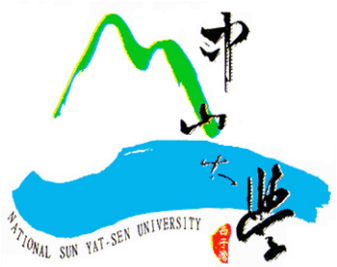
課程名稱：演算法設計與分析

Design and Analysis of Algorithms

Outline of This Lecture

1) Linear Programming

2) Mixed Strategy Game

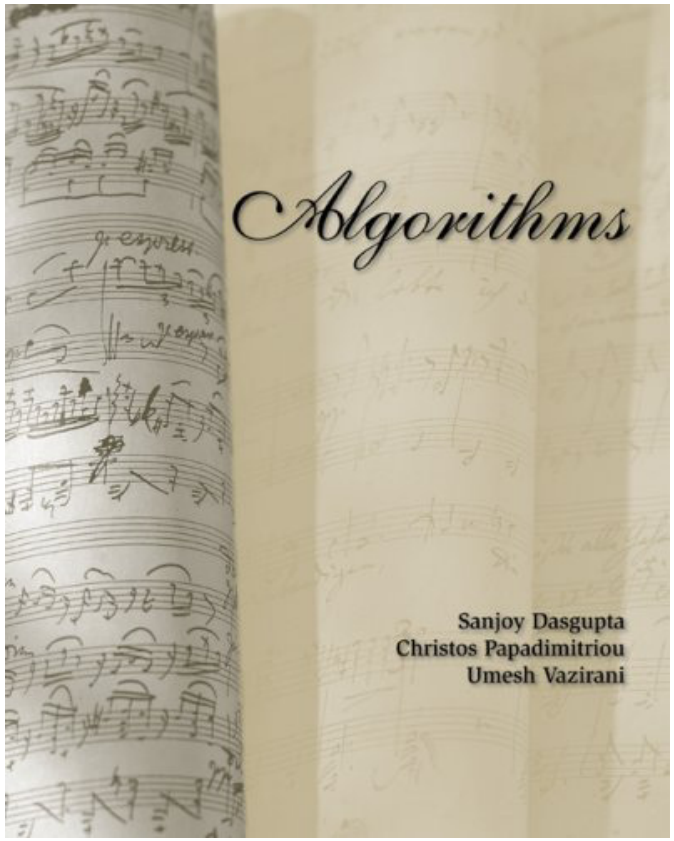


授課教師：周孜燦

國立中山大學電機系

聯絡方式：ztchou@ee.nsysu.edu.tw

Reference



本章內容參考 Sanjoy Dasgupta, Christos Papadimitriou, Umesh Vazirani, *Algorithms*, McGraw-Hill, 2006。理由：雖然課本有介紹 linear programming，但就這個章節而言，寫的不如這本書簡潔精彩

★ 投影片網站有提供電子檔

Linear Programming：Example

In a linear programming problem, we are given a set of variables, and we want to assign real values to them so as to
1) satisfy a set of linear equations/inequalities involving these variables
2) maximize or minimize a given linear objective function.

假設一家蛋糕公司，每天生產巧克力蛋糕和水果蛋糕。根據經驗，巧克力蛋糕每日需求量最多不超過 200 公斤，水果蛋糕每日需求量最多不超過 300 公斤。由於蛋糕師傅有限，每天最多只能做 400 公斤蛋糕。巧克力蛋糕一公斤價值美金 1 元，水果蛋糕一公斤價值美金 6 元，問每天應生產多少個巧克力蛋糕（x_1 公斤）及水果蛋糕（x_2 公斤），才能獲得最大的收益？ 這個問題可以表示成下列的 linear program

$$\begin{aligned} \text{Objective function} \quad & \max\ x_1 + 6x_2 \\ \text{Constraints} \quad & x_1 \le 200 \\ & x_2 \le 300 \\ & x_1 + x_2 \le 400 \\ & x_1, x_2 \ge 0 \end{aligned}$$

巧克力蛋糕

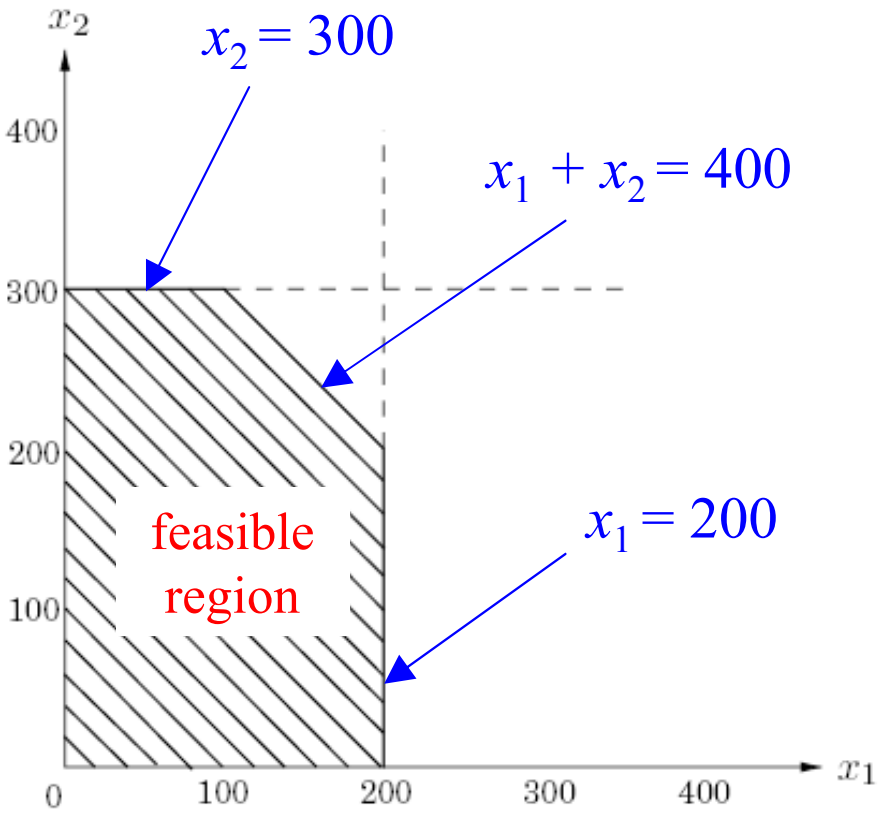
水果蛋糕

How to Solve the Profit Maximization Problem ?

Objective function $\max\ x_1 + 6x_2$

Constraints

$$x_1 \le 200$$
$$x_2 \le 300$$
$$x_1 + x_2 \le 400$$
$$x_1, x_2 \ge 0$$



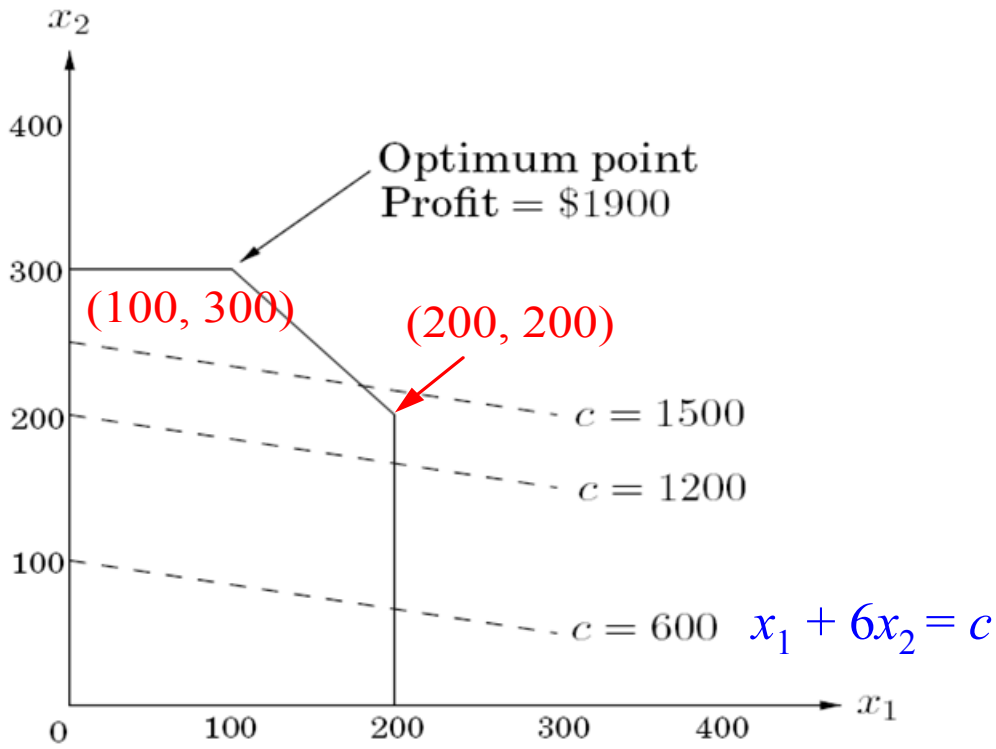
讓我們來看看如何解決上一頁的「收益最大化」問題。 如右圖，每一個不等式事實上都決定了一個「半平面」，因此滿足這些不等式的可行解（feasible solution）將落在這些半平面的交集裡頭，稱之爲 feasible region

Move the Contour Line to Find the Optimal Value

Objective function $\max x_1 + 6x_2$

Constraints

$$x_1 \le 200$$
$$x_2 \le 300$$
$$x_1 + x_2 \le 400$$
$$x_1, x_2 \ge 0$$



令 ***L*** 表示二維平面上 $x_1 + 6x_2 = c$ 這條線。*L* 和 feasible region 的交集便是可以讓收益 = *c* 的可行解（註：如果 *L* 和 feasible region 的交集是一條線，表示這條線上的所有點都可以讓 $x_1 + 6x_2$ 的值等於 *c*）。 我們注意到，當 *L* 這條線往往右上方移動時，*c* 的值愈來愈大。當 *L* 和 feasible region 交集在 (100, 300) 這個點的時候，$x_1 + 6x_2 = c$ 有最大值 = 1900

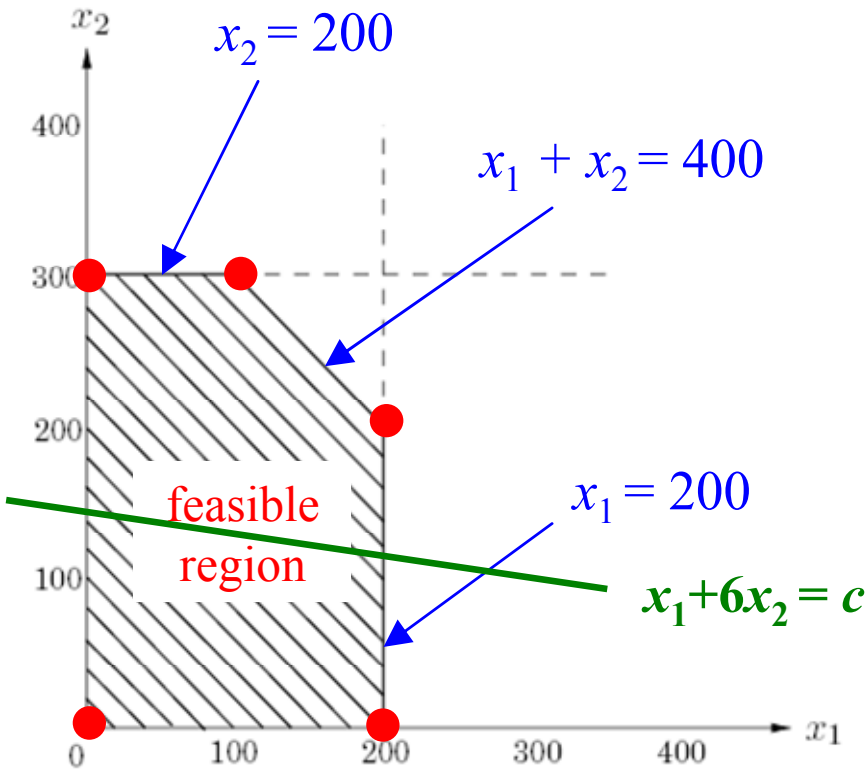
Optimal Solutions Are Vertices of Feasible Region

Objective function $\max\ x_1 + 6x_2$

Constraints

$$x_1 \le 200$$
$$x_2 \le 300$$
$$x_1 + x_2 \le 400$$
$$x_1, x_2 \ge 0$$

右圖的 feasible region 共有 5 個 vertices，分別爲
(0, 0), (200, 0), (200, 200), (100, 300), (0, 300)



我們把任意二個不等式的交點稱爲 **vertex**（**或者稱爲 extreme point**）。
從上面的推導，我們可發現最佳解 (x_1, x_2) 一定是落在 vertices 裡頭。
→ 所以只要檢驗所有的 vertices，看哪一個 vertex 可以使得目標值最大，那麼那一個 vertex 便是 optimal solution

