

# 活動 18

## 秘魯式拋硬幣 — 加密協定

### 活動摘要

這活動讓大家瞭解，如何實現一個簡單但看似無法達成的任務。在兩個只透過電話溝通且互不信任的人之間，透過拋硬幣做出一個公平隨機的選擇。

### 課程銜接

- 數學：邏輯推理
- 數學：布林邏輯

### 習得技能

- 布林邏輯
- 函數
- 解謎

### 適合年齡

- 9 歲以上

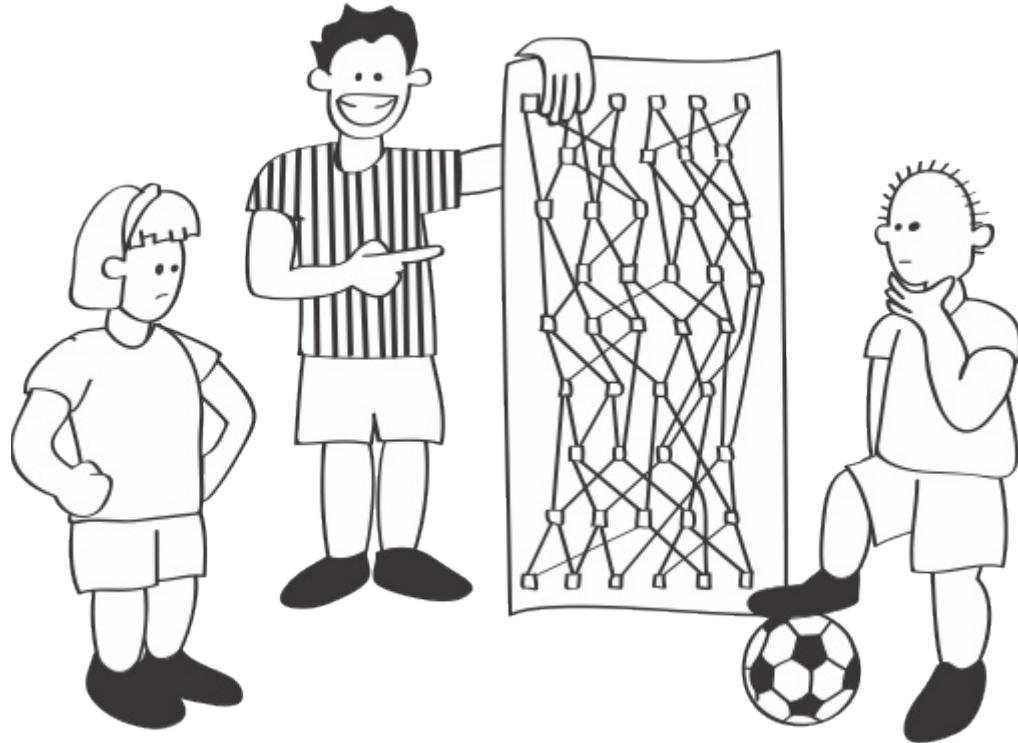
### 所需素材

每組學生需要：

- 活動學習單：秘魯式拋硬幣（第 203 頁）的影本
- 兩種不同顏色的按鈕或籌碼大約兩打



## 秘魯式拋硬幣



### 活動介紹

這個活動是本書的其中一個作者（MRF）在祕魯和學生上課的時候所發明的。你也可以想一個故事來套入這種情境。

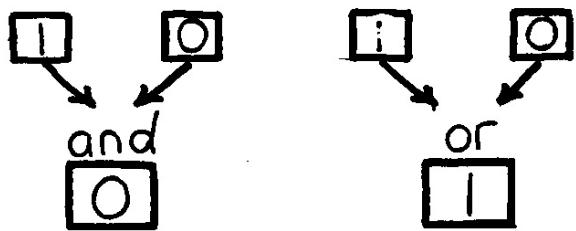
利馬跟庫科斯這兩支足球隊要決定冠軍賽的主場球隊是哪一隊，而拋硬幣決定是最簡單的方法。但是因為這兩個城市相隔十分遙遠，代表利馬的 Alicia 與代表庫科斯的 Benito 不能只是為了拋硬幣而花很多時間和金錢來見面。那他們可以藉由打電話來做到這件事嗎？比方說，讓 Alicia 拋硬幣而 Benito 猜是正反面。但這是不可行的，因為只要 Benito 猜正面，Alicia 只要說「抱歉，是反面」就能輕易取得主場優勢。Alicia 實在不奸詐，但這畢竟是個重要的比賽，對她的誘惑極大，而就算 Alicia 非常誠實好了，Benito 輸了的話又怎麼會信服呢？

