活動 16
冰之路 — 斯坦納樹

活動摘要

有的時候，對一個問題做一個渺小的、似乎微不足道的變動，結果會造成解決難度上巨大的改變。這個活動，類似於活動 9 的「泥濘城市問題」，也是要找到網路裡的最短路徑。然而，跟泥濘城市問題不同之處，是在可以縮短路徑的長度的前提下，允許加入一個新的節點進入網路。這一個小小的改變，把整個問題變成一個不可駕馭的問題。這個問題變得跟泥濘城市無關，而是在演算法上與「貧窮的製圖師」（活動 14）以及「旅遊小鎮」 （活動 15）一樣的問題了。

課程銜接

- 數學：位置與方向
- 數學：邏輯推理

習得技能

- 空間視覺化
- 幾何推理
- 演算法程序與複雜度

適合年齡

- 7 歲以上

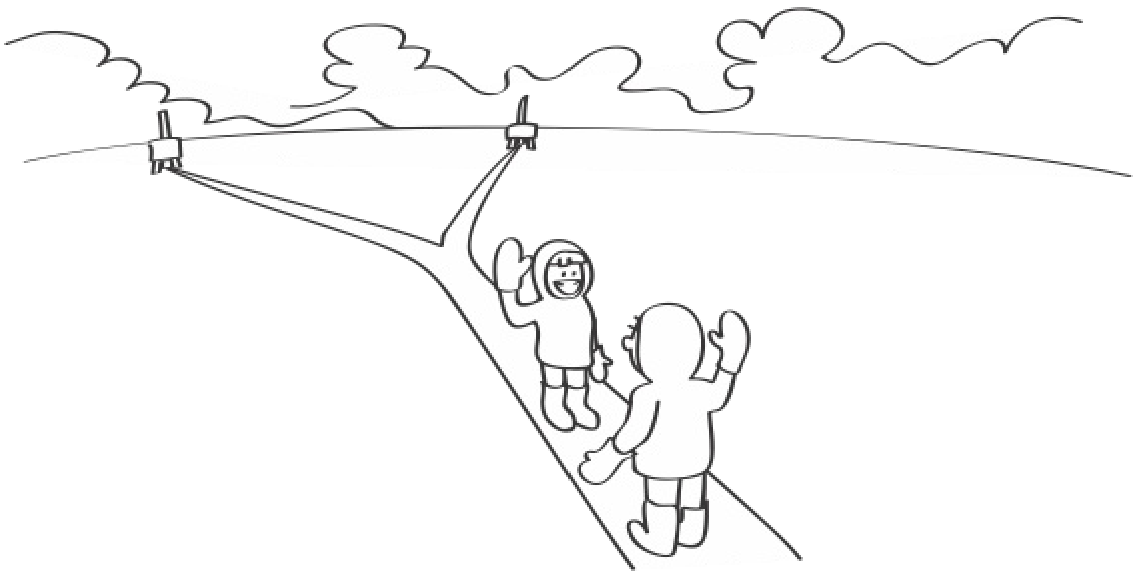
所需素材

每一組的學生會需要：

- 五或六個釘子放在地上（帳篷釘，或把衣架切成片折彎也可以）
- 幾公尺長的繩子或鬆緊帶
- 一個直尺或捲尺
- 筆跟紙用來做筆記

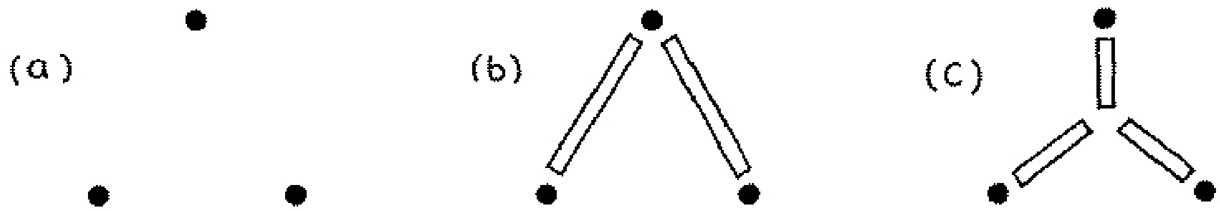
冰之路

活動介紹



