

活動 15

旅遊小鎮 — 支配集

活動摘要

很多現實生活上的情況都可以用網路或是像活動 14 中用來著色的圖的形式來表達。網路也提供很多開發特定用途演算法的機會。在這個活動中，我們在一些連接點或節點上做標記，讓所有沒標記的節點最多只要一步就可以到達有標記的節點。問題來了，我們最少要標記多少節點？這是個有趣且出乎意料困難的問題！

課程銜接

- 數學：位置與方向
- 數學：邏輯推理

習得技能

- 地圖
- 關係
- 解決問題
- 迭代目標搜尋法 (Iterative goal seeking)

適合年齡

- 7 歲以上

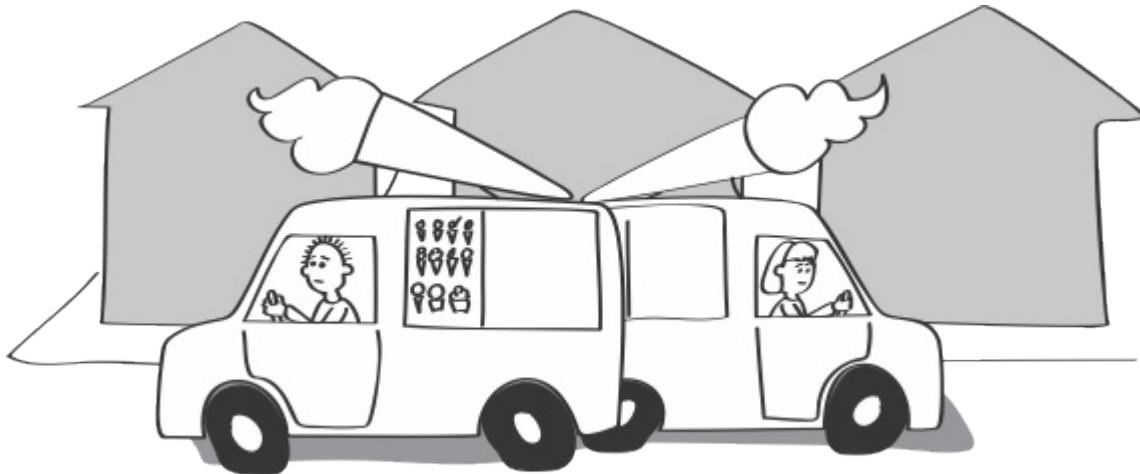
所需素材

每組學生需要：

- 冰淇淋車的圖
- 一些籌碼或是兩張不同顏色的撲克牌

你會需要：

- 冰淇淋車的投影圖，或是直接畫在白板上

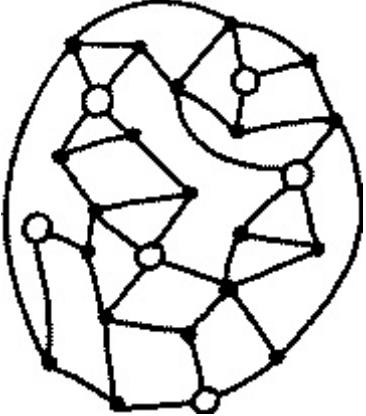


支配集 (Dominating Sets)