如果學生已經學過二進位表示法（活動 1）、同位元概念（活動 4）、還有單向函數（活動 15）的例子，那會對這活動更為上手，並學得更多。

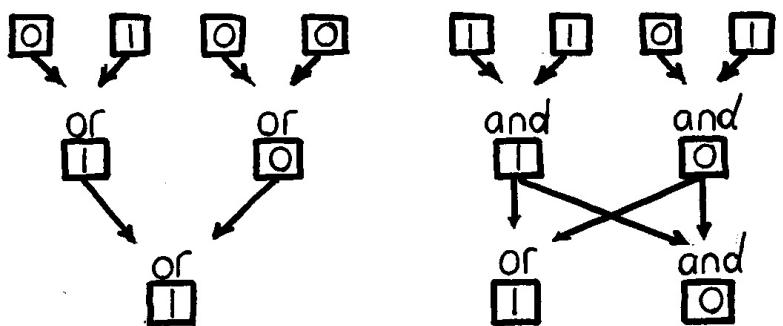
最後他們決定一起設計一個由「And 邏輯閘」與「Or 邏輯閘」所組成的電路。原則上他們能透過電話（或是用電子郵件也可以！）做這件事。雖然免不了在實作的過程中會有些乏味，但在過程中，其中的樂趣就是確定這個電路足夠複雜而另一人無法作弊，而最後設計出來的電路是大家一起想出來的智慧結晶。

## 活動討論

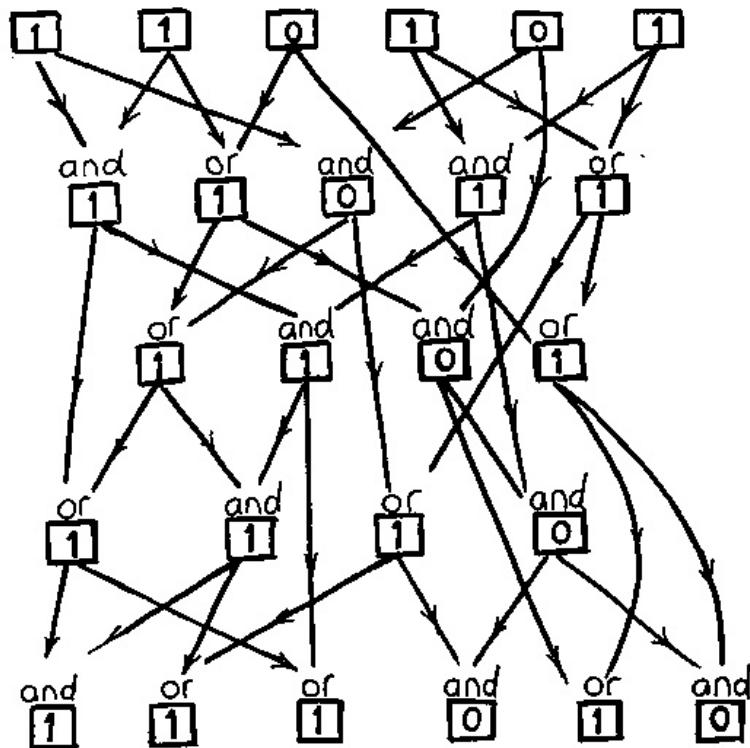
「And 邏輯閘」與「Or 邏輯閘」的原理很簡單。每個閘都有兩個輸入和一個輸出。輸入可以是 0 或 1，分別代表假 (false) 和真 (true)。當兩個輸入都是 1 的時候，And 邏輯閘的輸出是 1，其他種狀況下輸出都是 0。舉個例子，圖中的 And 邏輯閘的輸入為 0 和 1 (在上方)，所以輸出 (底部正方形) 是 0。而 Or 邏輯閘只要在其中一個輸入是 1 時，輸出就會是 1，只有兩個輸入都是 0 時，輸出才會是 0。換言之，當輸入是 1 和 0 時，輸出就會是 1。



