一列特效的素材庫。我們知道不同首歌有各自適合的特效，因此有些特效必須出現在其他特效之前。有了特效素材和各自的順序條件，就可以在各個 keyframe 插入對應的特效，我想找出所有可能的排列方式。



問題示意圖

解法：

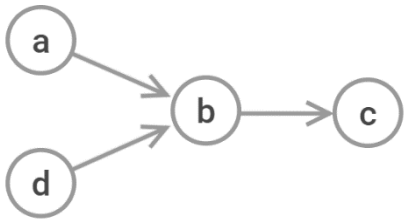
先定義資料樣式如下：

$effects$：特效集，標號由 0 到 n-1

$prerequisites[i] = [a_i, b_i]$：先決條件，表示特效 a_i 必須發生在 b_i 之前。

因為音樂是連續的，所以前者的 $prerequisites$ 也會是後者的 $prerequisites$。假設$prerequisites[i] = [[a_i, b_i], [b_i, c_i]]$，則 a_i 也必須發生在 c_i 之前。

因為有先後順序，我想用有向圖的方式來表示這個問題。



畫成有向圖之後，觀察這個問題應該可以用 topological sort 來解決。

先建立一個 list 叫做 $indeg[i]$ 用來記錄第 i 個 effect(node) 有多少個先決條件，也就是有多少累積的 incoming edges。再建立一個 set S 用來記錄不具有 incoming edges 的 node，即 $indeg = 0$ 的 node。建立一個 $prereq$ list 用來記錄每個 node 的先決條件。

依序跑過 S 中的每個 node，對他的相鄰子節點的 $indeg - 1$。對每個 node 和他的子節點的先決條件取聯集，再取代為子節點的先決條件。若迭代遞減後 $indeg = 0$，就將這個子節點加入 S。

因為每個節點的條件要包含之前的條件，所以演算法的時間複雜度是 O(EV)。