活動介紹

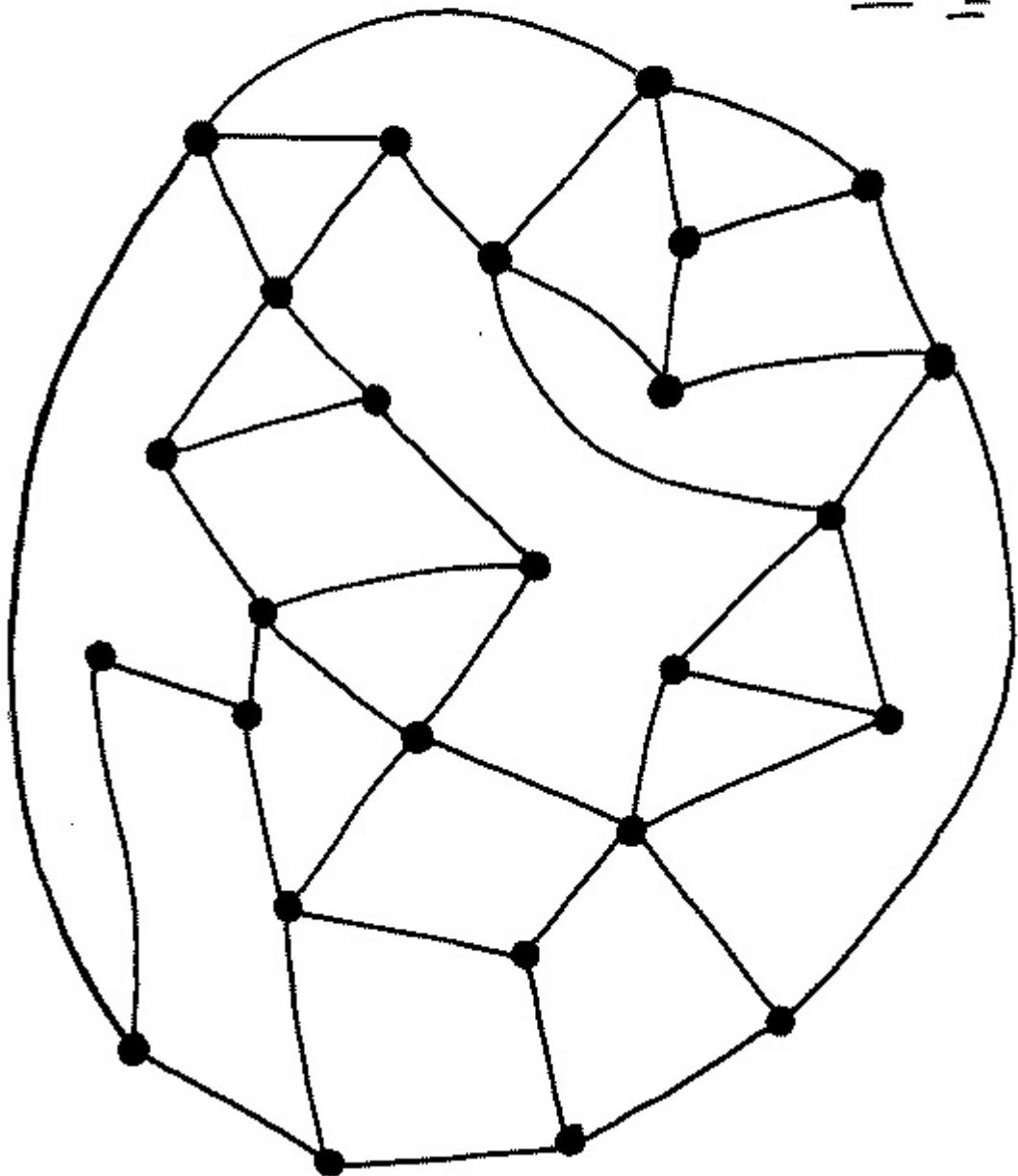
「活動學習單：冰淇淋車」上畫著一個旅遊小鎮的地圖。每一條線代表一條街道，而每個點則代表街角。這個城鎮所處的地區氣候炎熱，因此在夏天時冰淇淋箱型車都會停在街角賣冰淇淋給觀光客。我們現在要設立冰淇淋車駐紮的據點，讓每一位觀光客都能在每條街底找到廂型車，最多只需要走一條街即可買到冰淇淋。（也可以想像居民們大都居住在街口而非街道中，因此他們出門之後最多走到下一個街口就一定可以買到冰淇淋）。所以，我們的問題是：我們需要多少台冰淇淋車，並且應該設立在哪些街角？

活動討論

1. 將學生分組，並給各組一個旅遊小鎮的地圖，並說明故事內容與問題。
2. 說明該如何在某一個十字路口上放一個籌碼，標記為冰淇淋車，然後在其相距一條街的街口放另一個顏色的籌碼。這台冰淇淋車的服務範圍即為這些籌碼所標記的位置，也就是住在這個十字路口以及相鄰街上的居民。
3. 讓同學嘗試在不同位置駐紮冰淇淋車，找出能夠讓每個居民都能買到冰淇淋的駐紮方式。提醒同學，此問題涉及成本考量，冰淇淋車的駐紮據點要在能夠達成題目要求的情況下越少越好，這也是這個問題最有趣的地方。
4. 旅遊小鎮所需要的冰淇淋車最少數目為 6，而解答如右圖所示。這個解答並不好找喔！因此大家摸索一陣子之後，要跟同學們提示，此題的所需的冰淇淋車數目最少為六台。因為即使增加到八台或九台車去解難度都頗高，因此最後可能會有許多組放棄。鼓勵大家多去嘗試。
5. 這個旅遊小鎮的地圖製作，是由「活動學習單：冰淇淋車（解答）」下面的六個地圖區塊所開始組成。每個區塊只需要一個冰淇淋廂型車。可以畫上街道來將這幾個地圖區塊拼湊起來，將答案隱藏起來。重點不是將所有冰淇淋車據點（空心點）通通連接起來，而是要放在將據點與服務範圍（實心點）間的連結。將這個方法用板書或投影片示範給同學看。
6. 讓同學嘗試用這個的方法製作地圖。相信同學也會喜歡製作一個只有自己解得出來的題目！這個題目即是一個「單向函數」(one-way function) 的例子—除了製作謎題的人，其他人沒有辦法輕易解出這道謎題。「單向函數」在密碼學中佔有一個非常重要的角色（請參見活動 17、18）。