而一個閘的輸出可以再連結成其他閘的輸入，以此形成更複雜的電路。舉個例子，左手邊 Or 邏輯閘的輸出為第三個 Or 邏輯閘的輸入，因此 4 個輸入裡只要有一個是 1，最後的結果就會輸出 1。而右邊的電路，上方兩個 And 邏輯閘的輸出會餵進去下面兩個閘，所以這個電路總共會有兩個輸出。



對這個拋硬幣活動，我們需要更多複雜的電路。書上的電路共有 6 個輸出和 6 個輸入，下方是某種輸入後結果的舉例。



透過電話進行拋硬幣遊戲的電路操作方法如下，Alicia 選擇一組隨機的六個二進位位元（0 跟 1）輸入給這組電路，這一組輸入要保密。接著她告訴 Benito 電路輸出的結果，而 Benito 就必須猜測 Alicia 的輸入是奇數個 1 還是偶數個 1—換句話說，他要猜測的是其奇偶性。如果這個電路夠複雜，那 Benito 就無法隨意算出答案，所以只能隨機地做個選擇（就好像丟硬幣來猜一樣）。如果 Benito 猜對了，則那庫科斯就得到主場優勢。反之，利馬得到主場優勢。當 Benito 把他的猜測告訴 Alicia 之後，Alicia 會把她保密的輸入告訴 Benito，而 Benito 也可以藉由同樣的電路來證實 Alicia 告訴他的輸入是否正確。

1. 把學生分為兩組，給每組電路圖和一些籌碼，並告訴他們這個故事。可以把情境設定成跟學生比較相關，比方說，學生參與的運動校隊正在決定冠軍賽的先攻後攻。規定一下籌碼的顏色—比方說紅色是 0，藍色是 1—然後叫學生在這張紙上面的記下來以免搞錯。
2. 示範如何利用籌碼把 Alicia 選擇的位元放進輸入裡，然後在這張紙的下方解釋 And 邏輯閘與 Or 邏輯閘的規則（可以叫學生用顏色標記）。
3. 示範如何把籌碼放在每個節點上，然後經由電路產生相對應的輸出。這必須要小心不要弄錯。下表（不能給學生看到）有對應每種輸入的正確輸出，可以讓老師確認學生的結果是否正確。

<b>Input</b>	000000	000001	000010	000011	000100	000101	000110	000111
<b>Ouput</b>	000000	010010	000000	010010	010010	010010	010010	010010
<b>Input</b>	001000	001001	001010	001011	001100	001101	001110	001111
<b>Ouput</b>	001010	011010	001010	011010	011010	011010	011010	011010
<b>Input</b>	010000	010001	010010	010011	010100	010101	010110	010111
<b>Ouput</b>	001000	011010	001010	011010	011010	011010	011010	011010
<b>Input</b>	011000	011001	011010	011011	011100	011101	011110	011111
<b>Ouput</b>	001010	011010	001010	011010	011010	011010	011010	011010
<b>Input</b>	100000	100001	100010	100011	100100	100101	100110	100111
<b>Ouput</b>	000000	010010	011000	011010	010010	010010	011010	011010
<b>Input</b>	101000	101001	101010	101011	101100	101101	101110	101111
<b>Ouput</b>	001010	011010	011010	011010	011010	011010	011010	011010
<b>Input</b>	110000	110001	110010	110011	110100	110101	110110	110111
<b>Ouput</b>	001000	011010	011010	011010	011010	111010	011010	111011
<b>Input</b>	111000	111001	111010	111011	111100	111101	111110	111111
<b>Ouput</b>	001010	011010	011010	011010	011010	111010	011010	111011

4. 現在每組都都分成兩半，並分別選出一個人當 Alicia，一個人當 Benito。Alicia 要隨機選擇一組輸入放到電路中並計算輸出，然後把結果告訴 Benito。Benito 接著要猜測輸入的奇偶性（有奇數個還是或偶數個 1）。這個過程中必須確定 Benito 的猜測也是隨機的，然後 Alicia 會跟大家說她所選擇的輸入值。如果 Benito 猜對的話他就贏了。Benito 可以藉由在電路中檢驗輸出是否正確來驗證 Alicia 有沒有亂說。