先前的活動，「旅遊小鎮」的故事發生在一個非常炎熱的國家；而這個故事卻剛好相反。在加拿大寒冷的北方（夠冷了吧，這樣故事才進行得下去），冬天的時候整個湖會結冰；而掃雪機則會在巨大的冰湖上做出一條條可以連結工作站的路，讓工作人員可以拜訪彼此。而因為那裡很冷，所以他們希望道路的距離越短越好。而你的工作就是要找出路要造在哪裡。基本上沒有條件限制：湖面都是結凍而且被覆蓋住的。全部都是平坦的。路可以開在雪地上的任何地方。

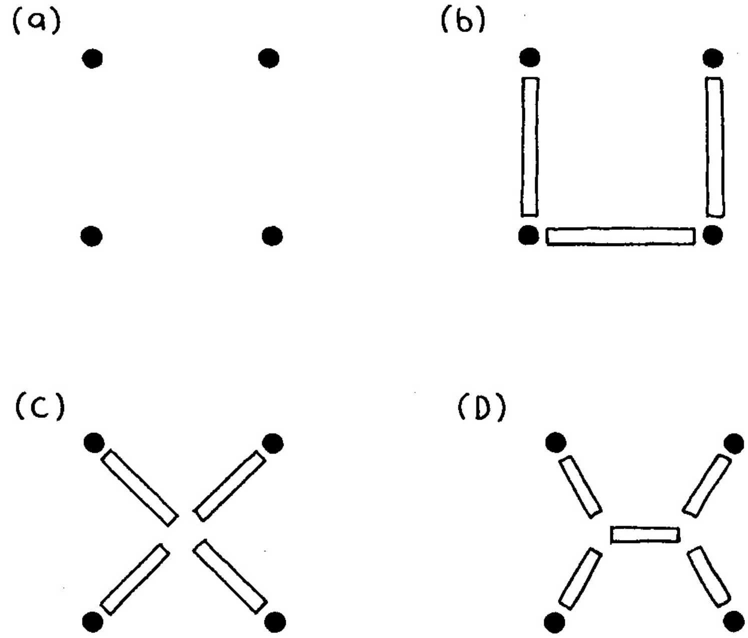
明顯地每條路應該都是直的，因為彎曲的道路會增加不必要的長度。但是不是把所有的工作站用直線連起來就好了，因為在結凍的荒地上，有時加上一個交叉點反而可以減少道路的總長度 — 要記得，重要的是道路的總長度要最短，而不是從一個工作站移動到另一個工作站的時間要最短。



在圖中，(a) 有三個工作站。把其中一個工作站跟另外兩個連起來（如 (b) 圖）是可接受的一個解法。另一個方式是在三個點的中央附近建立一個交叉點，將交叉點連結到三個工作站（如同 (c) 圖）。如果你測量一下的道路的長度，會發現 (c) 圖確實是個比較好的解法。這個額外的交叉點被稱為「斯坦納點」，因為瑞士數學家雅各 · 斯坦納（Jacob Steiner, 1796-1863）而得名。他描述了這個問題，而且是第一個指出可以藉由產生新的節點去減少道路的總長度。你可以將斯坦納點想像成一個新的，虛擬的工作站。