Not All Linear Programs Have Bounded Solutions

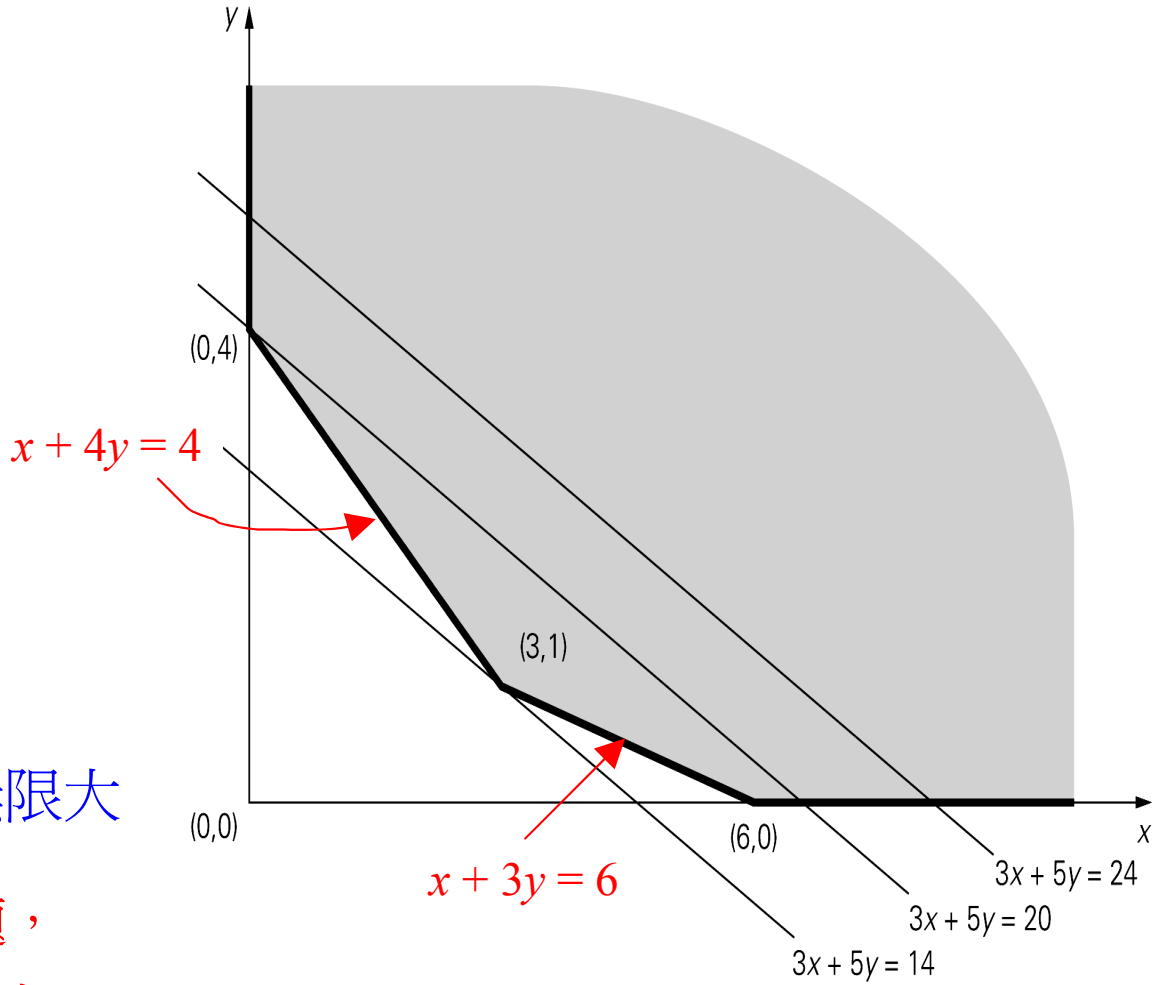
並不是所有的 linear program 都有 bounded solution，例如：

maximize $3x + 5y$

subject to $x + 4y \geq 4$

$x + 3y \geq 6$

$x \geq 0$ and $y \geq 0$



如右圖，此題不存在 bounded solution，因爲 x 和 y 的值可以無窮增加，導致 $3x + 5y$ 的值無限大

註：我們暫時先不考慮這樣的問題，等介紹 simplex algorithm 時，我們會學到如何判斷何時會發生這種情況

Add New Variables and Inequalities

假設蛋糕公司又增加新的產品 —「造型蛋糕」，每公斤售價 13美金，預計每日生產 x_3 公斤。由於蛋糕師傅並沒有增加，所以仍須滿足 $x_1 + x_2 + x_3 \leq 400$ 的限制。
此外，由於水果蛋糕和造型蛋糕共用相同的包裝機器，所以有 $x_2 + 3x_3 \leq 600$ 的限制。現在我們想知道這三種蛋糕每天應生產多少公斤才能獲得最大的收益？新的 linear program 如下：

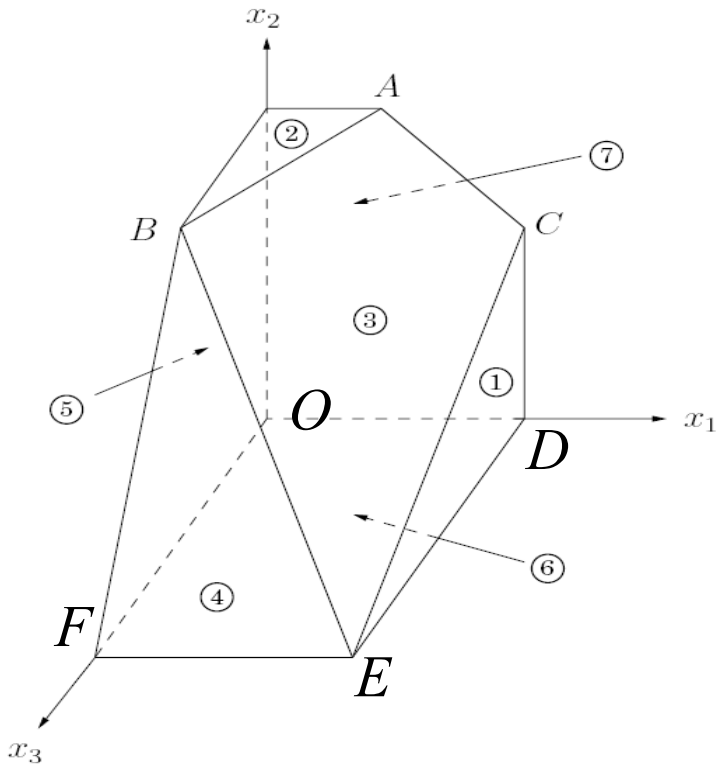
巧克力蛋糕

水果蛋糕

$$\begin{aligned} \max \quad & x_1 + 6x_2 + 13x_3 \\ & x_1 \leq 200 \\ & x_2 \leq 300 \\ x_1 + x_2 + x_3 & \leq 400 \\ x_2 + 3x_3 & \leq 600 \\ x_1, x_2, x_3 & \geq 0 \end{aligned}$$

造型蛋糕

Number of Vertices Increases Exponentially

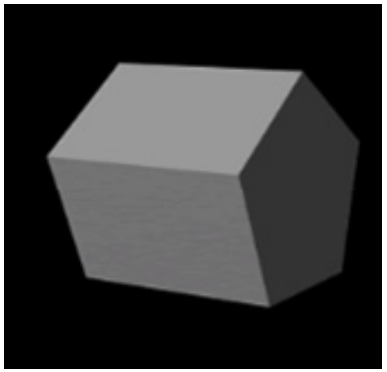


$$
\begin{aligned}
\max \quad & x_1 + 6x_2 + 13x_3 & \\
& x_1 \le 200 & ① \\
& x_2 \le 300 & ② \\
x_1 + x_2 + x_3 & \le 400 & ③ \\
x_2 + 3x_3 & \le 600 & ④ \\
& x_1 \ge 0 & ⑤ \\
& x_2 \ge 0 & ⑥ \\
& x_3 \ge 0 & ⑦
\end{aligned}
$$

現在，linear program 牽涉到 3 個變數（x_1, x_2, x_3），每一個不等式將決定出一個「半空間」。滿足這些不等式的可行解（feasible solution）將落在這些半空間的交集裡頭，稱之爲 feasible region。同樣地，我們只要檢驗這個 feasible region 的 vertices 便能求出最佳解。然而，我們注意到：**隨著變數和不等式的增加，vertices 總數暴增的非常快，呈現指數成長。這意味著若想要有效率地解決這個這問題，我們希望能避免檢查所有的 vertices。**

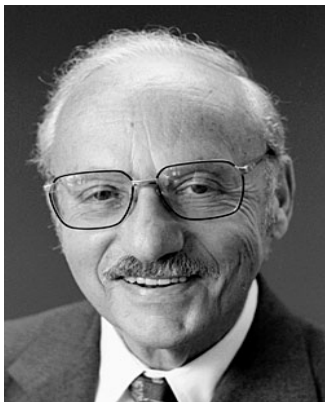
Simplex Algorithm

現在，我們要介紹由 George Dantzig 在 1947 年所開發出來的方法，稱之爲 simplex algorithm（又稱爲「單體法」，因爲由多個不等式所形成的 feasible region 爲一個 simple convex polytope「單純的多面體」）



convex polytope

concave polytope



George Dantzig

George Dantzig，1914 年出生，二次世界大戰期間，擔任美國空軍總部統計控制的戰鬥分析處主任。在此他開發出 linear programming 技術，負責解決物資補給的問題。現在，SIAM（Society for Industrial and Applied Mathematics）協會每三年頒發 George B. Dantzig Prize 給對於「最佳化」技術有所貢獻的人

Simplex Algorithm：Basic Idea

Step 1：

將最佳化的問題轉成求解最大化的「標準」型式（將不等式變成等式，方便代數運算）

Step 2：

尋找 initial feasible solution（簡寫成 fs，意即所有變數都 ≥ 0 的解）→ 相對右圖，尋找 feasible region 的起始 vertex（通常是原點）

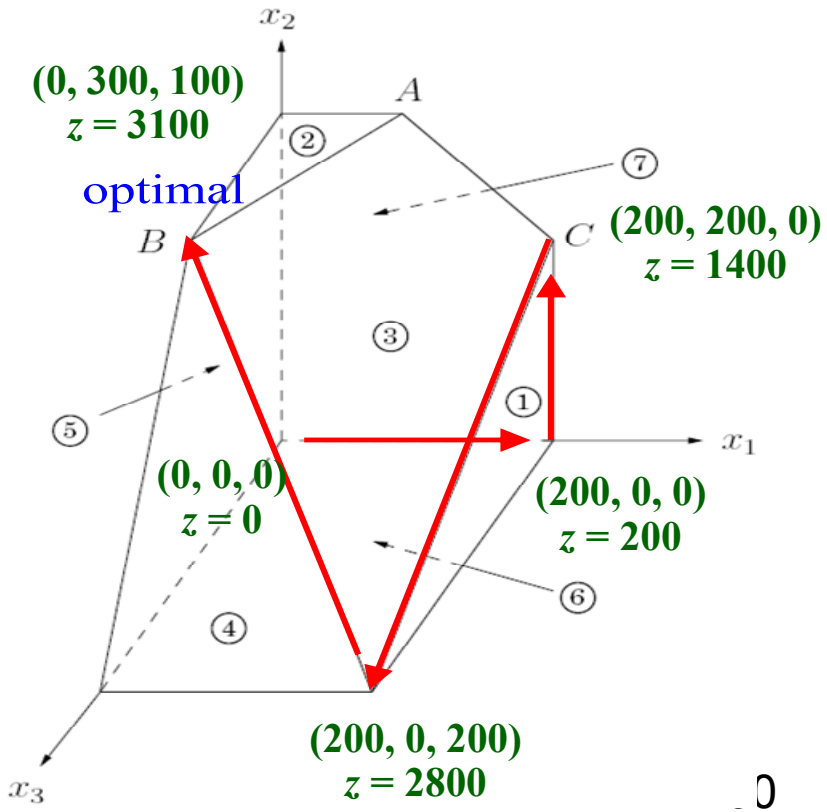
Step 3：

判斷目前這個 fs（vertex）是否爲最佳解？
如果是，程式終止。

Step 4：

如果目前這個 fs（vertex）不是最佳解，則尋找相鄰的 fs（vertex）可以讓 z 的值更大。
接著執行 Step 3。

$$\begin{aligned}
\max \quad & x_1 + 6x_2 + 13x_3 & \\
& x_1 \le 200 & ① \\
& x_2 \le 300 & ② \\
x_1 + x_2 + x_3 & \le 400 & ③ \\
x_2 + 3x_3 & \le 600 & ④ \\
& x_1 \ge 0 & ⑤ \\
& x_2 \ge 0 & ⑥ \\
& x_3 \ge 0 & ⑦
\end{aligned}$$



Convert Linear Program to Standard Form (1/4)

首先，我們要將 linear program 轉成下列的標準型式

$$\max \ z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$$

$$\text{subject to } \begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ \vdots \qquad\quad \vdots \qquad\qquad \vdots & \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_m &= b_m \\ x_i \geq 0 & \end{aligned}$$

} 這邊全部變成等式，以方便代數運算

目標式	$\min y = -x_1 + x_2 + x_3$
限制式	$x_1 + 2x_2 \leq 40$
	$x_1 + 3x_3 \geq 60$
變數範圍	$x_1 \geq -5$
	$x_2 \leq 20$
	x_3 範圍不限

轉變方法

變數變換

$\min y = -x_1 + x_2 + x_3$
$x_1 + 2x_2 \leq 40$
$x_1 + 3x_3 \geq 60$
$y_1 = 5 + x_1$ ，於是 $y_1 \geq 0$
$y_2 = 20 - x_2$ ，於是 $y_2 \geq 0$
令 $x_3 = y^+_3 - y^-_3$ 於是 $y^+_3 \geq 0$ 且 $y^-_3 \geq 0$

Convert Linear Program to Standard Form (2/4)

首先，我們要將 linear program 轉成下列的標準型式

$$\max\ z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$$

$$\text{subject to } \begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \qquad\quad \vdots \qquad\qquad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_m = b_m \end{cases}$$

這邊全部變成等式，以方便代數運算

$$x_i \geq 0$$

轉變方法

目標式	$\min y = -x_1 + x_2 + x_3$
限制式	$x_1 + 2x_2 \leq 40$
	$x_1 + 3x_3 \geq 60$
變數範圍	$y_1 = 5 + x_1$ ，於是 $y_1 \geq 0$
	$y_2 = 20 - x_2$ ，於是 $y_2 \geq 0$
	令 $x_3 = y^+_3 - y^-_3$ 於是 $y^+_3 \geq 0$ 且 $y^-_3 \geq 0$

將新變數代入目標式與限制式

$\min y = -y_1 - y_2 + y^+_3 - y^-_3 + 25$
$y_1 - 2y_2 \leq 75$
$y_1 + 3y^+_3 - 3y^-_3 \geq 65$
$y_1 = 5 + x_1$ ，於是 $y_1 \geq 0$
$y_2 = 20 - x_2$ ，於是 $y_2 \geq 0$
令 $x_3 = y^+_3 - y^-_3$ 於是 $y^+_3 \geq 0$ 且 $y^-_3 \geq 0$

Convert Linear Program to Standard Form (3/4)

