NP-Complete

• 由指定的起點前往指定的終點,途中經過所有其他節點且只經過一次 • 在圖形上,經過所有的點且不 重複點而回到原來的出發點,這樣的問題我們稱 之為“漢米爾頓迴路(Hamilton cycle)"。

• 在圖形上給定任意兩點 A、B,從 A 出發要 經過所有的點且不重複點而到達到 B,這樣的問題稱作“漢米爾頓路徑 (Hamilton path)"

[1] NP問題介紹:

Hamilton迴路是一個NP問題，要驗證一條路是否恰好經過每一個頂點可在多項式的時間之內完成，但要找出一個Hamilton迴路卻要窮舉所有可能的答案，無法直接求解。

參考資料:

# <https://codertw.com/%E7%A8%8B%E5%BC%8F%E8%AA%9E%E8%A8%80/548457/#outline__3>

# (程式前言: P問題、NP問題、NPC問題（NP完全問題）、NPH問題和多項式時間複雜度)

[2]解法/證明:

設一個無向圖中有N個節點，若所有路由器的度數都大於等於N / 2，則Hamilton迴路一定存在。注意，“ N / 2”中的除法不是整除，甚至實數除法。如果N是偶數，當然沒有歧義；如果N是奇數，則該條件中的“ N / 2”等價於“⌈N/2⌉”

參考資料:

[https://baike.baidu.com/item/%E6%B1%89%E5%AF%86%E5%B0%94%E9%A1%BF%E5%9B%9E%E8%B7%AF](https://baike.baidu.com/item/%E6%B1%89%E5%AF%86%E5%B0%94%E9%A1%BF%E5%9B%9E%E8%B7%AF" \t "_blank)

**(百度百科: 漢密爾頓迴路 )**

\*漢米爾頓迴路 ->為NPC問題 ->也符合NP 問題

[3]

程式碼:

def hamilton(G, size, pt, path=[]):

    print('hamilton called with pt={}, path={}'.format(pt, path))

    if pt not in set(path):

        path.append(pt)

        if len(path)==size:

            return path

        for pt\_next in G.get(pt, []):

            res\_path = [i for i in path]

            candidate = hamilton(G, size, pt\_next, res\_path)

            if candidate is not None:  # skip loop or dead end

                return candidate

        print('path {} is a dead end'.format(path))

    else:

        print('pt {} already in path {}'.format(pt, path))

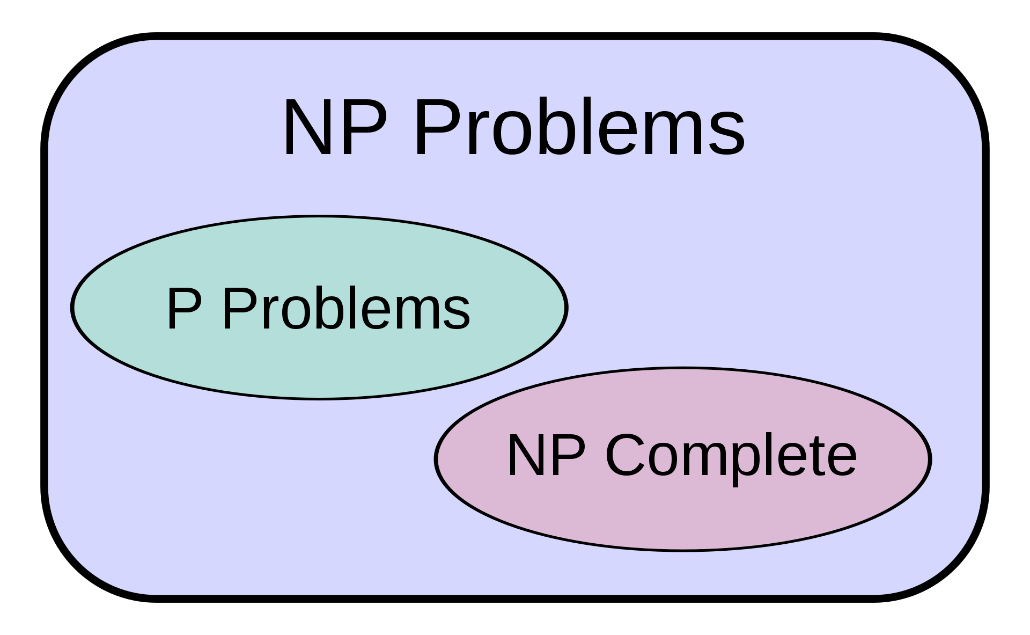
    # loop or dead end, None is implicitly returned