活動討論

1. 向大家描述接下來學生將進行的活動。使用上面的範例向學生展示，在三個點之外額外增加一個點，有時反而能夠減少所建道路的總長度。



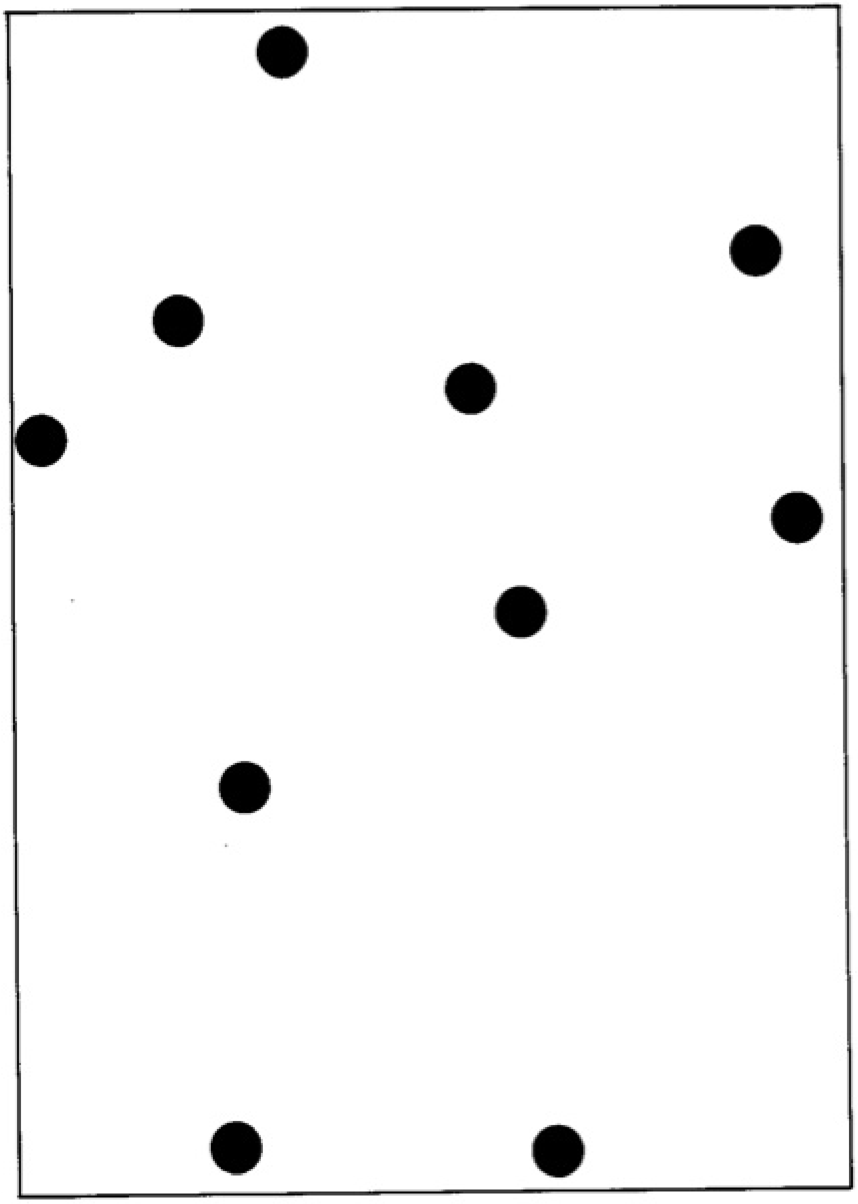
2. 學生將使用四個點排列而成的方形，如同圖 (a)。到外面去讓每一組在草地上用釘子釘出約 1 公尺見方的範圍。

3. 讓學生試試看用繩子或有彈性的物品去連接每個釘子，測量最小所需的長度並且記錄起來。在這個階段還不要加入使用任何斯坦納點。（因此這個階段，要達到總長度的最小值，要把正方形的三邊接起來，如圖 (b)，總長度為 3 公尺。）