首先，我們要將 linear program 轉成下列的標準型式

$$\max \; z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$$

$$\text{subject to } \begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ \vdots \quad\quad \vdots \quad\quad &\quad\quad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_m &= b_m \end{aligned}$$

這邊全部變成等式，以方便代數運算

$$x_i \geq 0$$

目標式	$\min y = -y_1 - y_2 + y^+_3 - y^-_3 + 25$
限制式	$y_1 - 2y_2 \leq 75$
	$y_1 + 3y^+_3 - 3y^-_3 \geq 65$
變數範圍	$y_1 \geq 0,\ y_2 \geq 0,$ $y^+_3 \geq 0,\ y^-_3 \geq 0$

轉變方法

引入 slack variable

引入 excess variable

$\min y = -y_1 - y_2 + y^+_3 - y^-_3 + 25$
$y_1 - 2y_2 + s_1 = 75$，其中 $s_1 \geq 0$
$y_1 + 3y^+_3 - 3y^-_3 - e_1 = 65$，其中 $e_1 \geq 0$
$y_1 \geq 0,\ y_2 \geq 0,$ $y^+_3 \geq 0,\ y^-_3 \geq 0$

Convert Linear Program to Standard Form (4/4)

首先，我們要將 linear program 轉成下列的標準型式

$$\max\ z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$$

$$\text{subject to } \begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \qquad\quad \vdots \qquad\qquad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_m = b_m \end{cases}$$

這邊全部變成等式，以方便代數運算

$$x_i \geq 0$$

目標式	$\min y = -y_1 - y_2 + y^+_3 - y^-_3 + 25$
限制式	$y_1 - 2y_2 + s_1 = 75$， 其中 $s_1 \geq 0$
	$y_1 + 3y^+_3 - 3y^-_3 - e_1 = 65$， 其中 $e_1 \geq 0$
變數範圍	$y_1 \geq 0,\ y_2 \geq 0,$ $y^+_3 \geq 0\ ,\ y^-_3 \geq 0$

min 轉成 max

轉變方法

$\max z = -y$
$y_1 - 2y_2 + s_1 = 75$， 其中 $s_1 \geq 0$
$y_1 + 3y^+_3 - 3y^-_3 - e_1 = 65$， 其中 $e_1 \geq 0$
$y_1 \geq 0,\ y_2 \geq 0,$ $y^+_3 \geq 0\ ,\ y^-_3 \geq 0$

Simplex：Find the First Feasible Solution

例如：我們想求解

$$\left.\begin{aligned}&\max z = 3x_1 + 5x_2\\&\text{subject to } x_1 + x_2 \le 4\\&\qquad\qquad x_1 + 3x_2 \le 6\\&x_1,\ x_2 \ge 0\end{aligned}\right\} \text{轉換成標準型式} \Rightarrow \left\{\begin{aligned}&\max z = 3x_1 + 5x_2\\&\text{subject to } x_1 + x_2 + s_1 = 4\\&\qquad\qquad x_1 + 3x_2 + s_2 = 6\\&x_1,\ x_2,\ s_1,\ s_2 \ge 0\end{aligned}\right.$$

現在，我們要解聯立方程式 $\begin{cases} x_1 + x_2 + s_1 = 4 \\ x_1 + 3x_2 + s_2 = 6 \end{cases}$

⇒ 有 4 個未知數，卻只有 2 個 equations，所以有無窮多組解。

此時我們可以令其中 2 個變數爲 0（因爲 0 是可行解），

這 2 個變數稱爲 non-basic variables（簡寫 NBV）

⇒ 注意：我們要求 NBV 必須爲 0。

剩下來的變數則稱爲 basic variables（簡寫成 BV）。

Basic variables 會有唯一解，例如：令 $NBV = \{x_1,\ x_2\}$ ，

那麼 $BV = \{s_1,\ s_2\}$ ，唯一解爲 $s_1 = 4,\ s_2 = 6$ 。

Simplex Tableau

$$\left.\begin{aligned}&\max z = 3x_1 + 5x_2\\&\text{subject to}\\&x_1 + x_2 + s_1 = 4\\&x_1 + 3x_2 + s_2 = 6\\&x_1,\ x_2,\ s_1,\ s_2 \ge 0\end{aligned}\right\} \Rightarrow \left.\begin{aligned}&z = 3x_1 + 5x_2\\&x_1 + x_2 + s_1 = 4\\&x_1 + 3x_2 + s_2 = 6\\&x_1,\ x_2,\ s_1,\ s_2 \ge 0\end{aligned}\right\} \Rightarrow \begin{aligned}&z - 3x_1 - 5x_2 = 0\\&x_1 + x_2 + s_1 = 4\\&x_1 + 3x_2 + s_2 = 6\end{aligned}$$

最右邊那個 linear program 可用下方的 simplex tableau 來表示

	z	x_1	x_2	s_1	s_2	等號右邊	BV
Row 0	1	−3	−5	0	0	0	
Row 1	0	1	1	1	0	4	$s_1 = 4$
Row 2	0	1	3	0	1	6	$s_2 = 6$
solution		0	0	4	6		

NBV的解永遠 = 0　　**BV的解 ≥ 0**

注意：我們保持 **Row 0**（目標式）裡頭 **BV** 的係數全部爲 **0**

→ 這樣才能觀察出 *z* 的值是否已達最佳解（不受 **BV** 的影響）

Whether the Current Solution Is Optimal ?

$$\max z = 3x_1 + 5x_2$$
$$\text{subject to}$$
$$x_1 + x_2 + s_1 = 4$$
$$x_1 + 3x_2 + s_2 = 6$$
$$x_1,\ x_2,\ s_1,\ s_2 \geq 0$$

$\Rightarrow$

	z	x_1	x_2	s_1	s_2	等號右邊	BV
Row 0	1	−3	−5	0	0	0	
Row 1	0	1	1	1	0	4	$s_1 = 4$
Row 2	0	1	3	0	1	6	$s_2 = 6$
solution		0	0	4	6		

NBV **BV**

現在，我們已經有了一個 feasible solution $(x_1, x_2, s_1, s_2) = (0, 0, 4, 6)$。我們想知道這個 solution 是不是最佳解？ 如果是的話，simplex algorithm 可以終止

讓我們增加某個 **NBV** 的值看看，看 z 的值是否會增加，如果會的話，那麼目前的解還不是最佳解。我們觀察到如果讓 NBV 裡頭 x_1 或 x_2 的值增加，那麼 $z = 3x_1+5x_2$ 的值就會增加，所以現在的 solution 還不是最佳解。

→結論：當 **Row 0** 裡頭 **NBV** 的係數都 $\geq$ **0** 時，程式可終止

Choose the Entering Variable

$$\max z = 3x_1 + 5x_2$$
$$\text{subject to}$$
$$x_1 + x_2 + s_1 = 4$$
$$x_1 + 3x_2 + s_2 = 6$$
$$x_1,\ x_2,\ s_1,\ s_2 \geq 0$$

$\Rightarrow$

	z	x_1	x_2	s_1	s_2	等號右邊	BV
Row 0	1	−3	−5	0	0	0	
Row 1	0	1	1	1	0	4	$s_1 = 4$
Row 2	0	1	3	0	1	6	$s_2 = 6$
solution		0	0	4	6		

NBV **BV**

如果讓 NBV 裡頭 x_1 或 x_2 的值增加，那麼 $z = 3x_1+5x_2$ 的值就會增加，但是我們應該增加 x_1 的值，還是增加 x_2 的值呢？ 觀察 z 的公式，若 x_1 增加 1，z 的值會增加 3；x_2 增加 1，z 的值會增加 5，所以我們選擇增加 x_2 的值。但我們要求 NBV 的值必須爲 0，所以我們必須讓 x_2 轉成 BV。此時我們稱 x_2 爲 entering variable（簡寫 EV）

→ 結論：挑選 **Row 0** 裡頭係數負的最多的 **NBV** 做爲 **entering variable**

Ratio Test：Choose the Leaving Variable

$$\max z = 3x_1 + 5x_2$$
$$\text{subject to}$$
$$x_1 + x_2 + s_1 = 4$$
$$x_1 + 3x_2 + s_2 = 6$$
$$x_1,\ x_2,\ s_1,\ s_2 \geq 0$$

$\Rightarrow$

	z	x_1	x_2	s_1	s_2	等號右邊	BV	ratio test
Row 0	1	−3	−5	0	0	0		
Row 1	0	1	1	1	0	4	$s_1 = 4$	4/1 = 4
Row 2	0	1	3	0	1	6	$s_2 = 6$	6/3 = 2
solution		0	0	4	6	**EV = x_2，LV = s_2**		

NBV **BV**

由於我們要求 NBV 的解必須爲 0，如果我們增加 x_2 的值，那麼必定要有一個 BV 的值變成 0（此一變數稱爲 leaving variable），轉成 NBV。

那我們要挑 s_1 還是 s_2 呢？ 觀察一下：因爲 $x_1 = 0$，所以我們有 $x_2 + s_1 = 4$ 且 $3x_2 + s_2 = 6$。又 $s_1 \geq 0$ 且 $s_2 \geq 0$，所以 $x_2 \leq 4$ 且 $3x_2 \leq 6$ (即 $x_2 \leq 2$)。 x_2 的值愈大愈好，但又必須滿足 $x_2 \leq \min\{4, 2\}$ 的限制。當 $x_2 = 2$ 時，s_2 的值爲 0，所以 s_2 適合當 leaving variable（也就是 s_2 轉成 NBV）。

結論：令 $\theta_i = \dfrac{\text{等號右邊的值}}{\text{第 } i \text{ 列 entering variable 係數}}$，令 $\delta = \arg\min_{1 \leq i \leq m} \{\theta_i \mid \theta_i > 0\}$

挑第 δ 列係數不爲 0 的 BV 做爲 leaving variable。第 δ 列稱爲 pivot row。

How Simplex Detects Unbounded Solution ?

$$\max z = 3x_1 + 5x_2$$
$$\text{subject to}$$
$$x_1 + x_2 + s_1 = 4$$
$$x_1 + 3x_2 + s_2 = 6$$
$$x_1,\ x_2,\ s_1,\ s_2 \geq 0$$

$\Rightarrow$

	z	x_1	x_2	s_1	s_2	等號右邊	BV	**ratio test**
Row 0	1	−3	−5	0	0	0		
Row 1	0	1	1	1	0	4	$s_1 = 4$	**4/1 = 4**
Row 2	0	1	3	0	1	6	$s_2 = 6$	**6/3 = 2**
solution		0	0	4	6	**EV = x_2，LV = s_2**		

NBV

我們剛才提到 x_2 必須滿足 $x_2 \leq \delta = \min\{4, 2\}$ 的限制式。但如果對每個 NBV 而言，$\delta < 0$；而我們又要求所有 $\boldsymbol{x_i \geq 0}$，此時發生矛盾的現象。這意味此一 linear program 沒有 bounded solution，因此 simplex algorithm 可以終止

[偵測 unbounded solution 方法] 令 $\theta_i = \dfrac{\text{等號右邊的值}}{\text{第 } i \text{ 列 entering variable 係數}}$.

若 $\theta_i < 0$, 則改令 $\theta_i = \infty$. 令 $\theta = \min\limits_{1 \leq i \leq m}\{\theta_i\}$. 若 $\theta = \infty$, 則該 linear programming 問題不存在 bounded solution。（證明參見課本第 872 頁）

Pivot：Generate the New Simplex Tableau (1/2)

$$\max z = 3x_1 + 5x_2$$
$$\text{subject to}$$
$$x_1 + x_2 + s_1 = 4$$
$$x_1 + 3x_2 + s_2 = 6$$
$$x_1,\ x_2,\ s_1,\ s_2 \geq 0$$

$\Rightarrow$

	z	x_1	x_2	s_1	s_2	等號右邊	BV	**ratio test**
Row 0	1	−3	−5	0	0	0		
Row 1	0	1	1	1	0	4	$s_1 = 4$	**4/1 = 4**
Row 2	0	1	3	0	1	6	$s_2 = 6$	**6/3 = 2**
solution		0	0	4	6	**EV = x_2，LV = s_2**		

現在，我們有了新的 BV（x_2 的加入），但要如何求出新的 BV 的解呢？這意味著原先的聯立方程式的解也必須跟著改變。我們採用高斯消去法（Gaussian elimination）：首先將 pivot row 裡頭的 EV（entering variable）的係數變成 1 → 將第二列全部係數乘以 1/3（這樣做不會改變第二個 equation 的等號關係。意即 $x_1+3x_2+s_2=6 \equiv x_1/3+x_2+s_2/3=2$ ）

	z	x_1	x_2	s_1	s_2	等號右邊
Row 0	1	−3	−5	0	0	0
Row 1	0	1	1	1	0	4
Row 2’	0	1/3	1	0	1/3	2

Pivot：Generate the New Simplex Tableau (2/2)

Row 0：$z - 3x_1 - 5x_2 = 0$

Row 1：$x_1 + x_2 + s_1 = 4$

Row 2'：$\frac{1}{3}x_1 + x_2 + \frac{1}{3}s_2 = 2$

	z	x_1	x_2	s_1	s_2	等號右邊
Row 0	1	−3	−5	0	0	0
Row 1	0	1	1	1	0	4
Row 2’	0	1/3	1	0	1/3	2