然後這個拋硬幣就算完成了。

但如果 Benito 可以輕易找到一組輸入產生相同的輸出結果，他就可以作弊了。所以 Alicia 必須確定這個電路的計算是個單向函數（如活動 15、活動 16 討論的），才能防止 Benito 作弊。單向函數就是在有輸入值時，可以很容易計算輸出值；但是只有輸出值時會很難算出輸入值為何。

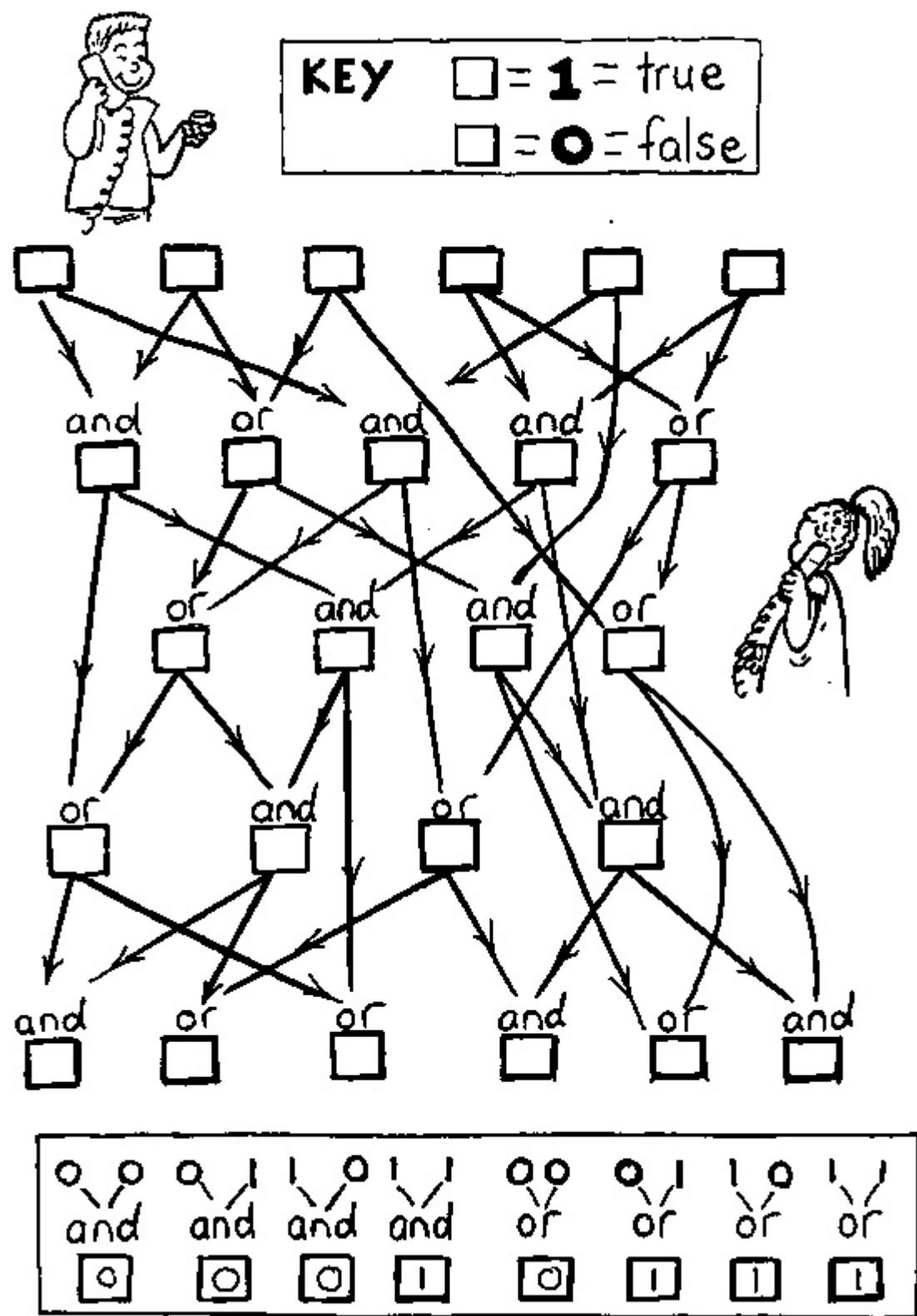
相對地，Alicia 如果可以找到不同奇偶性的輸入可以產生相同輸出，那就變成她就可以作弊了。因為不管 Benito 猜甚麼答案，Alicia 都可以說他是錯的，因此 Benito 也必須確定這個電路不會有多種輸入可以產生相同輸出。