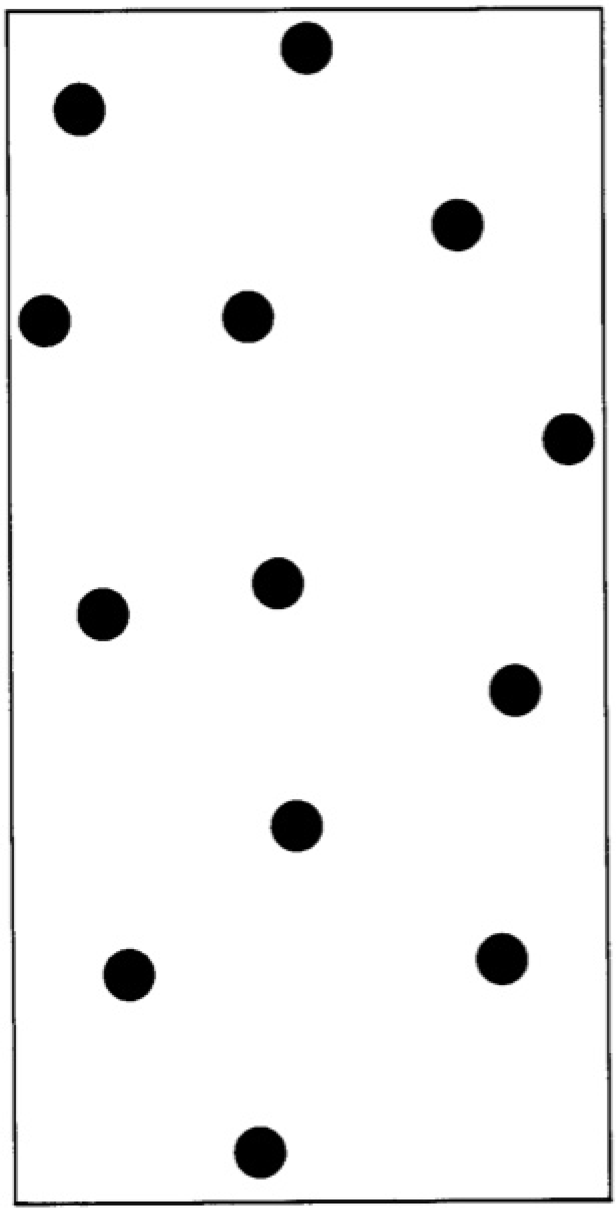
4. 現在讓學生加入一個斯坦納點，看看有沒有更短的解法。（最好的點在正方形的中心，如圖 (c)。總長度為 $2\sqrt{2}$ ）好了之後，讓大家放兩個斯坦納點，看看能不能得到更好的解法。（可以，把兩個點擺在圖 (d) 的地方，道路之間形成 120度的夾角。總長度為 $1 + \sqrt{3} = 2.73$ 公尺。）

如果加入三個斯坦納點會有更好的解答嗎？（不 — 加入兩個點已經是最佳解了。再多並不會得到更好的解答。）
與學生討論為何這些問題看起來如此困難。（這是因為不知道該在哪裡放斯坦納點，有太多可能性要試了。）