接著我們讓其餘的 rows 裡頭，所有新的 EV（意即 x_2）的係數全部變 0。

但要如何做才不會改變所有 equations 的等號關係呢？

新的 Row 1 = Row 1’ = Row 1 – Row2’ → Row 1’ = $2x_1/3 + s_1 - s_2/3 = 2$

新的 Row 0 = Row 0’ = Row 0 + 5 × Row2’ → Row 0’ = $z - 4x_1/3 + 5s_2/3 = 10$

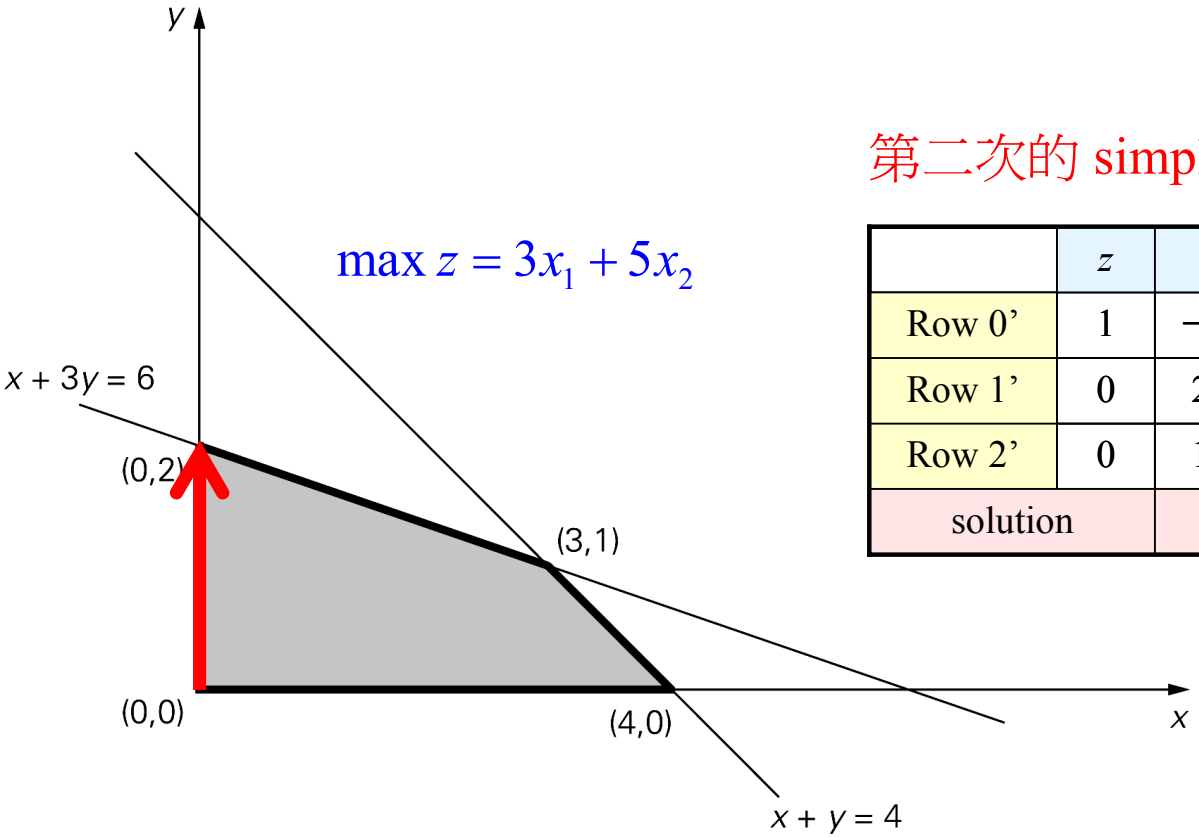
所以新的 simplex tableau 如下：（註：此時 BV 的解可以很容易地求出）

	z	x_1	x_2	s_1	s_2	等號右邊	BV
Row 0’	1	−4/3	0	0	5/3	10	
Row 1’	0	2/3	0	1	−1/3	2	$s_1 = 2$
Row 2’	0	1/3	1	0	1/3	2	$x_2 = 2$
solution		0	2	2	0		

BV

Verify the Current Result

從下圖，我們可發現 simplex algorithm 確實在尋找更好的 adjacent vertex，使得 z 的值增加（從 0 變成 10）



$$\max z = 3x_1 + 5x_2$$

第一次的 simplex tableau

	z	x_1	x_2	s_1	s_2	等號右邊	BV
Row 0	1	−3	−5	0	0	0	
Row 1	0	1	1	1	0	4	$s_1 = 4$
Row 2	0	1	3	0	1	6	$s_2 = 6$
solution		0	0	4	6		

第二次的 simplex tableau

	z	x_1	x_2	s_1	s_2	等號右邊	BV
Row 0’	1	−4/3	0	0	5/3	10	
Row 1’	0	2/3	0	1	−1/3	2	$s_1 = 2$
Row 2’	0	1/3	1	0	1/3	2	$x_2 = 2$
solution		0	2	2	0		

Repeat the Above-Mentioned Procedure

	z	x_1	x_2	s_1	s_2	等號右邊	BV
Row 0’	1	−4/3	0	0	5/3	10	
Row 1’	0	2/3	0	1	−1/3	2	$s_1 = 2$
Row 2’	0	1/3	1	0	1/3	2	$x_2 = 2$
solution		0	2	2	0		

$$\text{Row } 0' : z - \frac{4}{3}x_1 + \frac{5}{3}s_2 = 10$$

$$\text{Row } 1' : \frac{2}{3}x_1 + s_1 - \frac{2}{3}s_2 = 2$$

$$\text{Row } 2' : \frac{1}{3}x_1 + x_2 + \frac{1}{3}s_2 = 2$$

上表顯示的「第二次 simplex tableau」是否爲最佳解？

No！ 因爲**第 0 列還有係數爲負的 NBV**。

所以接著挑選 EV（x_1），執行 ratio test，挑選 LV（s_1），並執行高斯消去法，最後產生下表（同學們自己練習看看）

	z	x_1	x_2	s_1	s_2	等號右邊	BV
Row 0’’	1	0	0	2	1	14	
Row 1’’	0	1	0	3/2	−1/2	3	$x_1 = 3$
Row 2’’	0	0	1	−1/2	1/2	1	$x_2 = 1$
solution		3	1	0	0		

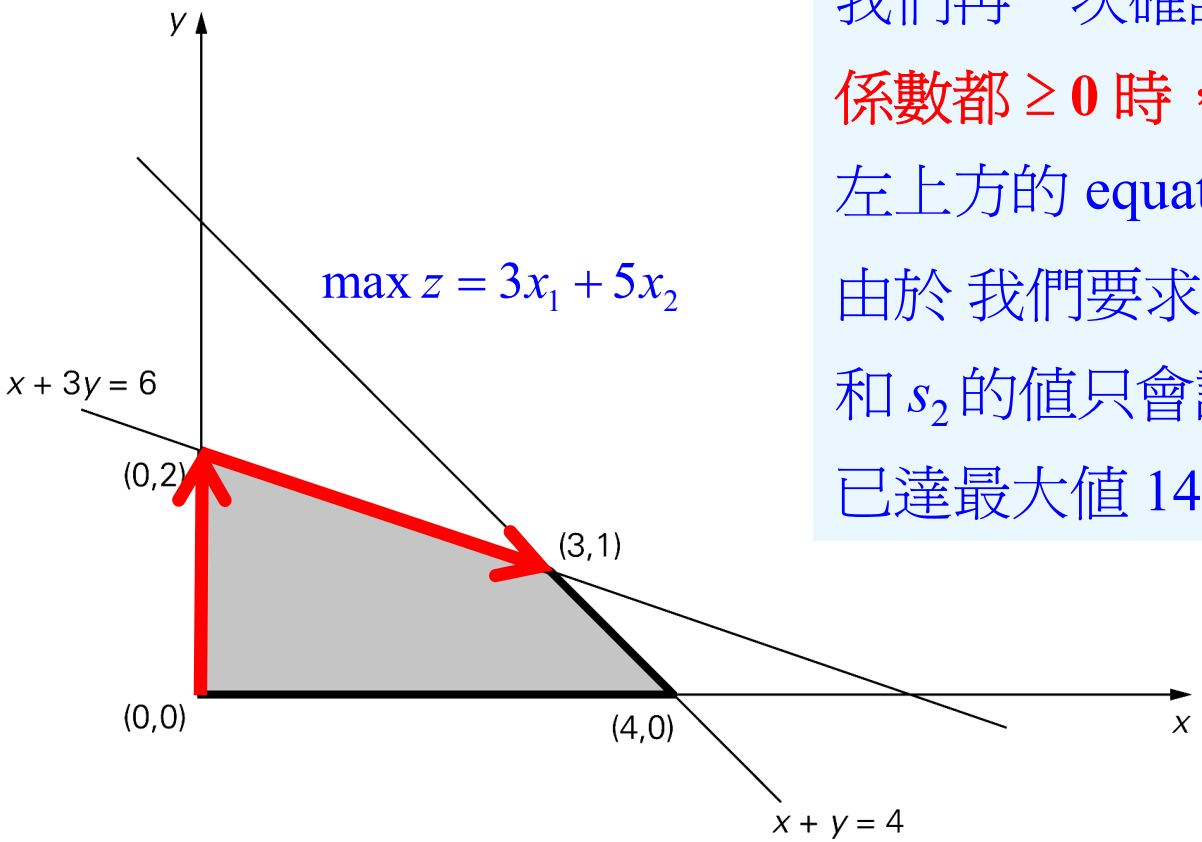
此表已產生最佳解，因爲第 0 列 NBV 的係數皆 **≥ 0**

Simplex Stops When Coefficients of NBVs > 0

Row 0" : $z + 2s_1 + s_2 = 14 \Rightarrow z = 14 - 2s_1 - s_2$

Row 1" : $x_1 + \frac{3}{2}s_1 - \frac{1}{2}s_2 = 3$

Row 2" : $x_2 - \frac{1}{2}s_1 + \frac{1}{2}s_2 = 1$



從左圖，我們可觀察到 simplex algorithm 確實沿著 adjacent vertex 找到最佳解。讓我們再一次確認：爲何**當第 0 列 NBV 的係數都 ≥ 0 時，表示已經找到最佳解**？

左上方的 equations 顯示，$z = 14 - 2s_1 - s_2$。

由於 我們要求 s_1 和 s_2 皆 ≥ 0，增加 NBV s_1 和 s_2 的值只會讓 z 的值更小，所以此時 z 已達最大值 14。

Constraints Combination as a Solution Certificate

$\max z = 3x_1 + 5x_2$

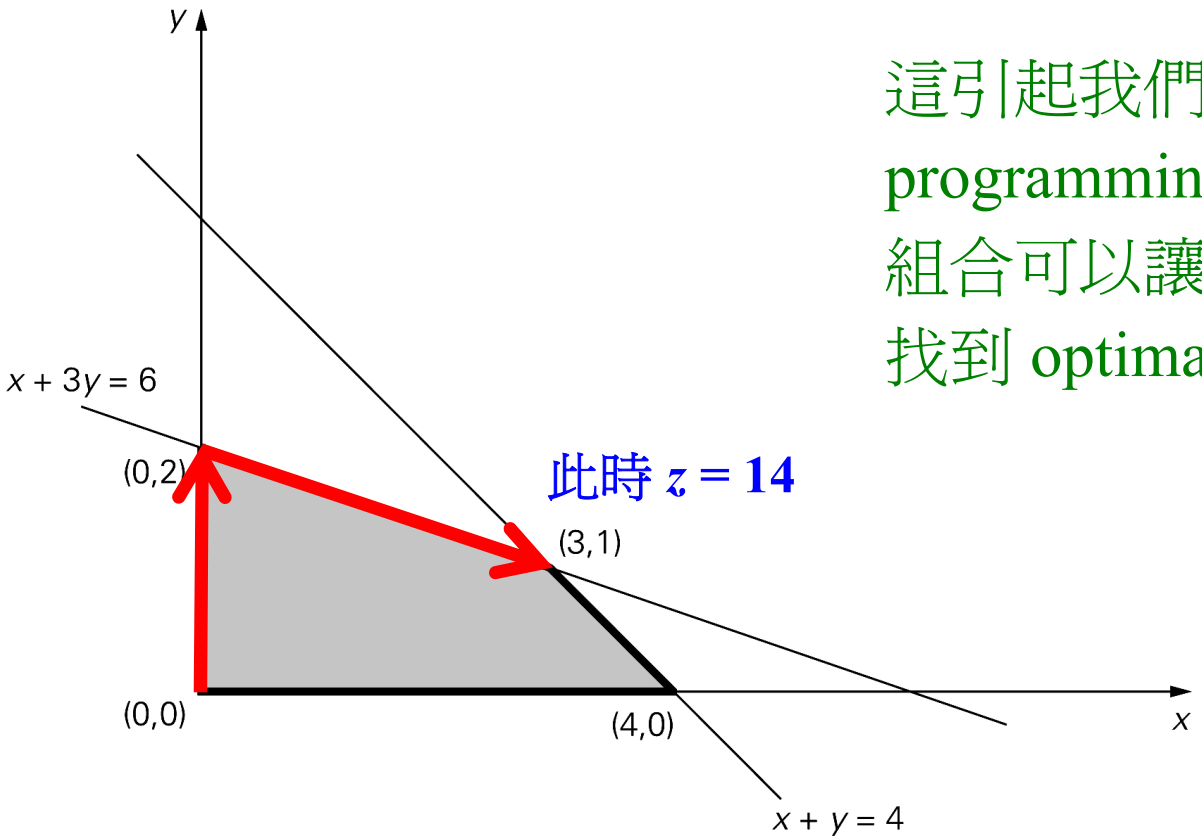
subject to $x_1 + x_2 \le 4$ .....(1)

$x_1 + 3x_2 \le 6$ .....(2)

$x_1,\ x_2 \ge 0$

$2\times(1) + 1\times(2) \Rightarrow 3x_1 + 5x_2 \le 14$

又再一次驗證 z 的最大值確實是 14



這引起我們的好奇，是否所有 linear programming 的 constraints 的（某種）組合可以讓我們用來檢驗是否已經找到 optimal solution？

Primal Problem and Dual Problem (1/2)

$z^* = \max\ z = \max\ 3x_1 + 5x_2$

subject to $x_1 + x_2 \le 4$ .....(1)

$x_1 + 3x_2 \le 6$ .....(2)

$x_1,\ x_2 \ge 0$

因爲(6),(7) 是 ≥
所以 $z \le w$
且 $z^* = w^*$

$w^* = \min w = \min\ 4y_1 + 6y_2$

subject to $y_1 + y_2 \ge 3$ .....(6)

$y_1 + 3x_2 \ge 5$ .....(7)

$y_1,\ y_2 \ge 0$

把(1)和(2) 左右二邊分別同時乘上 y_1 和 y_2

（注意：我們要求 $y_1, y_2 \ge 0$, 以確保不等號方向沒有改變）

$y_1(x_1 + x_2) \le 4y_1$ .....(3), $y_2(x_1 + 3x_2) \le 6y_2$ .....(4)

$(3)+(4) \Rightarrow y_1(x_1 + x_2) + y_2(x_1 + 3x_2) \le 4y_1 + 6y_2$

$\Rightarrow (y_1 + y_2)x_1 + (y_1 + 3y_2)x_2 \le 4y_1 + 6y_2$ .....(5)

爲了確保 (5) 可以用來檢驗 z^* 的值是否已經求出，我們希望

$y_1 + y_2 = 3$.....(6), $y_1 + 3y_2 = 5$.....(7), 且 $4y_1 + 6y_2$ 的值愈小愈好

因爲 $z = 3x_1 + 5x_2 \le 4y_1 + 6y_2$, 所以當 $z = 4y_1 + 6y_2$ 時, 表示找到 z 的最大值

$\Rightarrow$ 把 (6) 改成 $y_1 + y_2 \ge 3$, 把 (7) 變成 $y_1 + 3y_2 \ge 3$, 我們有右邊的線性規劃

註：如果把 (6),(7) 改成 $y_1 + y_2 \le 3$, $y_1 + 3y_2 \le 3$, 那麼 w 的最小值爲 $-\infty$.