5. 看看學生能不能找出讓 Benito 和 Alicia 作弊的方法。從這圖表的第一行，可以看到有好幾種輸入（000001, 000011, 000101）會產生相同的輸出 010010。因此如果 Alicia 紿的輸出是 010010，而 Benito 猜偶數個 1，Alicia 可以說她輸入的是 000001；如果 Benito 猜奇數個 1，Alicia 也可以說輸入是 000011。

這電路對 Benito 來說是難以作弊的。但如果輸出是 011000，那輸入一定是 100010—沒有其他的可能性了（可以看圖表確認）。若 Alicia 紿 Benito 的輸出剛好是 011000，那 Benito 猜偶數的時候就會是正確的，而且沒有爭議性。電腦系統會用更多位元，想作弊時就會多出更多可能性要試。

6. 現在讓每組學生為這遊戲設計出自己的電路，看他們能不能設計出讓 Alicia 可以作弊，或讓 Benito 可以作弊的電路。這電路不一定要六個輸入位元，甚至可以讓輸入與輸出的位元數不同。

## 活動學習單：秘魯式拋硬幣



選擇電路的輸入值，並算出輸出值。

## 活動變化與延伸：

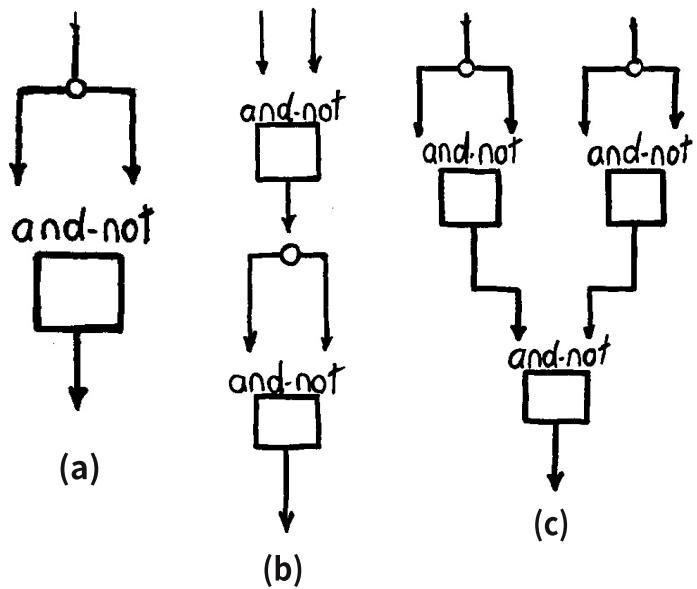
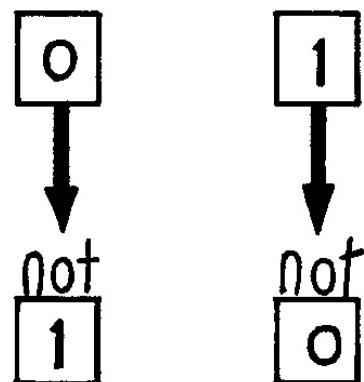
1. 這個方法在實務上有個明顯的問題，就是得建立一個 Alicia 和 Benito 都能接受的電路。這對小孩來說可能有趣，實際上卻可能使得整個程序無法進行—特別是透過電話！然而，有一個簡單的替代方案。Alicia 和 Benito 可以分別建構自己的電路，並把它們公開。然後 Alicia 分別在兩個電路輸入她選擇的位元，然後並比較兩個輸出對應的位元。如果兩組在某一個位元相同，那個位元的最後輸出就是 1，反之則為 0。在這樣的狀況下，只要有一個電路是單向函數，它和其他電路的組合也會是單向函數，參與者就無法作弊。