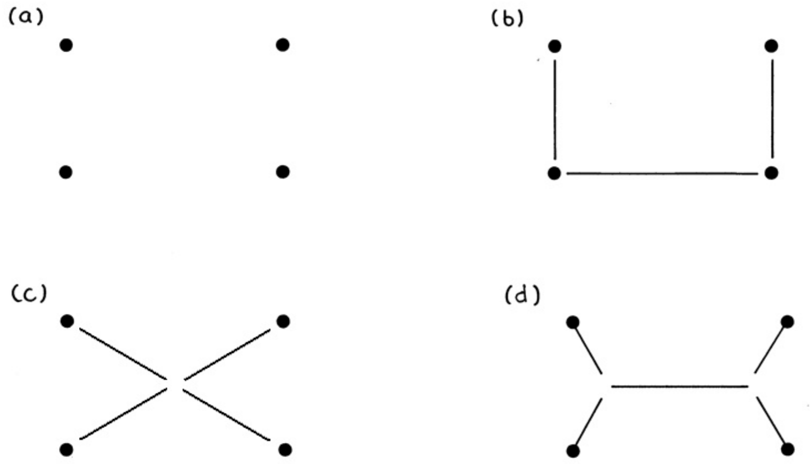
活動學習單：斯坦納樹範例 1



活動學習單：斯坦納樹範例 2

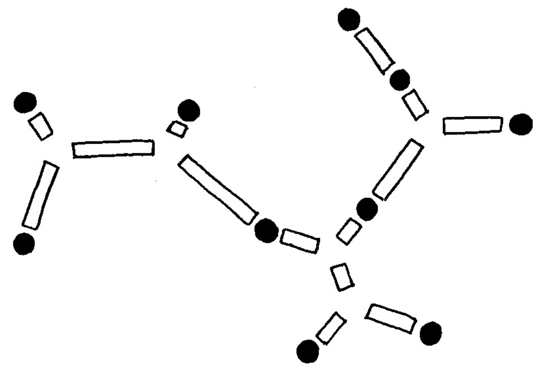


活動變化與延伸

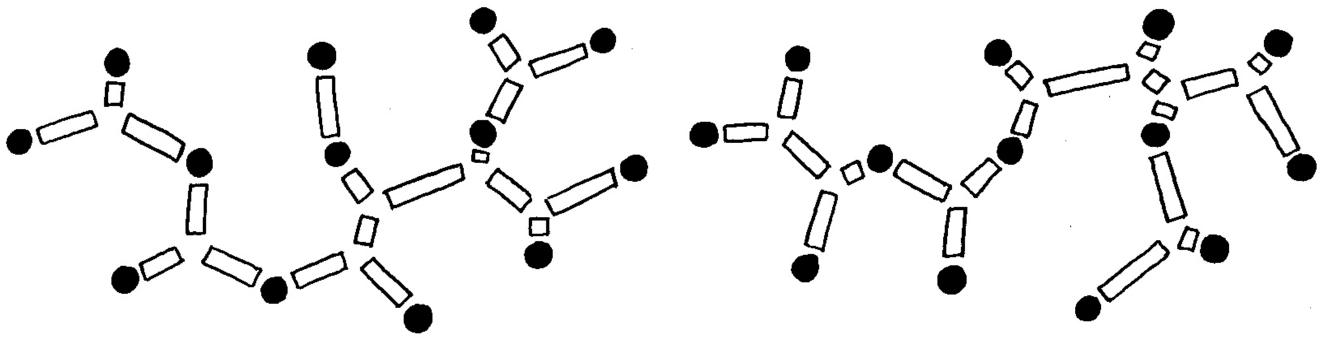


1. 如果有幾組提早完成前面的活動，可以讓他們進行一個有趣的實驗。圖 (a) 是個長 2 公尺寬 1 公尺的矩形。讓他們試試同樣的活動，學生們會發現加入一個斯坦納點反而會讓事情變糟，但是加入二個點則可以得到較短的路徑。（圖 (b) 的路徑長是 4 公尺，圖 (c) 是 $2\sqrt{5} = 4.47$ 公尺，而圖 (d) 是 $2 + \sqrt{3} = 3.73$ 公尺。）看看他們能不能夠找出為什麼在長方形中加入一個斯坦納點反而會比在正方形裡加入還糟。（這是因為當正方形拉長成長方形時，在圖 (b) 和 (d) 伸長量只有一條會增加，但圖 (c) 卻是兩條對角線都增加。）

2. 較年長的學生可以試試較大型的題目。他們可以在學習活動單上的兩張圖實驗不同的解決方式。可以把圖印出來，或用白板筆寫在蓋著透明投影片的圖上。或者到戶外去，用釘子在地上標記出來。當他們每次創下最短總距離的紀錄時可以跟其他人分享。（右圖是第一個範例的最佳解，而下頁的圖顯示的是兩種第二個範例的兩個可能的解法。兩種方式的距離相當接近。）這兩種解法如此接近，也說明為什麼這類問題會這麼困難 — 要在哪裡放置斯坦納點，有太多種選擇了。