Primal Problem and Dual Problem (2/2)

$z^* = \max\ z = \max\ 3x_1 + 5x_2$

subject to $x_1 + x_2 \le 4$ .....(1)

$x_1 + 3x_2 \le 6$ .....(2)

$x_1,\ x_2 \ge 0$

稱爲 primal linear programming problem

對照左右二邊
顯然 $z \le w$
且 $z^* = w^*$

$w^* = \min w = \min\ 4y_1 + 6y_2$

subject to $y_1 + y_2 \ge 3$ .....(6)

$y_1 + 3x_2 \ge 5$ .....(7)

$y_1,\ y_2 \ge 0$

稱爲 dual linear programming problem

一般式

Primal LP:

$$\max\ c_1x_1 + \cdots + c_nx_n$$
$$a_{i1}x_1 + \cdots + a_{in}x_n \le b_i \quad \text{for } i \in I$$
$$a_{i1}x_1 + \cdots + a_{in}x_n = b_i \quad \text{for } i \in E$$
$$x_j \ge 0$$

Dual LP:

$$\min\ b_1y_1 + \cdots + b_my_m$$
$$a_{1j}y_1 + \cdots + a_{mj}y_m \ge c_j \quad \text{for } j \in N$$
$$a_{1j}y_1 + \cdots + a_{mj}y_m = c_j \quad \text{for } j \notin N$$
$$y_i \ge 0$$

Duality Theorem：不同問題，相同目標

$z^* = \max\ z = \max\ 3x_1 + 5x_2$

subject to $x_1 + x_2 \le 4$ .....(1)

$x_1 + 3x_2 \le 6$ .....(2)

$x_1,\ x_2 \ge 0$

稱爲 primal linear programming problem

對照左右二邊
顯然 $z \le w$
且 $z^* = w^*$

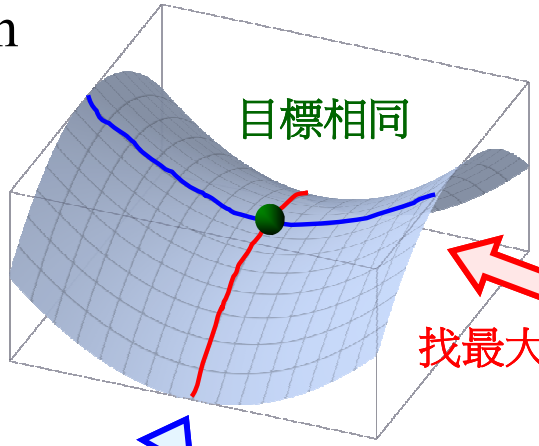
$w^* = \min w = \min\ 4y_1 + 6y_2$

subject to $y_1 + y_2 \ge 3$ .....(6)

$y_1 + 3x_2 \ge 5$ .....(7)

$y_1,\ y_2 \ge 0$

稱爲 dual linear programming problem



找最大值

primal LP problem

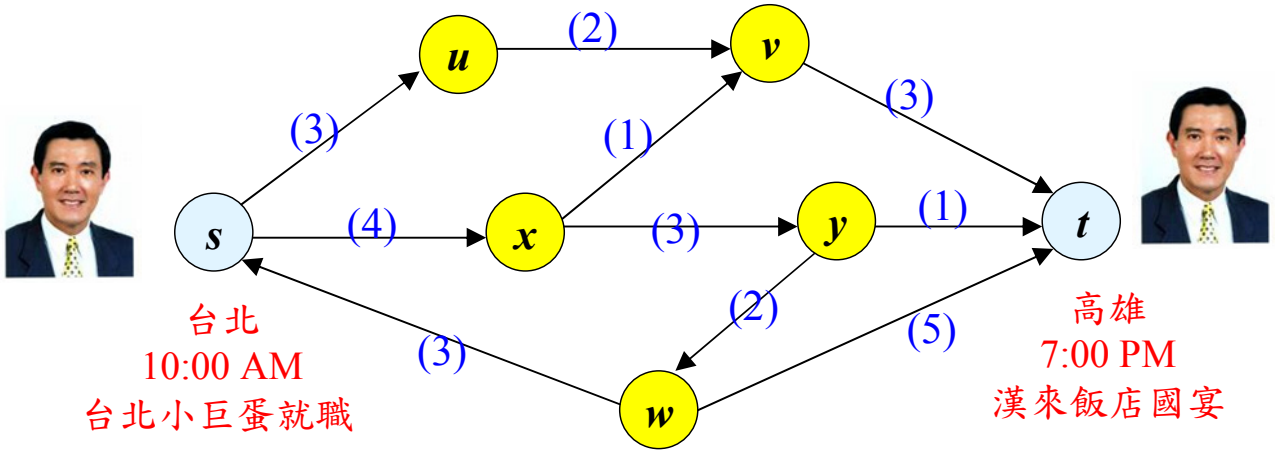
找最小值

dual LP problem

這二個問題等價，因爲目標相同

Duality Theorem：Primal LP 的最佳解 z^* = Dual LP 的最佳解 w^*

Maximum Flow Problem Is an LP Problem



事實上，maximum flow problem 也能 model 成 linear programming 的問題 。

令變數 $x_{u\to v}$ 表示 edge (u, v) 上的 flow ，$c_{u\to v}$ 表示 edge (u, v) 上的 capacity

那麼對應 maximum flow problem 的 linear program 如下：

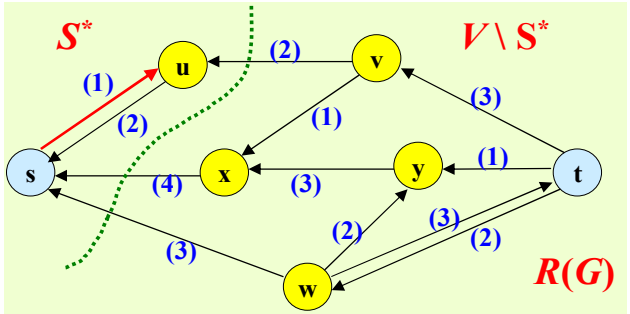
maximize $f(G) = \sum_{u\in N(s^{\rightarrow})} x_{s\to u} - \sum_{v\in N(s^{\leftarrow})} x_{v\to s}$

subject to

capacity contraint： $0 \le x_{u\to v} \le c_{u\to v}$, for all edge (u,v)

conservation law： $\sum_{v\in N(u^{\rightarrow})} x_{u\to v} = \sum_{v\in N(u^{\leftarrow})} x_{v\to u}$, for all $u \in V \setminus \{s,t\}$

Primal-Dual Algorithms

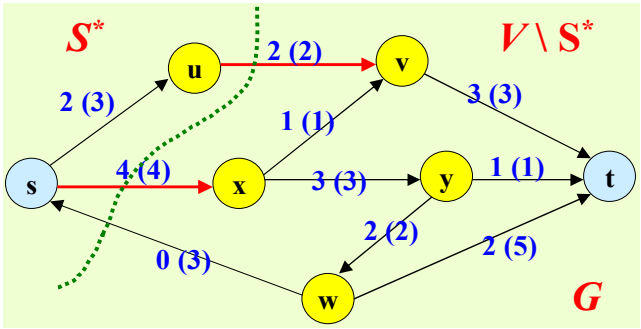


轉換

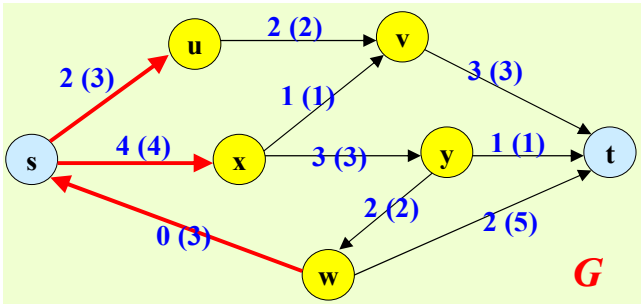
$R(G)$ 裡頭是否存在 augmenting path ?

否

對偶問題



必然是 max flow



原始問題 是否爲 max flow ?

找到最小 capacity 的 cut

$$S^* = \left\{ u \,\middle|\, \begin{array}{l} u \text{ is reachable from } s \text{ in } R(G), \\ \text{which has no augmenting paths} \end{array} \right\}$$

藉由判斷「是否已經找到 **dual LP** 的解」來判斷「是否已經找到 **primal LP** 的解」的演算法稱爲 **primal-dual algorithm**。

範例： 如何判斷目前的 **flow**是否已經是最大值？ 當 **residual network** 裡頭不存在 **augment path** → 此時必然存在一個 **cut**，其 **capacity** 值最小，等於 **max flow** 的值 → 利用 **min cut (dual LP)** 來判斷是否已經找到 **max flow (primal LP)**

Two-Phase Simplex Algorithm：Example

現在，讓我們看另一個例子

$$\left.\begin{aligned} &\min w = 2x_1 + 4x_2 \\ &0.3x_1 + 0.1x_2 \le 2.7 \\ &0.5x_1 + 0.5x_2 = 6 \\ &0.6x_1 + 0.4x_2 \ge 6 \\ &x_1,\ x_2 \ge 0 \end{aligned}\right\} \quad \text{轉成標準型式} \Rightarrow \quad \left\{\begin{aligned} &\max z = -2x_1 - 4x_2 \\ &0.3x_1 + 0.1x_2 + s_1 = 2.7 \ \text{.....(1)} \\ &0.5x_1 + 0.5x_2 = 6 \ \text{.....(2)} \\ &0.6x_1 + 0.4x_2 - e_1 = 6 \ \text{.....(3)} \\ &x_1,\ x_2,\ s_1,\ e_1 \ge 0 \end{aligned}\right.$$

按之前的範例，我們令 initial solution $(x_1,\ x_2) = (0,\ 0)$ 爲 NBV。

從 (1) 可得 $s_1 = 2.7$ 。從 (3) 可得 $e_1 = -6$ ，但 e_1 並沒有 ≥ 0 。

而且將 $x_1 = 0,\ x_2 = 0$ 帶入 (2) ，等號並不成立。

怎麼回事？ $\Rightarrow$ 如果推導過程正確，卻得到錯誤答案，

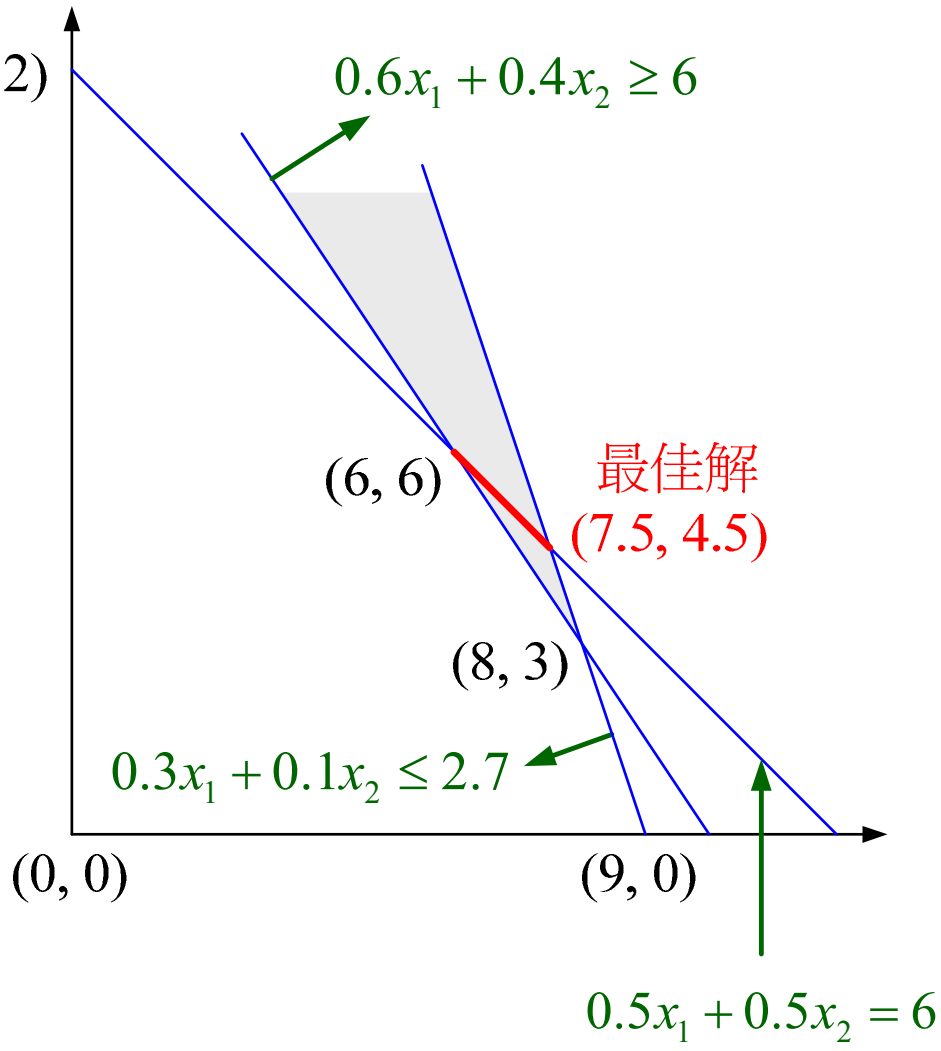
那便是假設錯誤。觀察原本的限制式， $(x_1,\ x_2) = (0,\ 0)$ 並不是可行解