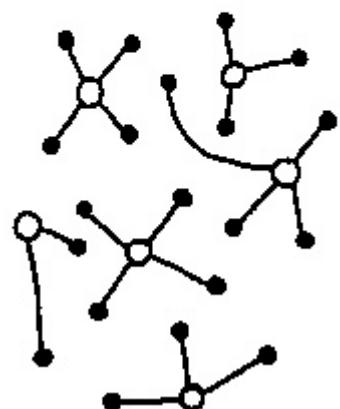
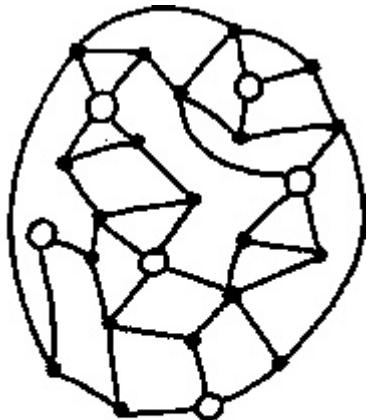
活動學習單：冰淇淋車

請找出如何在交叉路口擺放冰淇淋車，使所有的交叉路口都可以相連到有車子的交叉路口。



活動學習單：冰淇淋車（解答）

對同學顯示下面的圖來說明題目是如何建構的。



活動變化與延伸

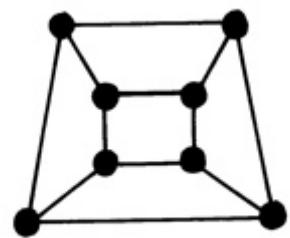
在做城市規劃時會有很多情況面臨到這類問題，像是擺設郵筒、井、消防局等等。而在現實生活中的地圖也不會用那麼簡單的方式來組成。如果你真的當上市長，需要解決這類問題的話，你會怎樣做呢？

有個比較直接的方法是：把所有擺放冰淇淋車的方法列出來，然後看看哪一個是最適合的。旅遊小鎮內有 26 個街口，所以有 26 種方法可以擺放一台冰淇淋車。檢查所有 26 種可能性不難，但是很明顯地這些方法都沒有滿足所需的條件。用兩輛冰淇淋車的時候，有 26 個地方可以擺第一台車，然後剩下 25 個地方可以擺第二台車（因為你總不會在同一個路口擺兩輛車吧！）。所以總共有 $26 \times 25 = 650$ 種可能性要檢查。嗯，要全部檢查嘛，也是不難，但已經開始有點讓人抓狂了。真正說起來，你只需要檢查一半（325 種可能性），因為車子的順序不重要：如果你已經看過車子 1 在交叉路口 A 和車子 2 在交叉路口 B 的情況，則沒有必要再檢查車子 1 在 B 和車子 2 在 A 的情況。三輛車子呢？有 2600 種可能性。四輛車子呢？有 14950 種可能性……依此類推。當然，因為只有 26 個交叉路口，並且同一個地方也不需要兩台車，因此最多 26 台車就可以了。另一種評估可能性的方式是考慮所有可能的配置：26 個交叉口和任何數量的貨車。由於每個路口只有兩種可能：有車或沒有車，因此所有配置的可能性的數量是 2^{26} ，也就是 67,108,864。

這種方法被稱為「暴力」演算法 (brute-force)，通常需要很長的時間才能完成。大家常以為電腦計算的速度超級快，不管有多少工作，都可以在很短的時間內解決幾乎所有的問題，但實際上並沒有。暴力演算法需要多長的時間，取決於它檢查一種可能性是否是最佳解的時間。在冰淇淋車的問題中，要檢查一種可能性，得要檢查每一個路口，來確認最近的車子的距離。假設每一種可能性可以在一秒鐘檢查完，那旅遊小鎮的地圖需要多長的時間來測試所有 2^{26} 種可能性？（答案： 2^{26} 大約等於 6700 萬；一天是 86400 秒，所以 2^{26} 秒相當於 777 天左右，也就是大約兩年。）假設一種可能性不用到一秒，我們算千分之一秒好了。那麼在相同的兩年，電腦大約可以找出 36 個路口的配置，因為 2^{36} 大約是 2^{26} 的 1000 倍。好，即使電腦快一百萬倍，也就是一秒鐘可以檢查一百萬種可能性，那兩年的時間一樣只能處理 46 個路口的冰淇淋車配置。這些都不算是非常大的城市喔！（想想看，你住的城鎮有多少交叉路口？）