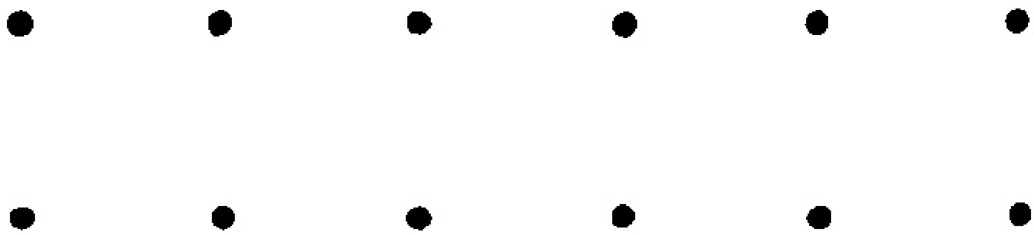


第一個範例的最佳解

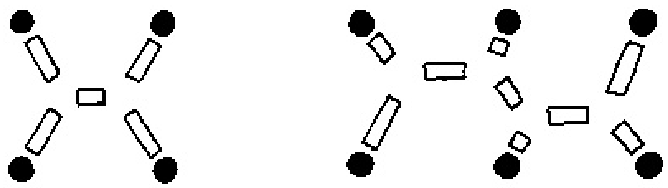


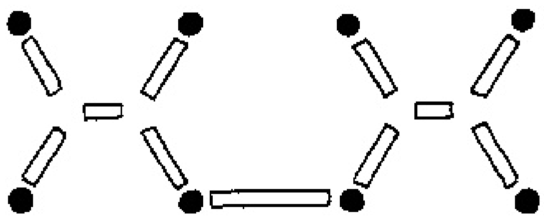
第二個斯坦納樹範例的兩種可能的解法

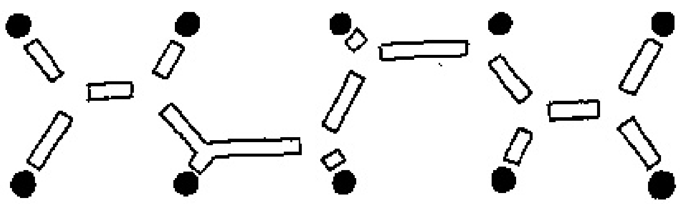
3. 梯子型網路（如下圖）也提供了此類問題的不同變化。



這裡是梯子型網路的一些最小斯坦納樹解法。

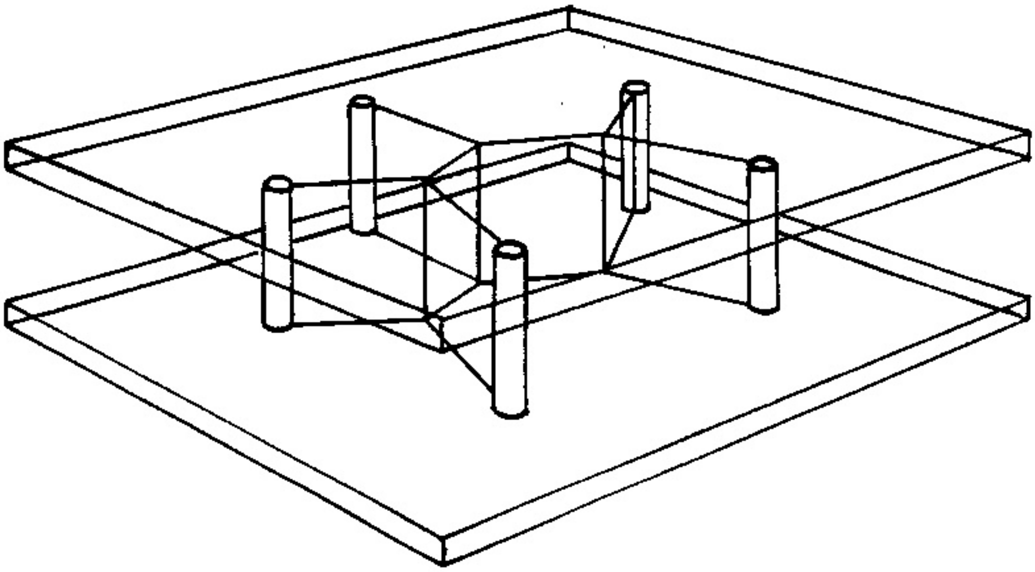






梯級為二的梯子型網路其實就是一個正方形。然而，梯級為三的梯子型網路，解法是相當不同的—試著用你記憶中的方式去畫畫看就會發現！梯級為四的解法，就跟把兩個梯級為二的梯子型網路組合在一起。而梯級為五的，又比較像梯級為三的網路的延伸。一般來說，梯子型網路的最小斯坦納樹的形狀取決於它的梯級是偶數還是奇數。如果是偶數，那就是多個梯級為二的網路組合在一起。反之為奇數，就會跟梯級三一樣的解法。但要嚴謹地這證明以上的結論，是相當不容易的。