Notice：(0, 0) Is Not in the Feasible Region

右圖顯示下列 linear program 的 feasible region（只有紅色的那個線段）

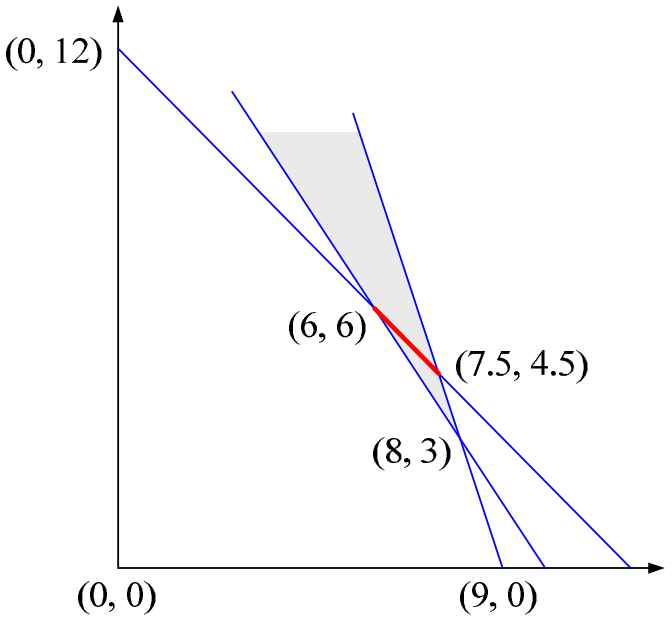
$$\begin{cases} \min w = 2x_1 + 4x_2 \\ 0.3x_1 + 0.1x_2 \le 2.7 \\ 0.5x_1 + 0.5x_2 = 6 \\ 0.6x_1 + 0.4x_2 \ge 6 \\ x_1,\ x_2 \ge 0 \end{cases}$$

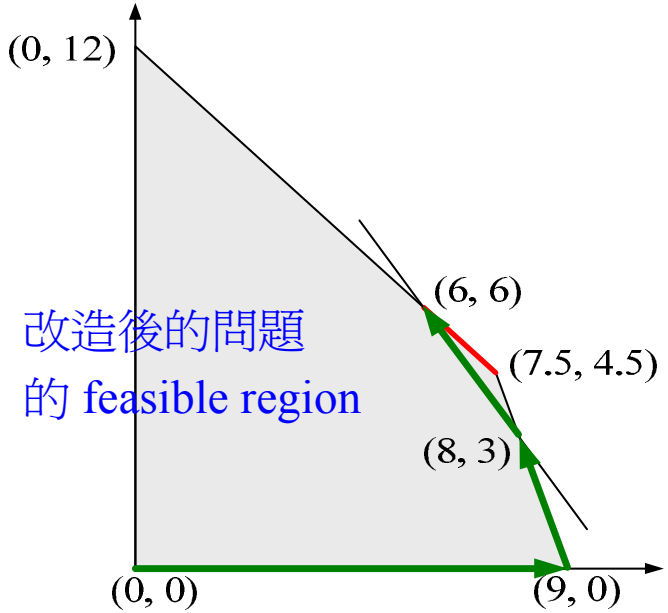
我們發覺，原點 (0, 0) 並沒有落在 feasible region 裡頭

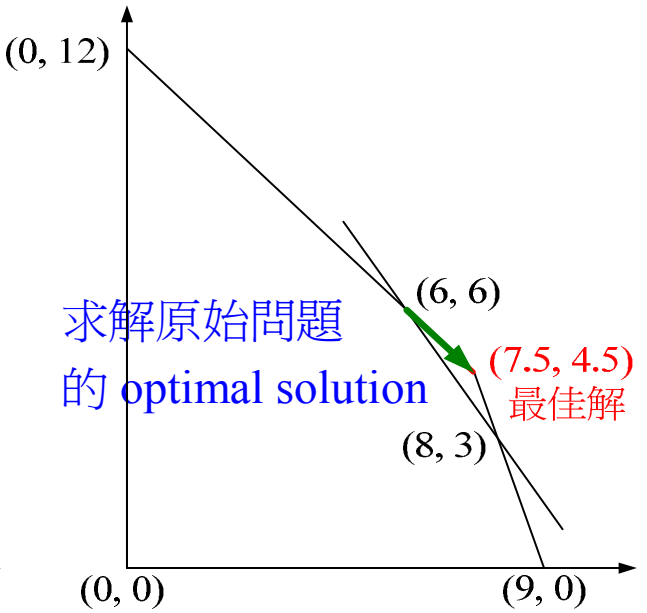


Two-Phase Simplex：Basic Idea

要執行 simplex algorithm，必須要有一個起始的 feasible solution，該怎辦呢？







檢查原點（NBV）是否有滿足所有的限制式

如果沒有，則對原始問題添加「人工變數」，並使用 simplex algorithm 找到原始問題的其中一個 feasible solution

有了 feasible solution 之後，我們就能針對原始問題使用 simplex algorithm 找到 optimal solution

First-Phase Simplex：Add Artificial Variables

$$\begin{cases} \min w = 2x_1 + 4x_2 \\ 0.3x_1 + 0.1x_2 \le 2.7 \\ 0.5x_1 + 0.5x_2 = 6 \\ 0.6x_1 + 0.4x_2 \ge 6 \\ x_1,\ x_2 \ge 0 \end{cases}$$

轉成標準型式

$\Rightarrow$

$$\begin{cases} \max z = -2x_1 - 4x_2 \\ 0.3x_1 + 0.1x_2 + s_1 = 2.7 \\ 0.5x_1 + 0.5x_2 = 6 \\ 0.6x_1 + 0.4x_2 - e_1 = 6 \\ x_1,\ x_2,\ s_1,\ e_1 \ge 0 \end{cases}$$

$$\begin{cases} \max z = -2x_1 - 4x_2 \\ 0.3x_1 + 0.1x_2 + s_1 = 2.7 \\ 0.5x_1 + 0.5x_2 = 6 \\ 0.6x_1 + 0.4x_2 - e_1 = 6 \\ x_1,\ x_2,\ s_1,\ e_1 \ge 0 \end{cases}$$

由於原點不是 feasible solution
所以每一列都添加人工變數。
最後，更改目標式可得 $\Rightarrow$

$$\begin{cases} \min w_0 = a_1 + a_2 + a_3 \\ 0.3x_1 + 0.1x_2 + s_1 + a_1 = 2.7 \\ 0.5x_1 + 0.5x_2 + a_2 = 6 \\ 0.6x_1 + 0.4x_2 - e_1 + a_3 = 6 \\ x_1,\ x_2,\ s_1,\ e_1,\ a_1,\ a_2,\ a_3 \ge 0 \end{cases}$$

如果最後 min w_0 的最小值爲 0（這樣才不會改變原本等號的關係），表示原本 max z 的問題有解。此時我們可以找到 max z 的一個 feasible solution

如果最後 min w_0 的最小值不爲 0，表示原本 max z 的問題無解。程式終止

First-Phase Simplex：Row 0 Loses Information

現在，我們要使用 simplex algorithm 來求解

$$\left.\begin{aligned} &\min w_0 = a_1 + a_2 + a_3 \\ &0.3x_1 + 0.1x_2 + s_1 + a_1 = 2.7 \\ &0.5x_1 + 0.5x_2 + a_2 = 6 \\ &0.6x_1 + 0.4x_2 - e_1 + a_3 = 6 \\ &x_1, x_2, s_1, e_1, a_1, a_2, a_3 \geq 0 \end{aligned}\right\} \text{轉成} \Rightarrow \left\{\begin{aligned} &\max z_0 = -(a_1 + a_2 + a_3) \\ &0.3x_1 + 0.1x_2 + s_1 + a_1 = 2.7 \\ &0.5x_1 + 0.5x_2 + a_2 = 6 \\ &0.6x_1 + 0.4x_2 - e_1 + a_3 = 6 \\ &x_1, x_2, s_1, e_1, a_1, a_2, a_3 \geq 0 \end{aligned}\right.$$

右邊那個 linear program 可用下方的 simplex tableau 來表示

	z_0	x_1	x_2	s_1	e_1	a_1	a_2	a_3	等號右邊	BV
Row 0	1	0	0	0	0	1	1	1	0	
Row 1	0	0.3	0.1	1	0	1	0	0	2.7	$a_1 = 2.7$
Row 2	0	0.5	0.5	0	0	0	1	0	6	$a_2 = 6$
Row 3	0	0.6	0.4	0	−1	0	0	1	6	$a_3 = 6$
solution		0	0	0	0	2.7	6	6		

上表所顯示的「simplex tableau」是否爲最佳解？

首先，**Row 0 裡頭 BV 的係數都不是 0**

→ 我們必須設法使 **Row 0 裡頭 BV 的係數全爲 0**

First-Phase Simplex：Modify Row 0

	z_0	x_1	x_2	s_1	e_1	a_1	a_2	a_3	等號右邊	BV
Row 0	1	0	0	0	0	1	1	1	0	
Row 1	0	0.3	0.1	1	0	1	0	0	2.7	$a_1 = 2.7$
Row 2	0	0.5	0.5	0	0	0	1	0	6	$a_2 = 6$
Row 3	0	0.6	0.4	0	−1	0	0	1	6	$a_3 = 6$
solution		0	0	0	0	2.7	6	6		

爲什麼 BV 的係數不爲 0 呢？ 這是因爲我們將目標式從 max z 改成 max z_0 的緣故

→ 要如何使得目標式的 BV 係數全爲 0，而又不會改變 equations 的等號關係呢？

利用代數運算，新的 Row 0 = Row 0' = Row 0 − (Row 1 + Row 2 + Row 3)

	z_0	x_1	x_2	s_1	e_1	a_1	a_2	a_3	等號右邊	BV	**ratio test**
Row 0	1	−1.4	−1	−1	1	**0**	**0**	**0**	−14.7		
Row 1	0	0.3	0.1	1	0	1	0	0	2.7	$a_1 = 2.7$	**2.7/0.3=9**
Row 2	0	0.5	0.5	0	0	0	1	0	6	$a_2 = 6$	**6/0.5=12**
Row 3	0	0.6	0.4	0	−1	0	0	1	6	$a_3 = 6$	**6/0.6=10**
solution		0	0	0	0	2.7	6	6	**EV = x_1，LV = a_1**		

Second-Phase Simplex: Change Objective Function

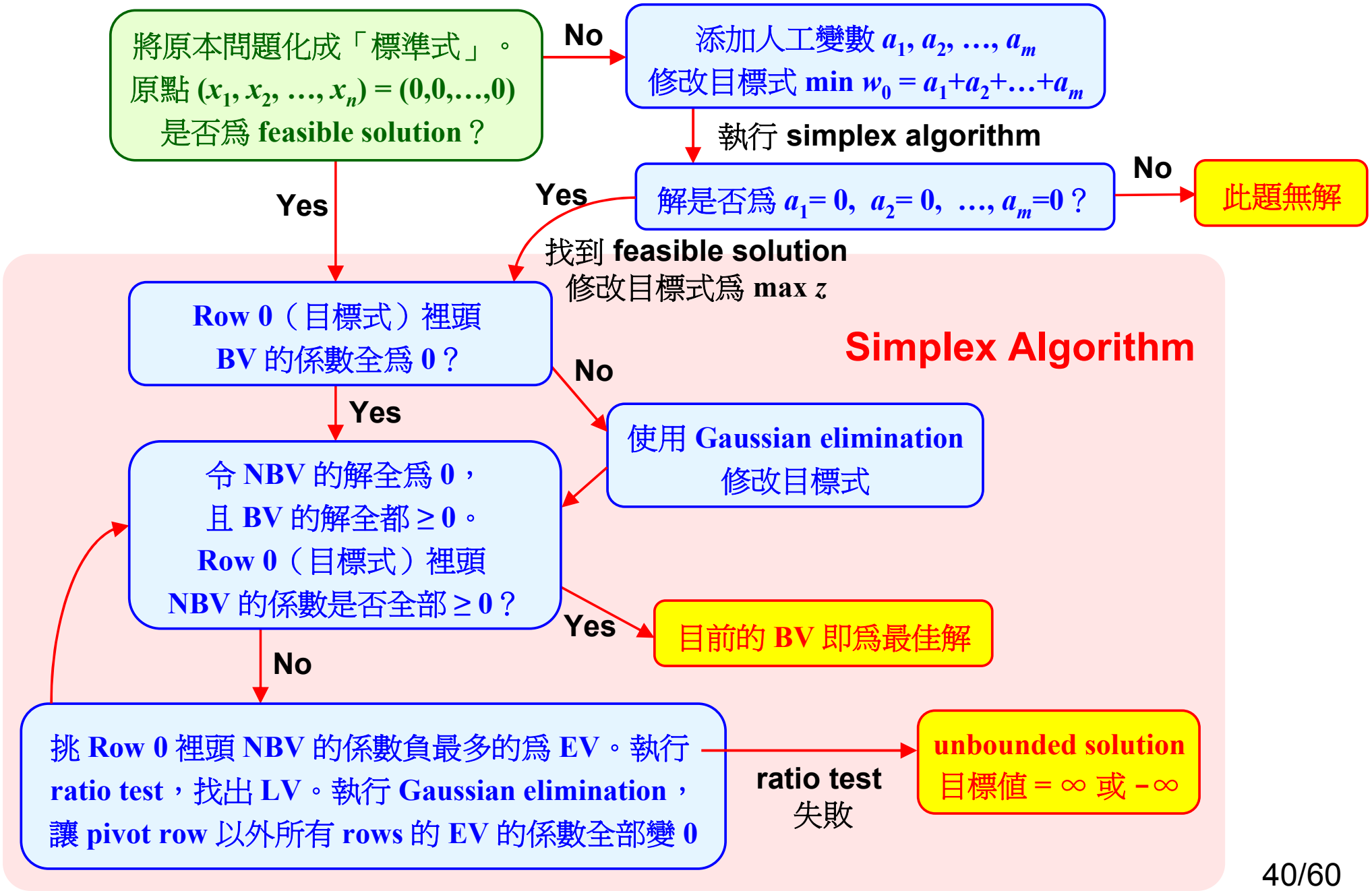
	z_0	x_1	x_2	s_1	e_1	a_1	a_2	a_3	等號右邊	BV
Row 0*	1	0	0	0	0	1	1	1	0	
Row 1*	0	1	0	0	−5	0	−4	5	6	$x_1 = 6$
Row 2*	0	0	0	1	1	1	3/5	−1	0.3	$s_1 = 0.3$
Row 3*	0	0	1	0	5	5	6	5	6	$x_2 = 6$
solution		6	6	0.3	0	0	0	0		

