練習題完全詳解

不可判定性 · NP 完備 · DPLL · 2SAT · 機率圖靈機 · 單純形法 · TSP

5CCS2FC2 Foundations of Computing II — 測驗題逐題解析

涵蓋 Exercise-1 至 Exercise-14 全部題目

2026 年 7 月

本詳解的使用方式

十四張截圖共包含兩份測驗：

- 第一部分（**Exercise-1 ∼ 3**）：不可判定性測驗，共 5 題選擇題，主題對應 Week 10（可判定語言、停機問題、Entscheidungsproblem）。
- 第二部分（**Exercise-4 ∼ 14**）：綜合測驗，共 11 題，涵蓋 NP 類別、SAT → CLIQUE 歸約、主定理、DPLL 演算法、2SAT 蘊涵圖、機率圖靈機、單純形法、TSP 的 2-opt 與近似演算法。

每題的格式為：題目重述 → 背景觀念 → 逐選項詳細分析（或完整計算過程） → 最終答案。建議先自己作答，再對照解析。

Contents

1 第一部分：不可判定性測驗（5 題）

1.1 Q1：哪些問題已知是不可判定的？

題目（複選，2 分）

下列哪些問題已知（**known**）是不可判定的？

(A) 布林可滿足性問題（The Boolean Satisfiability Problem, SAT）

(B) 陷阱題！不可判定問題只是被「假設」存在，至今沒有人找到實際例子。

(C) 圖靈機的空語言問題（The emptiness problem for Turing Machines）

(D) 圖靈機的停機問題（The Halting problem for Turing Machines）

(E) 旅行推銷員問題（The Travelling Salesman Problem, TSP）

背景觀念：「很難」與「不可判定」是兩回事

「不可判定（undecidable）」的意思是：數學上證明了不存在任何健全、完備且對所有輸入都會停機的演算法。它跟「需要很久才算得完」（如 NP-hard 問題）完全不同——NP-hard 問題再慢仍然有演算法可解，不可判定問題則是根本沒有演算法。

逐選項分析

- **(A) SAT**：× 錯誤。SAT 是可判定的——公式只有有限多個變數 n，把 2^n 種真值指派全部列出來逐一檢查，必定在有限時間內得到答案。SAT 的著名之處是它是 **NP-complete**（Cook–Levin 定理），意即「可判定但（目前相信）很難快速解」，這與不可判定是兩個不同層次的概念。
- **(B)** 陷阱題選項：× 錯誤。不可判定問題不是假設，而是被嚴格證明存在。圖靈 1936 年就用對角線論證給出了具體例子（停機問題）。這個選項描述的情境比較像 P $\neq$ NP 那種「尚未證明的猜想」，但不可判定性早已是定理。
- **(C)** 圖靈機的空語言問題：✓ 正確。此問題問：「給定圖靈機 M，M 接受的語言是否為空集合（$L(M) = \varnothing$）?」它已被證明不可判定：可以把接受問題歸約到它——想知道 M 是否接受 w，可以造一台新機器 M_w，M_w 忽略自己的輸入、直接模擬 M 執行於 w，且唯有 M 接受 w 時 M_w 才接受。於是 $L(M_w) \neq \varnothing \iff M$ 接受 w。若空語言問題可判定，接受問題就可判定，矛盾。（更一般地，萊斯定理指出：關於圖靈機語言的任何非平凡性質都不可判定。）
- **(D)** 停機問題：✓ 正確。這是最經典的不可判定問題：「給定 $\langle M, w\rangle$，M 在輸入 w 上是否會停機 ?」圖靈用對角線法（自我指涉的唱反調機器）證明不存在判定它的演算法。
- **(E) TSP**：× 錯誤。TSP 是可判定的：n 個城市的走訪順序只有 $(n-1)!/2$ 種，全部列舉、取最短即可——極慢（NP-hard），但保證會停。

答案

(C) 圖靈機的空語言問題與 **(D)** 圖靈機的停機問題。

1.2 Q2：不可判定語言必然具有哪些性質？

題目（複選，2 分）

設 L 是不可判定語言。下列敘述哪些必然為真？

(A) L 是 NP-hard。

(B) L 是無窮語言，即 L 包含無窮多個字。

(C) 任何對 L 健全（sound）且完備（complete）的演算法，必在某些輸入上無法終止。

(D) 任何對 L 健全且完備的演算法，必在所有輸入上無法終止。

(E) L 是 NP-complete。

逐選項分析　**(A)** L 是 **NP-hard**：✓ 正確。NP-hard 的意思是「至少跟 NP 裡的每一個問題一樣難」。NP 裡的每個問題都是可判定的（頂多用指數時間暴力搜尋所有證書即可判定）；而 L 連「可判定」都做不到，也就是說任何 NP 問題若能解，也遠遠碰不到 L 的難度天花板——L 嚴格難於 NP 中的一切問題，因此是 NP-hard。反過來想更清楚：如果 L 竟然「不比某個 NP 問題難」，那 L 的難度就被一個可判定問題罩住，與 L 不可判定矛盾。