4. 另一個有趣的活動是建構斯坦納樹的肥皂泡模型。你可以用兩片透明的投影片，然後把針插在它們中間，去表示這些點。如下圖：



接著把整個模型都浸入肥皂泡溶液中。當你把它拿出來時，你會發現肥皂泡泡的膜連著針而呈現出一個美麗的斯坦納樹網路。

然而，它所呈現出來的不一定是個最小的斯坦納樹。這些肥皂膜確實呈現出一個總距離最小化的狀態，但這個最小值只是局部的，不一定是總體的。可能有個完全不一樣的放置斯坦納點的方式，總距離更短。舉例來說，把活動學習單範例二的結構浸到肥皂泡溶液中。當它從肥皂液體中拿出來時，你可能會看到前面所說的第一種斯坦納樹，也有可能看到第二種。

這個活動在說什麼？

這個活動在講「最小斯坦納樹」。之所以叫「樹」是因為整個圖形裡沒有循環，只有像真的樹一樣分支，不會插入另一個分支或主幹上一起生長。而之所以叫「斯坦納樹」是因為可以在原來的樹所連接的工作站中間加入新的點（也就是「斯坦納點」）。最後，之所以說是「最小」是因為追求的是連接所有工作站形成的最短總路徑。在「泥濘城市」（活動 9）中，我們學到了找出連接所有點而使總長度最小所形成的樹，叫做「最小生成樹」（minimal spanning tree）：斯坦納樹也是一樣，只是在過程中允許加入新的點。

有沒有一種有效率的演算法來找出最小生成樹呢？之前我們試過貪婪法，也就是重複地找出任兩點中距離最小的，如果有尚未連接的點就把它連起來。這個方法對最小生成樹是可行的，但是對最小斯坦納樹則不然 — 目前還沒有一個通用有效率的方法。為什麼？因為在斯坦納樹問題中，你必須決定要把斯坦納點放在哪裡。事實上嘛，令人驚訝的是，困難的部份並不是精確地決定斯坦納點要放在哪裡，而是估計斯坦納點的位置：比方說在學習活動單範例二中，兩個解之間的差別其實非常的小。一旦知道要放在哪個區域，決定最佳的斯坦納點其實就只是微調位置的問題而已；而估計哪個區域可以真正縮短總長度才是困難所在。肥皂膜可以很有效率地做到這一點，電腦也可以。

尋找最小斯坦納樹對過去的電話業務來說是一個很重要，可以節省大量成本的事。在 1967 年以前，在美國的企業客戶要有自己的私有電話網路，要向電話公司租線路。他們計費的方式不是實際上鋪了多長的線路，而是以最短可以運作的網路為基準計費。理由是客戶不需要為了電話公司的方便而支付多餘的線路費用。最開始，計算要收費多少的方式是用最小生成樹。然而，在 1967 年左右，有個客戶（是間航空公司，有三個主要的電話中轉站）不小心發現了如果他們要求在某個點設第四個中轉站，線路的總長度反而會減少。電話公司只好依照假裝第四個中轉站（也就是斯坦納點）存在的狀況下計算，而被迫減少收費。雖然典型的狀況下，最小斯坦納樹只比最小生成樹少大約 5% 到 10%，但對花大錢的客戶來說，節省的成本還是很可觀的。斯坦納樹有時又被稱為「最短網路問題」，因為它就是在找連接一組工作站所需的最短網路。

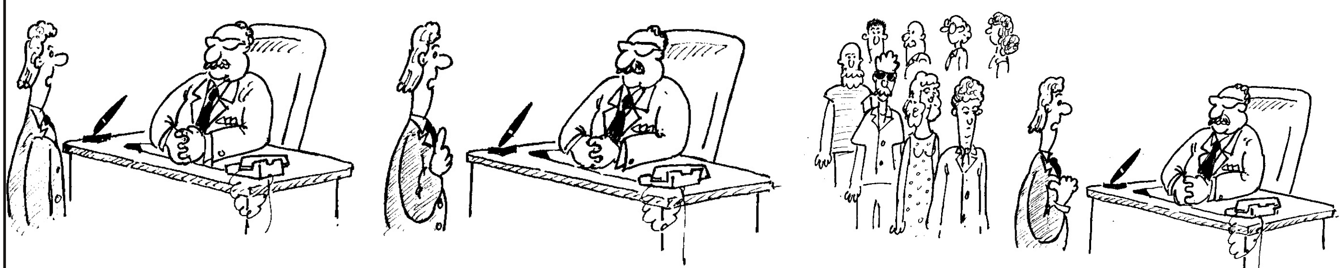
如果你有玩過前兩個活動，也就是「貧窮的製圖師」與「旅遊小鎮」，那你應該不意外 — 斯坦納樹問題也是一個「NP 完全問題」。當工作站的數量增加，可能的斯坦納點的樹量也會跟著增加，而找出所有可能性的時間就會成指數成長。這個問題跟其他成千上萬的問題一樣，目前還不知道到底有沒有多項式時間演算法來解決這個問題。如果有，那它的多項式時間演算法就可以轉換成著色問題，還有最小支配集，還有許許多多其他 NP 完全問題的多項式時間演算法。