上表爲第一階段 simplex algorithm 的執行結果，最後 z_0 的值爲 0，且 $a_1 = a_2 = a_3 = 0$，所以原本 max z 的 feasible solution $(x_1, x_2, s_1, e_1) = (6, 6, 0.3, 0)$ 就求出來了。接著要如何利用上表求出 max z 呢？ 由於人工變數的值確實爲 0，所以 a_1、a_2、a_3 所在的 columns 也就不需要了。此外，由於 max z_0 和 max z 只是目標式不同，所以我們只要更改 Row 0* 即可。此時可得到下表。再利用 simplex algorithm 即可求出 max z

	z	x_1	x_2	s_1	e_1	等號右邊	BV
Row 0	1	2	4	0	0	0	
Row 1	0	1	0	0	−5	6	$x_1 = 6$
Row 2	0	0	0	1	1	0.3	$s_1 = 0.3$
Row 3	0	0	1	0	5	6	$x_2 = 6$
solution		6	6	0.3	0		

注意：**Row 0 裡頭 BV 係數不爲 0 所以仍須修改目標式**

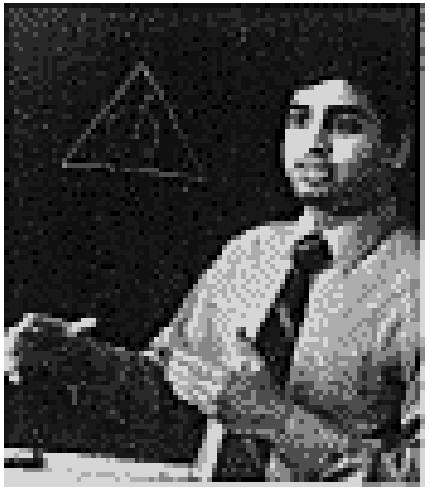
Flow Chart of Simplex Algorithm



Running Time Analysis

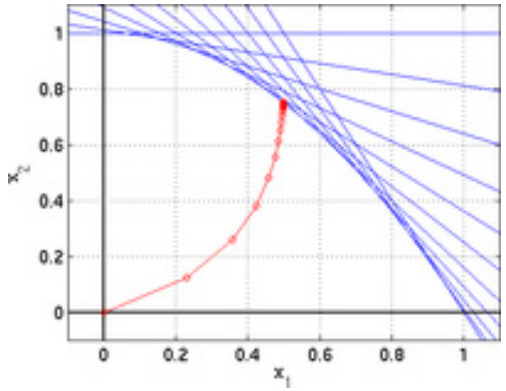
$$\begin{aligned}
\max \quad & x_1 + 6x_2 + 13x_3 \\
& x_1 \le 200 && ① \\
& x_2 \le 300 && ② \\
& x_1 + x_2 + x_3 \le 400 && ③ \\
& x_2 + 3x_3 \le 600 && ④ \\
& x_1 \ge 0 && ⑤ \\
& x_2 \ge 0 && ⑥ \\
& x_3 \ge 0 && ⑦
\end{aligned}$$

$m = 7$



Narendra Karmarkar，27 歲那年在 AT&T 工作時，發明 interior-point method

interior-point method



給定 n 個變數和 $m > n$ 個不等式，linear program 的 vertices 總數達 $C(m, n)$。Worst case 發生在 $n = m/2$。此時

$$C(m, m/2) = \frac{(m/2+1) \times (m/2+2) \times \cdots \times (m/2+m)}{1 \times 2 \times \cdots \times (m/2)} \ge 1.5^m$$

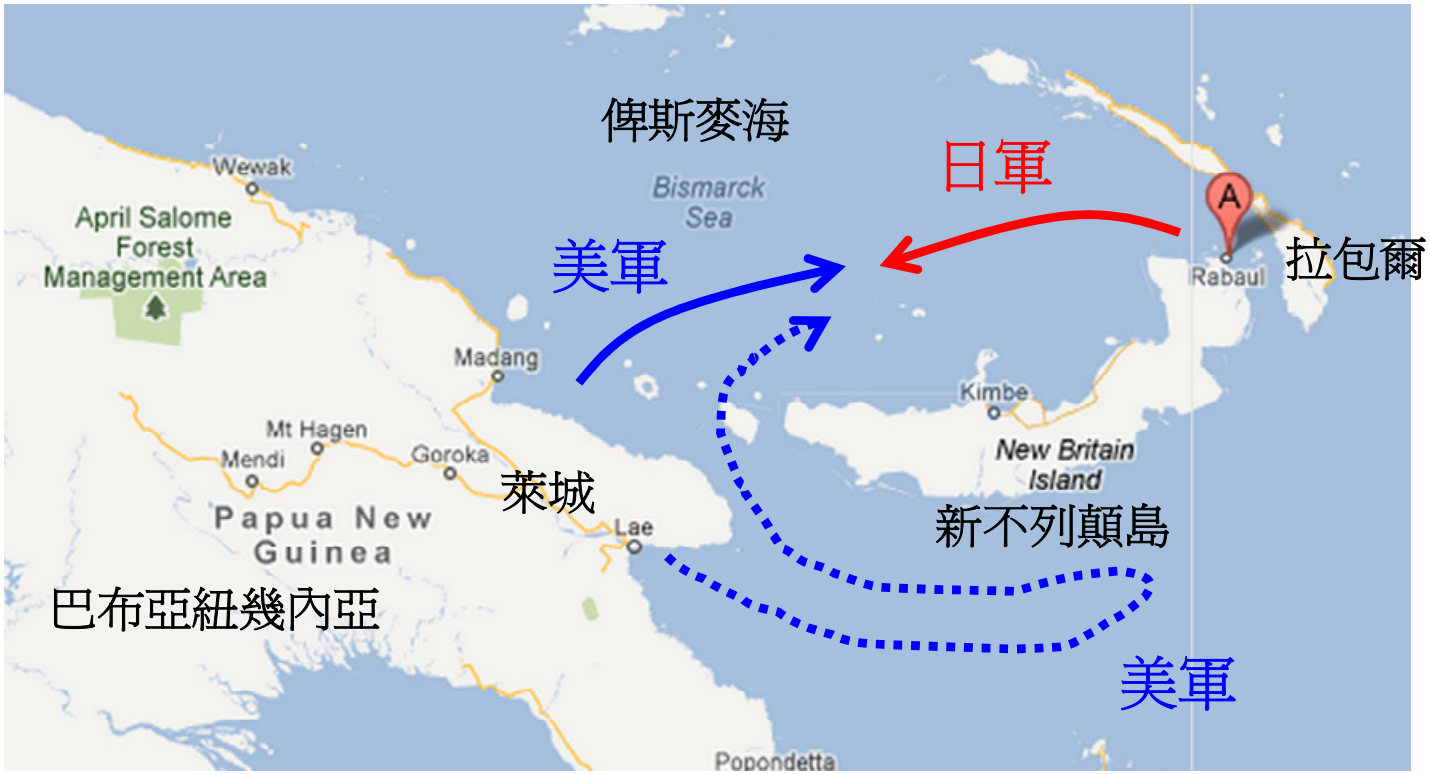
這表示 simplex algorithm 的 time complexity 爲指數時間。另一方面，由於 simplex algorithm 慎選 EV 和 LV，因此實務上，執行速度很快。這讓人們懷疑 linear programming 並不是 NP-complete。1984 年，N. Karmarkar 發明 interior-point method（美國專利號碼：4744026），可在多項式時間之內解決 linear programming 問題，勝過 simplex algorithm。1988 年 Karmarkar 獲得 Fulkerson Prize

A Story about Game Theory (1/3)

二戰末期，日軍在「巴布亞紐幾內亞」和美軍交戰，戰況相當不利。1943 年 2 月 28 日，美方獲得情資：日軍預計從「拉包爾」派遣 8 艘驅逐艦 和 8 艘運兵船前往「萊城」支援。預計航行共需三天。日軍有二個選項：走北方航線（實線）或者南方航線（虛線）。根據氣象資料，北方航線氣候惡劣，而南方航線天氣晴朗。然而美軍並不知道日軍會選擇哪個航線。

A Story about Game Theory (2/3)

美軍打算派遣軍機去轟炸日本軍艦。假設日本軍艦走北方航線。美方軍機有二個選項：如果飛行北方航線，由於氣候惡劣，三天的軍機飛行裡頭，只有二天能轟炸。如果美軍飛行南方航線，則會因爲路線猜錯的關係，必須耗費一天更改航線，緊急飛往北方路線，所以只剩一天能夠轟炸。



A Story about Game Theory (3/3)

美軍打算派遣軍機去轟炸日本軍艦。假設日本軍艦走南方航線。美方軍機有二個選項：如果飛行南方航線，由於天氣晴朗，適合轟炸，所以可以轟炸三天。如果美軍飛行北方航線，則會因爲路線猜錯的關係，必須耗費一天時間更改航線，緊急飛往南方航線，所以只剩二天能夠轟炸。

研究 Game Theory 的目的：預測賽局結果

美軍指揮官
喬治肯尼

日軍指揮官
木村昌福

payoff matrix (美軍, 日軍)		日軍	
		北方航線	南方航線
美軍	北方航線	（2，-2） （轟炸2天，被轟炸2天）	（2，-2） （轟炸2天，被轟炸2天）
	南方航線	（1，-1） （轟炸1天，被轟炸1天）	（3，-3） （轟炸3天，被轟炸3天）

由於美日雙方都有氣象和彼此軍備的資料，因此雙方都知道上面這個戰況分析表。如果你是日軍指揮官，你會選擇「南方航線」還是「北方航線」？ 如果你是美軍指揮官，你要派遣軍機飛行「南方航線」還是「北方航線」？

Best Reply Function for US Air Force

payoff matrix (美軍, 日軍)		日軍	
		北方航線	南方航線
美軍	北方航線	（2，-2） （轟炸2天，被轟炸2天）	（2，-2） （轟炸2天，被轟炸2天）
	南方航線	（1，-1） （轟炸1天，被轟炸1天）	（3，-3） （轟炸3天，被轟炸3天）

payoff matrix (美軍, 日軍)		日軍	
		北方航線	南方航線
美軍	北方航線	（2，-2） （轟炸2天，被轟炸2天）	（2，-2） （轟炸2天，被轟炸2天）
	南方航線	（1，-1） （轟炸1天，被轟炸1天）	（3，-3） （轟炸3天，被轟炸3天）

從上面這個分析表，我們得知：如果日軍選擇「北方航線」，那麼美軍選擇「北方航線」比較好。如果日軍選擇「南方航線」，那麼美軍選擇「南方航線」比較好。問題是美軍怎麼知道日軍要選擇哪個航線？

Best Reply Function for Japanese Navy

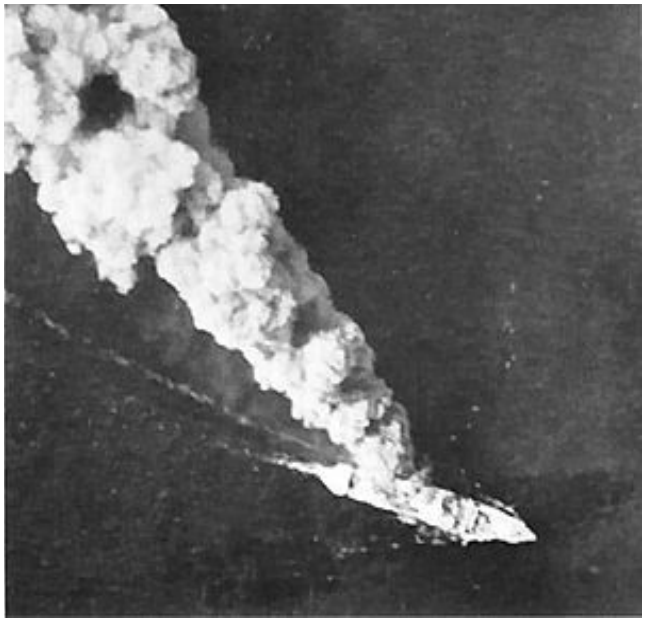
payoff matrix (美軍, 日軍)		日軍	
		北方航線	南方航線
美軍	北方航線	（2，-2） （轟炸2天，被轟炸2天）	（2，-2） （轟炸2天，被轟炸2天）
	南方航線	（1，-1） （轟炸1天，被轟炸1天）	（3，-3） （轟炸3天，被轟炸3天）

payoff matrix (美軍, 日軍)		日軍	
		北方航線	南方航線
美軍	北方航線	（2，-2） （轟炸2天，被轟炸2天）	（2，-2） （轟炸2天，被轟炸2天）
	南方航線	（1，-1） （轟炸1天，被轟炸1天）	（3，-3） （轟炸3天，被轟炸3天）

從上面這個分析表，我們得知：如果美軍選擇「北方航線」，那麼日軍選擇「北方航線」和「南方航線」沒什麼差。如果美軍選擇「南方航線」，那麼日軍選擇「北方航線」較好。所以無論在何種情況之下，「選擇北方航線」對日軍來說都是好的。

Conclusion：Battle of the Bismarck Sea

當然，美軍也會幫日軍分析。美軍知道日軍是聰明人，會選擇「北方航線」。
在日軍會選擇「北方航線」的情況下，美軍當然選擇「北方航線」較佳。



結論：1943年3月2、3日，美日雙方在「北方航線」爆發「俾斯麥海」海戰。
美軍共 2 架戰鬥機 和 3 架轟炸機被擊毀，士兵陣亡 13 人。
而日軍 4 艘驅逐艦被毀，8 艘運兵船全數沈沒，士兵陣亡超過 3000 人。

Pure Strategy Game（靜態賽局）

payoff matrix (美軍, 日軍)		日軍	
		北方航線	南方航線
美軍	北方航線	（2，-2） （轟炸2天，被轟炸2天）	（2，-2） （轟炸2天，被轟炸2天）
	南方航線	（1，-1） （轟炸1天，被轟炸1天）	（3，-3） （轟炸3天，被轟炸3天）

定義：A pure strategy game $G = \langle N, (A_i), (u_i) \rangle$ is a game that consists of

1) a set of players $N = \{1, 2, \ldots, n\}$,

2) for each player i, a set of actions A_i,

3) for each player i, a utility (or payoff) u_i over each action profile $(a_1, a_2, \ldots, a_n)$.