(B) L 是無窮語言：✓ 正確。用反證法：若 L 是有限的，例如 $L = \{w_1, \ldots, w_m\}$，那就能寫一個超簡單的判定器——把這 m 個字硬編碼成查詢表，輸入進來逐一比對，符合就輸出 True、否則 False。這個程式健全、完備、永遠停機，於是 L 可判定，矛盾。所以不可判定語言一定包含無窮多個字。（同理其補集 $\overline{L}$ 也必是無窮的。）

(C) 健全且完備的演算法必在某些輸入上不終止：✓ 正確。這正是不可判定的定義換句話說。判定器 = 健全 + 完備 + 對所有輸入終止。既然 L 沒有判定器，那任何做到健全與完備的演算法，就必然在第三個條件上失守——存在至少一個輸入讓它跑不完。

(D) 必在所有輸入上不終止：× 錯誤。「某些」不能強化成「所有」。例如停機問題的辨識器（用萬能圖靈機直接模擬）：對每一個「會停」的輸入它都能終止並輸出 True，只在「不會停」的輸入上卡住。一個在所有輸入上都不終止的程式反而什麼都沒計算，根本稱不上完備。

(E) L 是 **NP-complete**：× 錯誤。NP-complete = NP-hard + 屬於 **NP**。屬於 NP 意味著存在多項式時間的非決定性判定程序，特別地 L 必是可判定的（確定性地窮舉所有非決定性分支即可，指數時間但有限）。這與 L 不可判定矛盾。所以 L 是 NP-hard 卻永遠不可能是 NP-complete——「hard」沒有上限要求，「complete」有。

答案

(A)、**(B)**、**(C)**。記憶口訣：不可判定 ⇒ 難到 NP-hard、多到無窮、任何正確演算法必有跑不完的輸入；但絕不屬於 NP（故非 NP-complete），也不必在所有輸入上失敗。

1.3 Q3：為什麼不可判定語言的補集也不可判定？

題目（單選，2 分）

已知 L 是不可判定語言，為什麼其補集 $\overline{L}$ 也必然不可判定？

(A) 如果我們能為 $\overline{L}$ 找到健全、完備且會終止的演算法，只要把輸出的 True 與 False 互換，就能為 L 構造出健全、完備且會終止的演算法。

(B) 陷阱題！L 不可判定則必為無窮語言，所以其補集必為有限；而有限語言皆可判定！

(C) 陷阱題！不可判定問題的補集永遠是可判定的，因為「不可判定的相反」就是可判定。

(D) 每個語言都可以多項式歸約到它的補集，反之亦然。

詳細解答　正解是 **(A)**，論證如下（反證法）：

假設 $\overline{L}$ 可判定，即存在判定器 $\overline{M}$：健全、完備、對所有輸入終止。現在構造新機器 M：

$M(w)$：執行 $\overline{M}(w)$；若 $\overline{M}$ 輸出 True 則輸出 **False**，若輸出 False 則輸出 **True**。

逐項檢查 M 是 L 的判定器：

- 終止：$\overline{M}$ 對所有輸入都在有限時間內停，M 只是多做一步「翻轉輸出」，所以 M 也對所有輸入終止。（這一步是關鍵——翻轉輸出保留了終止性，這在「辨識器」上做不到，因為辨識器可能根本不停，無輸出可翻。）
- 健全：$M(w) = \text{True} \Rightarrow \overline{M}(w) = \text{False} \Rightarrow w \notin \overline{L} \Rightarrow w \in L$。
- 完備：$w \in L \Rightarrow w \notin \overline{L} \Rightarrow \overline{M}(w) = \text{False}$（$\overline{M}$ 健全且終止，對非成員必答 False）$\Rightarrow M(w) = \text{True}$。