我們在前一個活動的最後解釋了什麼叫做「NP」：「非確定性多項式」（non-deterministic polynomial），而「完全」（complete）指的是如果在其中一個問題找到的一個有效率的方式來解決，那麼此方法可適用於解決同組中任何一個其他的問題。可以用多項式時間演算法來解決的問題就稱為 P。那麼，關鍵的問題來了：NP 完全問題是否有多項式時間演算法可以解？換句話說，

P = NP 嗎？這個問題目前還沒有答案，而且是現代資訊科學中最大的謎之一。

那些可以用多項式時間演算法解決的問題 -- 即使看起來要花很多時間 — 稱為「可駕馭問題」（tractable）。反之，還沒找到多項式時間演算法的問題，則稱為「不可駕馭問題」（intractable），因為不管你的電腦多快，或是同時用多少台電腦下去跑，只要問題的規模做小規模的增加，就有可能變成現實上無解的狀況。目前 NP 完全問題，包括前面說到的貧窮的製圖師、旅遊小鎮、冰之路等等，還不知道是不是屬於易解問題。但是大多數的電腦科學家對 NP 完全問題是否找得到多項式演算法解是持悲觀的態度。所以證明某問題為 NP 完全問題也同時被認為是本質上屬於難解問題的有力證據。

想像一下，如果你的老闆要求你對一個問題設計出一個有效率的演算法並找到最佳解，但是你做不到時怎麼辦？當然，如果你能夠證明這個問題沒有一個有效率的演算法得到最佳解，這樣最好。但在資訊科學的領域中，做這種負面結論的證明是非常困難的。因為誰也不知道未來會不會有個聰明的傢伙，偶然發現一個不起眼的方法，把整個問題解決了，就像前面的航空公司發現加上一個斯坦納點可以節省網路的成本那樣。所以，你其實很難告訴老闆說問題沒有有效率的解。不過，你可以證明該問題是個 NP 完全問題，並告訴老闆說已經有成千上萬的人在努力解跟老闆丟給你一樣的問題，而他們目前仍一無所獲。嗯，這樣是不會幫你加分啦，但至少可以讓你脫離困境！



「我找不到一個有效的演算法，我猜是因為我太笨了。」

「我找不到一個有效的演算法，因為這種演算法不存在。」

「我找不到一個有效的演算法，不過這些有名的人也一樣。」

你找不到一個有效演算法時的做法：三種可能性

當然啦，在現實生活中這些問題都還是要解決，而這種狀況下人們就退而求其次 — 找一個「啟發式演算法」（heuristics algorithm），也就是說這類演算法不保證找到最佳解，但是可以找到最佳解的一部份。啟發式演算法可以非常快，而找不到最佳解時所耗費的成本也相對少，因此足以解決現況。只是一旦找到比現有更好一點點的課表，或網路，或路線的時候，會覺得之前怎麼那麼笨。

延伸閱讀

上面的圖片來自 Garey 與 Johnson 的經典教科書 "Computers and Intractability"。

1984 年六月出版的美國科學雜誌（Scientific American）裡的「電腦娛樂」專欄中，有關於如何用肥皂泡泡產生斯坦納樹，還有很多其他模擬小工具的有趣描述，像是排序用的義大利麵電腦，用來在圖形中找最短路徑的貓咪的搖籃，還有判定一個數是否是質數的光反射鏡的裝置等等。這些東西也在 Dewdney 的 Turing Omnibus 裡關於模擬電腦的章節也有提到。