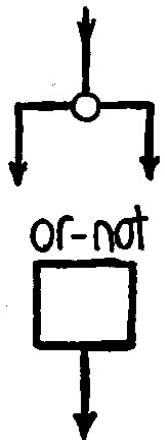
接下來的兩個變化型與加密協定或擲硬幣無關，而是關於建構一個沒有 And 邏輯閘與 Or 邏輯閘的電路。這裡探討了一些不只是電腦的電路，還有邏輯本身的重要概念。這種邏輯被稱為「布林代數」(Boolean Algebra)，以紀念數學家喬治·布爾 (George Boole, 1815-1864)。

2. 學生們可能會注意到，全 0 的輸入，也就是 000000，會產生全 0 的輸出。同樣的，全 1 的輸入，也會產生全 1 的輸出。(當然，也是有並非輸入全零，卻輸出全零的例子，反之亦然)。這結論的確是根據 And 與 Or 邏輯閘做出來的。而借由使用 Not 邏輯閘(使輸出輸入相反)，我們就可以輕易做出非(全 0 輸入與全 0 輸出)這樣的結果。

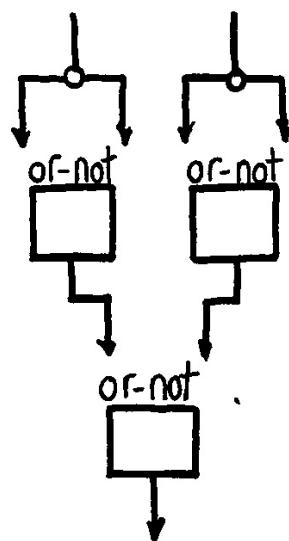
3. 另外有兩個重要的閘：Nand 和 Nor。他們代表 And 和 Or 邏輯閘後面接上一個 Not 邏輯閘。也就是說，把 a 跟 b 輸入到 Nand 邏輯閘，得到的輸出是 NOT (a AND b)。這看起來有點多此一舉，因為只要在 And 邏輯閘後面加一個 Not 邏輯閘就可以了。但是有一件有趣的事：所有其他的邏輯閘都可以只用 Nand 邏輯閘兜出來。只用 Nor 邏輯閘也可以。

向學生介紹 Nand 與 Nor 邏輯閘後，讓學生們試試看能不能用其他種邏輯閘兜出某一種，或甚至用同一種邏輯閘兜出某一種。右方跟下頁的六張圖，分別展示出用 Nand 兜出 Not, And 與 Or (a ~ c 圖)，以及用 Nor 兜出 Not, And 與 Or (d ~ f 圖) 的做法。

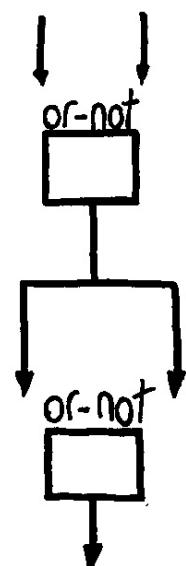




(d)



(e)



(f)

## 這個活動在說什麼？

近年來，在電腦網路進行的商務活動大幅的成長，其中不可或缺的是保證電子資金、機密交易還有所簽署具法律效力的文件交換的安全。密碼學的主題就是如何用安全且隱密的方式通信。幾十年前，資訊科學的研究人員發現一個違反直覺的結果：我們可以運用某些確保特定資訊公開的技術來達到將資訊保密的結果。這個結果就是我們會在活動 19 中所提到的「公開金鑰加密系統」，這也是現在廣泛用於交換訊息的主要加密方式。例如，你可能在瀏覽器中有看過 SSL (Secure Socket Layer, 安全通訊協定) 或 TLS (Transport Layer Security, 傳輸層安全協定) 的設定；這些系統都是基於公開金鑰系統，讓你的瀏覽器能夠與像是銀行這種網站建立安全的連線，就算有人在竊聽網路上的資料也沒有用。

不過密碼學不是單純保持資訊的機密那麼簡單。它也可以控制資訊，限制哪些部份可以被其他人找到；此外就是建立分隔兩地的人之間的可信任通訊管道。加密的規則或「協定」的發展已經讓一些看起來不可能的事情發生，像是無法偽造的數位簽章，還有告訴別人你的某項資訊，但是實際上不用真正把它顯露出來。透過電話來擲硬幣是一個比較簡單的比喻性質的問題，不過一開始看起來也像是不可能的任務。