於是 L 可判定——與前提矛盾。故 $\overline{L}$ 不可判定。■

其他選項為何錯

- **(B)**：× 錯誤。「無窮語言的補集必有限」是集合論錯誤：Σ^* 本身無窮，無窮集合挖掉一個無窮子集，剩下的完全可以仍是無窮（例如偶數的補集是奇數，皆無窮）。事實上 $\overline{L}$ 必為無窮（若有限則可判定，翻轉輸出後 L 也可判定，矛盾）。
- **(C)**：× 錯誤。「不可判定的相反是可判定」是文字遊戲：補集是對語言內容取補，不是對「可判定性」這個屬性取反。實際上可判定性在取補下封閉：L 可判定 $\iff$ $\overline{L}$ 可判定（正是 (A) 的雙向論證）。
- **(D)**：× 錯誤。「任何語言都可多項式歸約到其補集」並不成立——多項式歸約要求歸約函數在多項式時間內可計算，這對一般語言毫無保證；而且本題的層次是可判定性（用一般歸約即可），扯多項式歸約也文不對題。

答案

(A)：把 $\overline{L}$ 的判定器輸出翻轉，就得到 L 的判定器；因 L 無判定器，故 $\overline{L}$ 亦無。

1.4 Q4：只有健全與完備（但不保證終止）的演算法能給我們什麼？

題目（複選，2 分）

假設我們拿到一個對不可判定語言 L 健全且完備的演算法。下列敘述哪些必然為真？

(A) 若 $w \notin L$，我們可以確定機器終將輸出 False。

(B) 陷阱題！不可判定問題不可能有健全且完備的演算法。

(C) 若 $w \in L$，我們可以確定機器終將輸出 True。

(D) 該演算法必定在某些輸入字上不終止。

背景觀念：辨識器（recogniser）的不對稱性

健全 + 完備但不保證終止的演算法，就是所謂的辨識器。它的行為天生不對稱：

- 「$w \in L$」這一面有保障：完備性說 $w \in L \Rightarrow M(w) = 1$，而「輸出 1」隱含「有終止」——所以成員終將得到 True。
- 「$w \notin L$」這一面沒保障：健全性只說「若輸出 True 則真的屬於 L」，它允許機器對非成員永遠不給答案。你等不到 False，也永遠不確定是「還沒算完」還是「不會回來了」。

逐選項分析

- **(A)**：× 錯誤。如上所述，對 $w \notin L$，機器可能輸出 False，也可能永遠跑下去。健全與完備都沒有承諾「非成員會得到答案」。
- **(B)**：× 錯誤。存在具體反例：停機問題不可判定，但它有健全且完備的演算法——用萬能圖靈機模擬 $\langle M, w\rangle$，一旦 M 停機就輸出 True。所有會停的實例終將被正確接受（完備），且它說 True 時絕對正確（健全）；只是對不停機的實例它會陪著跑到永遠。（附註：並非所有不可判定語言都有辨識器——例如停機問題的補集就沒有；但本題題幹已假設「拿到了」這樣的演算法，且此選項宣稱「不可能有」，一個反例即足以推翻。）
- **(C)**：✓ 正確。這正是完備性的內容：$w \in L \Rightarrow$ 機器終將輸出 True。
- **(D)**：✓ 正確。若它對所有輸入都終止，它就同時滿足健全、完備、終止三條件，成為 L 的判定器，與 L 不可判定矛盾。所以必存在讓它跑不完的輸入（而且由健全 + 完備可知，這些輸入必然都是非成員）。

答案

(C)、**(D)**。一句話總結：辨識器對「Yes」有交代、對「No」沒交代，而且必有卡住的輸入。

1.5 Q5：Entscheidungsproblem 的中譯／英譯是什麼？

題目（單選，2 分）

Entscheidungsproblem 翻譯過來是：

(A) 鴨子離婚問題（The Duck Divorce problem）

(B) 停機問題（The Halting problem）

(C) 謂詞可滿足性問題（The Predicate satisfiability problem）

(D) 布林可滿足性問題（The Boolean satisfiability problem）

(E) 判定問題（The Decision problem）

詳細解答　德文 *Entscheidung* =「決定、判定」，*Problem* =「問題」，所以 Entscheidungsproblem 就是判定問題（**The Decision Problem**）——正解為 **(E)**。這是希爾伯特與阿克曼 1928 年提出的問題：找一個演算法，判定任意謂詞邏輯公式是否恆真。1936 年邱奇與圖靈分別證明它不可判定。

易混淆選項辨析

- **(B)** 停機問題：× 錯誤。停機問題是圖靈用來證明 Entscheidungsproblem 無解的工具，兩者密切相關但不是同一個問題，更不是翻譯關係。
- **(C)** 謂詞可滿足性：× 錯誤。很接近但不對——Entscheidungsproblem 問的是恆真性（**validity/tautology**），不是可滿足性。兩者互為對偶（F 恆真 $\iff$ $\neg F$ 不可滿足），在課堂證明中我們正是透過這個對偶把接受問題歸約過去，但「翻譯」上它是判定問題。
- **(A)** 鴨子離婚問題：出題者的幽默，感謝老師。