例如： 對於「俾斯麥海」海戰賽局 $G = \langle N, (A_i), (u_i) \rangle$ 來說

1) players 的集合 N = {美軍, 日軍},

2) 對於日軍和美軍，action set 都是 {北方航線, 南方航線}。

3) $u_{美軍}$(北方航線, 北方航線) = 2, $u_{日軍}$(北方航線, 北方航線) = −2

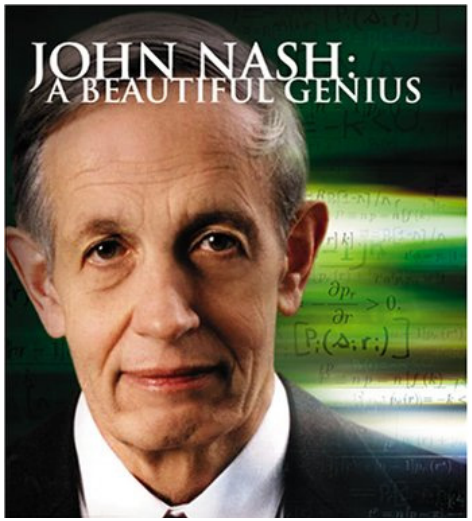
The Assumptions of Pure Strategy Game

Pure strategy game 假設

1. 每個 **player** 都知道有哪些對手。
2. 每個 **player** 都知道有對手有哪些 **action** 選項。
3. 每個 **player** 都知道每個 **action profile** 的 **payoff**。
4. 每個 **player** 都必須「同時出招」。所謂的「同時出招」不是堅持「必須同一個時間點」出招，而是 **player 1** 不能看到 **player 2** 出招的結果，然後才決定要出什麼招。換言之，**player 1** 出招時，不知道 **player 2** 出什麼招。

輪流出招的 game 稱爲 extensive game。有興趣的同學可以閱讀這本書

Nash Equilibrium for Pure Strategy Game



John Nash：1928 年出生，1950 年（**22 歲**）獲得普林斯頓大學博士學位，博士論文提出 Nash equilibrium。1994 年獲得諾貝爾經濟學獎。

Pure strategy game 的 Nash equilibrium（定義）：對於 $G = \langle N, (A_i), (u_i) \rangle$，若一個 action profile（行動組合）$a^* = (a_1^*, a_2^*, ..., a_n^*)$ 爲 Nash equilibrium，則對任意 $i \in N$ 和 $a_i \in A_i$ ，我們都有 $u_i(a_i^*, a_{-i}^*) \geq u_i(a_i, a_{-i}^*)$

在 pure strategy game 裡頭，一個 action profile（行動組合）被稱爲 Nash equilibrium（**簡寫成 NE**），表示在該 action profile 裡頭，沒有人可以單獨改變自己的行爲而獲得更高的利益。

Find Nash Equilibriums via Best Reply Function

payoff matrix (美軍, 日軍)		日軍	
		北方航線	南方航線
美軍	北方航線	(2，-2) (轟炸2天，被轟炸2天)	(2，-2) (轟炸2天，被轟炸2天)
	南方航線	(1，-1) (轟炸1天，被轟炸1天)	(3，-3) (轟炸3天，被轟炸3天)

(北方航線, 北方航線) 是 NE，因爲

$u_{美軍}$(北方航線, 北方航線) $= 2 \geq u_{美軍}$(南方航線, 北方航線) $= 1$

$u_{日軍}$(北方航線, 北方航線) $= -2 \geq u_{日軍}$(北方航線, 南方航線) $= -2$

NE 可以用來預測哪個 action profile 會發生：不管誰來玩，只要雙方都是聰明人，應該都會選擇 NE 的這個 action profile

定理（證明省略）：Nash equilibrium 必定是滿足下列條件的 action profile a^* ：$a_i^* \in \text{BestReply}(a_{-i}^*)$, for all $i \in N$.

意思是說：NE 必定發生在所有 players 都採取 best reply 的時候

Rock-Paper-Scissors：Zero-Sum Game

payoff matrix

A, B	B 出剪刀	B 出石頭	B 出布
A 出剪刀	0, 0	−1, 1	1, −1
A 出石頭	1, −1	0, 0	−1, 1
A 出布	−1, 1	1, −1	0, 0

A、B 兩人玩剪刀-石頭-布的遊戲，輸的人要賠 1 萬，贏的人可得 1 萬。

Not All Zero-Sum Games Have Pure NE

A, B	B 出 剪刀	B 出 石頭	B 出 布
A 出 剪刀	0, 0	−1, 1	1, −1
A 出 石頭	1, −1	0, 0	−1, 1
A 出 布	−1, 1	1, −1	0, 0

(A 出 石頭, B 出 剪刀) 這個 action profile 並不是 Nash equilibrium，因爲

u_B（石頭, 布）> u_B（石頭, 剪刀）

如果 (石頭, 剪刀) 是 Nash equilibrium，表示不管誰來玩這個 game，雙方都自然會採取 (石頭, 剪刀) 這個 action profile，而且雙方都沒有意願改變。然而，當玩家 C 知道玩家 A 沒有願意改變「石頭」這個的選項時，C 會出「布」。

其餘情況同學們自己推導看看。Rock-paper-scissors game 並不存在 (pure) Nash equilibrium。這個範例想告訴大家的是：並不是所有的問題 (game) 都有 deterministic solutions (pure action profiles)。

Mixed Strategy Game

A, B	B 選 剪刀 的機率 q_1	B 選 石頭的機率 q_2	B 選 布 的機率 $1-q_1-q_2$
A 選 剪刀 的機率 p_1	0, 0	−1, 1	1, −1
A 選 石頭 的機率 p_2	1, −1	0, 0	−1, 1
A 選 布 的機率 $1-p_1-p_2$	−1, 1	1, −1	0, 0

由於這個問題不存在 deterministic solution，因此我們尋求看看是否有 randomized solution。如上圖，我們想知道當 A 可以選擇 p_1 和 p_2 的值，B 也可以選擇 q_1 和 q_2 的值時，是否存在 (mixed) Nash equilibrium？

註：由於此時 player 不會純粹固定只選擇某個選項，因此這樣的 game 稱之爲 mixed strategy game.

Linear Programming for Mixed Strategy Game (1/2)

A, B	B 選 剪刀 的獲益	B 選 石頭 的獲益	B 選 布 的獲益
A 選 剪刀 的機率 p_1	0	p_1	$-p_1$
A 選 石頭 的機率 p_2	$-p_2$	0	p_2
A 選 布 的機率 $1-p_1-p_2$	$1-p_1-p_2$	$-(1-p_1-p_2)$	0
B 獲益的期望值	$1-p_1-2p_2$	$-1+2p_1+p_2$	$-p_1+p_2$

對 B 來說，他要最大化自己的利益，所以他會選擇 {剪刀、石頭、布} 裡頭的其中一個，看哪一個可以使得他的獲利 x 最大。所以我們有 $x = \max\{1-p_1-2p_2, -1+2p_1+p_2, -p_1+p_2\}$。然而 B 的獲利就是 A 的損失，所以 A 必須選擇 p_1 和 p_2 以便讓 x 的值最小。因此，A 必須解決下列的 linear program：

$$\left.\begin{array}{l}\min\ x\\ x \ge 1-p_1-2p_2\\ x \ge -1+2p_1+p_2\\ x \ge -p_1+p_2\\ 0 \le p_1, p_2 \le 1\\ -\infty < x < \infty\end{array}\right\} \Rightarrow \left.\begin{array}{l}\max\ z = -x\\ x+p_1+2p_2-e_1 = 1\\ x-2p_1-p_2-e_2 = -1\\ x+p_1-p_2-e_3 = 0\\ x = a-b,\ p_1-s_1 = 1,\ p_2-s_2 = 1\\ p_1, p_2, s_1, s_2, e_1, e_2, e_3, a, b \ge 0\end{array}\right\} \Rightarrow$$

最佳解爲 $p_1 = p_2 = 1/3$，此時 x 的值爲 0。

同學們自己練習看看

Justify *A*'s Mixed Strategy via *B*'s Expected Payoff

A, B	B 選 剪刀 (b_1) 的機率 q_1	B 選 石頭 (b_2) 的機率 q_2	B 選 布 (b_3) 的機率 $q_3 = 1-q_1-q_2$
A 選 剪刀 (a_1) 的機率 p_1	0, 0	−1, 1	1, −1
A 選 石頭 (a_2) 的機率 p_2	1, −1	0, 0	−1, 1
A 選 布 (a_3) 的機率 p_3	−1, 1	1, −1	0, 0

$$B \text{ 獲利的期望值} = \sum_{i=1}^{3}\sum_{j=1}^{3} u_B(a_i, b_j) \times (p_i \times q_j)$$
$$= (p_1q_2 - p_1q_3) + (p_2q_3 - p_2q_1) + (p_3q_1 - p_3q_2)$$

當 $p_1 = p_2 = p_3 = 1/3$ 時，B 獲利的期望值 ≡ 0（無論 B 出什麼招）

當 $p_1 = 0.4$，$p_2 = 03$，$p_3 = 0.3$ 時（A 出剪刀的機率稍微高一點點）

B 獲利的期望值 = $0.1*(q_2-q_3)$

令 $q_2 = 1$（B 百分之百出石頭），$q_3 = 0$（此時自然 $q_1 = 0$）

B 獲利的期望值在此時達到最大！（等於 0.1）

Linear Programming for Mixed Strategy Game (2/2)

A, B	B 選 剪刀 的機率 q_1	B 選 石頭的機率 q_2	B 選 布 的機率 $1-q_1-q_2$	A 獲益的期望值
A 選 剪刀 的獲益	0	$-q_2$	$1-q_1-q_2$	$1-q_1-2q_2$
A 選 石頭 的獲益	q_1	0	$-(1-q_1-q_2)$	$-1+2q_1+q_2$
A 選 布 的機率獲益	$-q_1$	q_2	0	$-q_1+q_2$

對 A 來說，他要最大化自己的利益，所以他會選擇 {剪刀、石頭、布} 裡頭的其中一個，看哪一個可以使得他的獲利 y 最大。所以我們有 $y = \max\{ 1-q_1-2q_2, -1 + 2q_1 + p_2, -q_1 + q_2\}$。然而 A 的獲利就是 B 的損失，所以 B 必須選擇 q_1 和 q_2 以便讓 y 的值最小。因此，B 必須解決下列的 linear program：

$$
\left.
\begin{aligned}
&\min\ y \\
&y \geq 1 - q_1 - 2q_2 \\
&y \geq -1 + 2q_1 + q_2 \\
&y \geq -q_1 + q_2 \\
&0 \leq q_1,\ q_2 \leq 1 \\
&-\infty < y < \infty
\end{aligned}
\right\}
\Rightarrow
\begin{aligned}
&\text{最佳解爲 } q_1 = q_2 = 1/3\ ， \\
&\text{此時 } y \text{ 的值爲 } 0\ 。\text{所以此一遊戲} \\
&\text{的 Nash equilibrium 爲：} \\
&\text{A 出剪刀、石頭、布的機率各 } 1/3 \\
&\text{B 出剪刀、石頭、布的機率各 } 1/3
\end{aligned}
$$

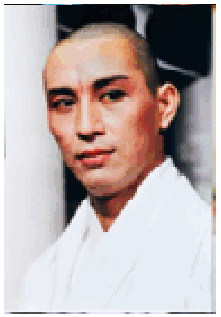
Nash Equilibrium for Mixed Strategy Game

Mixed strategy game 裡頭，假設 player i 的 action set 爲 A_i，那麼 player i 的 strategy（策略）α_i 爲其 action set A_i 上面的機率分配。例如：在「剪刀石頭布」的 game 裡頭，player 1 的 strategy 可以是 (1/3, 1/3, 1/3)，也可以是 (0.4, 0.3, 0.3)。

Mixed strategy game 的 Nash equilibrium 定義：一個 strategy profile（策略組合）$\alpha^* = \{\alpha_1^*, \cdots, \alpha_n^*\}$ 爲 NE，表示沒有人可以單獨改變自己 action set 上面的機率分配而獲得更高的利益期望值。例如：{(0.4, 0.3, 0.3), (1/3, 1/3, 1/3)} 不是 NE，因爲 player 2 的 strategy 改成 (0, 1, 0) 可獲得更高的利益期望值

自我評量

payoff matrix



中原隊, 西域隊	無花	石觀音
楚留香	3, −3	−1, 1
蘇蓉蓉	−2, 2	1, −1

楚留香和蘇蓉蓉勇闖西域，準備與石觀音及無花對決。上表顯示對決之後的結果。
如果楚留香和無花對決，將可勝三分，但和石觀音對決將輸一分。
若蘇蓉蓉和無花對決，將輸二分，但和石觀音對決將可勝一分。對決的時候只能有一人出場，試求出中原隊和西域隊比賽的 Nash equilibrium。

註：限用 simplex algorithm。計算過程中，若沒有出現 basic variable, non-basic variable, entering variable, leaving variable, ratio test，本題以 0 分計算。

[答案] Nash 均衡爲：中原隊以3/7的機率派出楚留香，西域隊以 2/7 的機率派出無花
[註] 雖然此時西域隊的期望值爲 −1/7，但若西域隊不採用此一策略，
那麼中原隊可改用其他策略使得西域隊的期望值比 −1/7 更低。