既然暴力演算法是如此地慢，有沒有其他的方法來解決這個問題呢？我們可以試看看在泥濘城市（活動 9）中非常成功的「貪婪方法」。我們需要想想看如何貪冰淇淋——我的意思是，如何應用貪婪的方法來解決冰淇淋車問題。這種方法是把第一台冰淇淋車放在相鄰交叉路口最多的路口，然後第二台車就在相鄰交叉路口第二多的路口，依此類推。然而，這樣做並不一定會產生冰淇淋車擺設最少的最佳解——實際上，在旅遊小鎮中，最多相鄰節點（5 個）的節點，並不適合擺冰淇淋車（跟班上同學確認這一點）。

讓我們來看一個比較簡單的問題。假設我們的目的只是要找出一個配置方法，不一定要最小或最佳解。在某些情況下，這相對容易許多。例如，這張簡單點的地圖，解答就很直覺。如果把地圖中的街道想像成立方體的邊，那很明顯地，只要擺兩台冰淇淋車立方體斜對面的頂點就夠了。此外，你應該也很清楚，不可能少於兩台車。而在旅遊小鎮的地圖中，就很難 — 雖然也不是不可能 — 說服大家相信只需要六台車就夠了。



這個活動在說什麼？

關於冰淇淋車問題，一件有趣的事情是，沒有人知道是否有一個尋找一組最佳解的演算法比暴力法明顯更快！暴力法花費的時間隨著交叉路口的數目呈指數增長 — 因此它又被稱為「指數時間演算法」(exponential-time algorithm)。而相對於指數時間演算法，「多項式時間演算法」(polynomial-time algorithm) 是指演算法的執行時間會隨著變數的平方、立方、十七次方或其他數目次方而增長。

多項式時間演算法，即使是十七次方，只要是夠大的地圖，也會比暴力法更快。因為指數級成長在它的參數大到一個程度時，一定會超過任何多項式級的成長（比方說， n^{17} 跟 2^n 來比較，哪個比較大？在 n 大於 117 之後，後者就會超越前者）。那麼，是否有一個多項式時間演算法可以來尋找最少需要在哪些位置？目前還沒有人知道，但大家已經很努力的在找了。而對於一個比較容易的問題：檢查一組特定的位置是否是最佳解也是一樣，使用暴力法所花費的時間是呈指數成長，而針對此問題的多項式時間演算法，既沒有被發現，也沒有被證明不存在。

這是否讓你想起地圖著色問題（活動 14）？你應該要想起來喔。冰淇淋車的問題，正式名稱為「最小支配集」(minimum dominating set) 的問題，是目前尚未確定多項式演算法的解法是否存在的問題之一。這一類的問題，範圍從邏輯，鋸狀排列問題到地圖著色問題，在地圖上找最佳路徑，還有調度流程問題等等，問題可以說成千上萬。但令人驚訝的是，所有的這些問題已被證明是相同的：如果其中一個問題找到了多項式時間的演算法，那其他所有的問題就可以經由此演算法做轉換而一起解決 — 要嘛就全部有解，要嘛就全部無解。

這些問題被稱為「NP 完全問題」(NP-complete problems)。NP 代表「非確定性多項式」(non-deterministic polynomial)。這可是內行人才會用的術語喔！這個術語表示只要有一台電腦可以立刻測試任意量的解決方案（所謂的「任意量」就是「非確定性」的部分），這個問題可以在合理的時間內解決。你可能會覺得這什麼鬼？未免也太不切實際了吧？沒錯，的確是這樣。我們的確不可能建立這種類型的電腦，因為它必須是非確定性的任意大！然而，這種機器的概念在原則上是很重要的，因為 NP 完全問題不能在沒有非確定性電腦的情況下在合理的時間量被解決。

此外，這一組問題被稱為「完全」是因為雖然問題似乎非常不同 — 例如，地圖著色問題和冰淇淋車問題看起來非常不同 — 但經過證明，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。這就是我們所謂的「要嘛就全部有解，要嘛就全部無解」。

有成千上萬的 NP 完全問題，研究人員一直在努力尋找有效率的解決方案，但幾十年來還沒有好消息。就像剛剛說的，只要找到其中一個問題的有效解法，那所有問題就可以一起被解決。不過也是因為這樣，它被強烈懷疑是不是根本就沒有有效率的解法。不管怎樣，證明這些問題必然需要指數時間，在理論資訊科學 — 說不定是在整個數學界 — 來說，都是今日最著名的開放問題。

延伸閱讀

Harel 的書 “Algorithmics” 介紹了幾種 NP 完全問題，討論多項式時間演算法是否存在的問題。Dewdney 所著的 “Turing Omnibus” 也討論了 NP 完全問題。關於這個問題的標準資訊科學教科書是 Garey & Johnson 的 “Computers and Intractability”，裡面介紹了數百個 NP 完全問題，並介紹證明 NP 完全性的技巧。然而，這本書真的很難，只適合於資訊科學專業人員。