答案

(E) The Decision problem（判定問題）。

2 第二部分：綜合測驗（11 題）

2.1 Q1：證明 SUBSET-SUM 屬於 NP（簡答，10 分）

題目

SUBSET-SUM 問題：輸入一個大小為 n 的整數集合 $S = \{a_1, \ldots, a_n\}$ 與目標整數 $k \in \mathbb{Z}$，若存在子集 $S' \subseteq S$ 使得 $\sum_{a \in S'} a = k$ 則回傳 True。請用自己的話描述一個多項式時間的非決定性演算法，證明 SUBSET-SUM 屬於 NP。

背景觀念：NP 的兩種等價定義

1. 非決定性機器觀點：存在非決定性圖靈機，在多項式時間內判定該問題——機器可以在每一步「分岔」，同時探索所有可能，只要任何一條分支接受即算接受。
2. 驗證者觀點：存在多項式時間驗證器 V，對每個 Yes 實例都有一個多項式長度的證書（certificate）使 V 接受。

兩者等價：非決定性的「猜測」就等於「拿到證書」。本題用哪個觀點作答都可以。

詳細解答（非決定性演算法） 演算法描述：

1. 猜測階段（非決定性）：對每個 $i = 1, 2, \ldots, n$，非決定性地選擇「a_i 放入 S'」或「a_i 不放入 S'」。這一步讓計算樹分岔成 2^n 條分支，每條分支對應一個候選子集 S'。
2. 驗證階段（決定性）：在自己這條分支上，計算 $\Sigma = \sum_{a \in S'} a$（至多 n 次整數加法），檢查 $\Sigma = k$ 是否成立；成立則接受，否則拒絕。

正確性：

- 若答案為 Yes（存在和為 k 的子集），則計算樹上「恰好猜中那個子集」的分支會接受 $\Rightarrow$ 非決定性機器接受。（完備）
- 若答案為 No，任何分支算出來的和都 $\neq k$，所有分支都拒絕 $\Rightarrow$ 機器拒絕。（健全）

時間複雜度：關鍵在於 NP 只要求每一條分支是多項式時間，不管分支總數。每條分支做 n 次二選一、至多 n 次加法、一次比較；若每個整數用 m 位元表示，一次加法花 $O(m)$，整條分支 $O(nm)$——是輸入長度的多項式。■

為什麼「暴力列舉 2^n 個子集」不行，非決定性卻可以？

決定性機器要依序檢查 2^n 個子集，總時間指數；非決定性機器把這 2^n 個檢查「攤平」到平行的分支上，每條分支只做自己那一份 $O(nm)$ 的工作。官方參考答案正是這樣說的：「有 2^n 個可能子集，這不是多項式；但用非決定性機器 M 可以平行地檢查每個子集，每個平行處理器最多只需做 n 次加法，花多項式時間。」

2.2 Q2：SAT $\to$ CLIQUE 歸約——哪條公式對應這張圖？（單選，5 分）

題目

CLIQUE 問題：輸入圖 $G=(V,E)$ 與整數 $k>2$，若 G 含大小為 k 的團（clique，兩兩相鄰的 k 個頂點）則回傳 True。圖中給定一個 9 頂點的圖，頂點標記為（上方群）$P,\ R,\ \neg Q$、（左側群）$Q,\ \neg R,\ P$、（下方群）$\neg P,\ Q,\ R$，並圈成三組。依課堂上的多項式歸約，下列哪條公式可滿足 $\iff$ 該圖含大小 $k=3$ 的團？

(A) $F=(P\vee\neg Q\vee R)\wedge(P\vee Q\vee\neg R)\wedge(\neg P\vee Q\vee R)$

(B) $F=(P\vee\neg Q\vee R)\wedge(\neg P\vee\neg Q\vee\neg R)\wedge(P\vee Q\vee R)$

(C) $F=(\neg P\vee\neg Q\vee\neg R)\wedge(\neg P\vee Q\vee\neg R)\wedge(\neg P\vee Q\vee R)$

(D) $F=(P\vee Q\vee R)\wedge(P\vee\neg Q\vee\neg R)\wedge(P\vee\neg Q\vee\neg R)$

背景觀念：課堂上的 SAT $\to$ CLIQUE 歸約

給定 CNF 公式 $F=C_1\wedge C_2\wedge\cdots\wedge C_k$（每個 C_j 是若干文字的析取），構造圖 G：

- 頂點：每個子句裡的每個文字出現（literal occurrence）各設一個頂點——所以子句就是圖上圈起來的「群」。
- 邊：兩頂點相鄰 $\iff$（1）它們屬於不同子句，且（2）它們不互相矛盾（不是 X 與 $\neg X$ 的組合）。
- 參數：$k=$ 子句數。