當然，在真實的情況下，Alicia 與 Benito 不會真正自己設計一個電路，而是會找個電腦程式來跑。實際上兩個人大概也都不太會對電腦程式裡怎麼運作感興趣。但是兩個人肯定都要確保對方不會設法影響最後的決定，不管對方是否有那種技術能力或時間。

原則上，任何的爭議都應該透過中立的仲裁者來解決。這個仲裁者拿到電路，還有 Alicia 的原始二進位值，她傳送給 Benito 的輸出值，還有 Benito 所送回去的猜測結果。當這些資訊交換完畢，通通都會變成是公開資訊，所以參與雙方都必須同意這最終的結果。仲裁者要能將 Alicia 的原始輸入放進電路中，並檢查輸出是否如 Alicia 所說，最後決定這個結果是否正確無誤。不用說，會有一個清楚明白的程序來確認大家都照著規則走，不致發生爭議。如果是用真正的硬幣，讓 Alicia 來丟然後 Benito 隔著電話來猜，沒有任何一個仲裁者會同意接受！



這個活動中使用的電路很小，現實上不太實用，因為很容易列出所有的輸入與輸出結果，要作弊就變得很簡單。一個方法是把輸入變成 32 個位元，這樣可靠度就高了一點。不過即使是這樣，依然不保證無法作弊 -- 因為整個結果是由特定電路所產生。另一種方法是用活動 15 「旅遊小鎮」所介紹的單向函數。

一般會使用的方法是依靠找出一個很大的數字的因數。這個是一個已知的困難問題（雖然在下一個活動的結尾，我們會知道它並不是一個 NP 完全問題）。

我們很容易可以確定某個數是不是另一個數的因數，但是如果要把一個很大的數字做因數分解有時是很花時間的。

數位簽章就是基於類似概念。Alicia 將她的輸入值放進電路中計算出輸出；而輸出值事實上也進一步證實了輸入該值的人的確就是 Alicia -- 因為如果這個「電路」是適當的單向函數，那麼就沒有人能找出第二組輸入值會得到相同的輸出。換句話說，因為除了 Alicia 手上那一組值之外不會有第二個可能的輸入值，那就沒有人可以假冒 Alicia！當然，要做出一個數位簽章，通常需要更為複雜的協定，確保 Alicia 可以「簽署」某個訊息，而且別人可以透過相同的機制確認這個訊息就是由 Alicia 簽署的 -- Alicia 想賴也賴不掉。雖然數位簽章的協定複雜很多，但原則跟上面所說的「電路」是一樣的。

另一個應用是透過電話玩撲克牌 -- 沒有裁判可以來發牌與紀錄雙方手上的牌。每件事都要靠玩家自己來，只在遊戲結束有爭端的時候靠仲裁者來仲裁。拿實際上的例子，就好比商業合同的談判。明顯地，在遊戲進行中，玩家要保密自己的牌。但是他們必須誠實，不能偷雞假裝宣稱自己有 A！各家手上的牌可以在遊戲結束時攤開來看，並且確認大家出牌的手順。但是怎麼洗牌同時讓各家的牌能夠保密不被偷看就是問題了。令人驚訝的是，洗牌這件事其實可以用一個跟擲硬幣一樣的加密協定來完成。

加密協定在電子交易上是非常非常重要的，不管是確認記帳卡或信用卡持有人的身份，授權某個手機可以通話，或是確認某封電子郵件的發信人是否是本人等等。穩定可信賴的加密機制對電子商務的成功來說是最基本的要素。



## 延伸閱讀

Harel 的書 “Algorithmics” 討論了數位簽章與相關的加密協定。它也說明了如何透過電話來玩梭哈；這個想法最初是在 1981 年，在 D. A. Klarner 所編著的 “The Mathematical Gardner” 中，“Mental poker” 這一章所提出來的。Dorothy Denning 所著的 “Cryptography and data security” 是資訊科學領域中，關於密碼學的一本非常棒的教科書。Dewdney 的 “Turing Omnibus” 則有一個章節提到布林邏輯，並討論如何建立此活動中所用的電路。