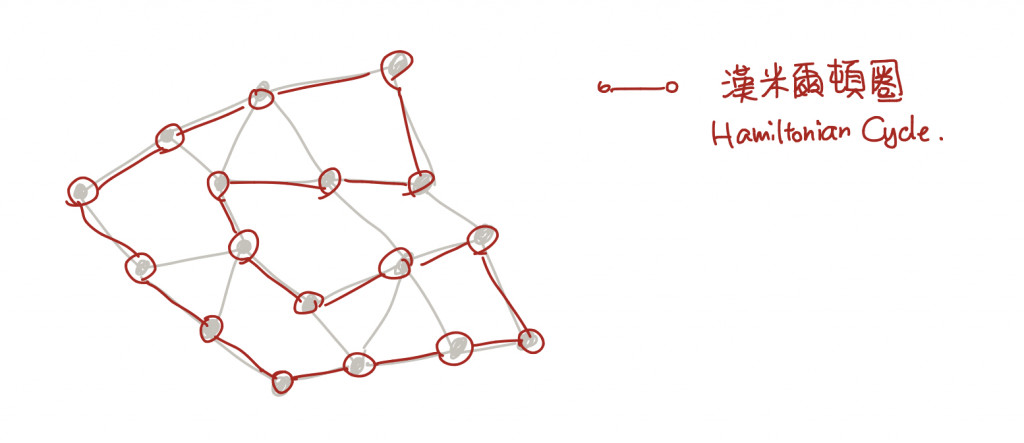
參考:

<https://stackoverflow.com/questions/47982604/hamiltonian-path-using-python>

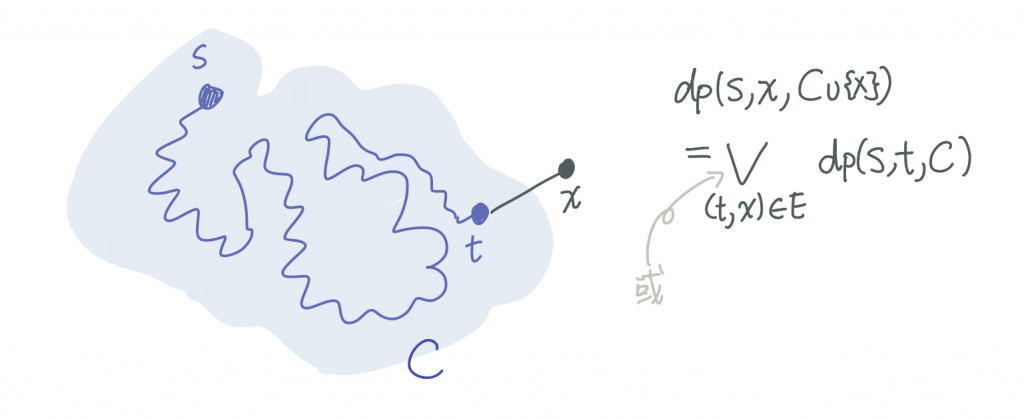
(stackflow - mkieger1)

(HAMILTION USING BY PYTHON)



**[4]**

**解題思考**

把路徑連接起來、或延伸的時候，我們可以只在乎路徑的端點、不需要在意路徑內部這些點到底是怎麼連起來的。因此這讓我們得到一個稱之為「子集合 DP」的一套動態規劃方法：對於兩個點 s, x 以及一個同時包含 s 和 x 的集合 S 來說，我們可以定義 dp(s, x, S) = 「是否存在一條漢米爾頓路徑從 s 出發、經過所有 S 中的點，然後結束在 x？」要找出遞迴也很簡單：枚舉看看 x 前一個點收在哪裡，然後把 x 從集合中釋放出來就好啦!

參考:

## <https://ithelp.ithome.com.tw/articles/10222426>

## (iT邦幫忙 - Day 15: 動態規劃可以解決一些著名的ＮＰ完備問題！ Part 2)