則 F 可滿足 $\iff$ G 有 k-團。直覺：一個 k-團必須從每個子句各挑一個文字（同子句不相鄰），且挑出的文字互不矛盾，恰好構成「每個子句都被滿足」的一致指派。

詳細解答　反向操作即可：從圖讀回公式。圖上的三個圈就是三個子句，圈內的文字就是子句的文字。

方法一（比對頂點多重集，最快）：數一數圖上 9 個頂點的標籤：

$$\{P,\ P,\ Q,\ Q,\ R,\ R,\ \neg P,\ \neg Q,\ \neg R\}$$

即 P,Q,R 各出現兩次、$\neg P,\neg Q,\neg R$ 各出現一次。逐一檢查選項的文字多重集：

- **(A)**：$\{P,\neg Q,R\}\cup\{P,Q,\neg R\}\cup\{\neg P,Q,R\}=P{\times}2,\ Q{\times}2,\ R{\times}2,\ \neg P,\ \neg Q,\ \neg R$。✓ 正確 完全吻合！
- **(B)**：含 $\neg Q$ 兩次（第一、二子句各一），但圖上只有一個 $\neg Q$ 頂點。× 錯誤
- **(C)**：含 $\neg P$ 三次，圖上只有一個 $\neg P$。× 錯誤
- **(D)**：含 P 三次、且 $\neg Q,\neg R$ 各兩次，皆與圖不符。× 錯誤

方法二（比對分組）：圖上三個圈分別是 $\{P,R,\neg Q\}$、$\{Q,\neg R,P\}$、$\{\neg P,Q,R\}$，直接寫出

$$F=(P\vee R\vee\neg Q)\wedge(Q\vee\neg R\vee P)\wedge(\neg P\vee Q\vee R),$$

調整文字順序即為選項 (A)。也可以抽查邊來驗證：例如上群的 P 與下群的 $\neg P$ 之間不該有邊（矛盾文字）、同群的 P 與 R 之間不該有邊（同子句），而上群 P 與左群 Q 之間該有邊——與圖一致。

答案

(A) $F = (P \vee \neg Q \vee R) \wedge (P \vee Q \vee \neg R) \wedge (\neg P \vee Q \vee R)$。

2.3 Q3：為什麼 3-團的文字構成 F 的滿足指派？（簡答，10 分）

題目

請用自己的話解釋：為什麼 G 中任何 3-團所含的文字，構成 F 的一個滿足指派（satisfying assignment）？

詳細解答　設 C 是 G 的一個 3-團（三個兩兩相鄰的頂點）。逐步論證：

第一步：C 必然「每個子句恰取一個文字」。歸約的建圖規則裡，同一子句內的頂點之間沒有邊。所以團裡不可能同時包含同一子句的兩個文字（它們不相鄰，違反團的定義）。C 有 3 個頂點、圖上恰有 3 個子句，由鴿籠原理，C 恰好從每個子句各取一個文字。

第二步：C 中的文字互不矛盾。建圖規則同時規定矛盾文字（X 與 $\neg X$）之間沒有邊。團中任兩頂點都相鄰，所以 C 不可能同時含 X 與 $\neg X$。

第三步：由 C 讀出一個合法的真值指派。把 C 中出現的每個文字設為真：若 $X \in C$ 令 $X = \text{True}$，若 $\neg X \in C$ 令 $X = \text{False}$（C 中未提及的變數任意設定）。由第二步，這個規則不會對同一變數下達矛盾指令，所以它是一個良定義（一致）的指派。

第四步：這個指派滿足 F。由第一步，每個子句 C_j 都有一個文字被選入 C、並在第三步被設為真；子句是「或」，一個文字為真即整個子句為真。三個子句皆真，故合取式 F 為真。■

對照官方參考答案

官方答案濃縮成兩點，正是上述第一、二步：「C 必須從每個子句各含一個文字，因為同子句的文字不相連；C 不可能包含互相矛盾的文字（那會使指派不合法），因為矛盾文字也不相連。」寫作答案時，建議像上面那樣補上第三、四步（指派良定義、每個子句被滿足），論證才完整。

2.4 Q4：主定理——哪些遞迴關係的解是 $\Theta(n^3)$？（複選，8 分）

題目

利用主定理（Master Theorem），判斷下列哪些遞迴關係描述的函數成長速率為 $\Theta(n^3)$。所有情況下皆假設 $T(1) = 1$。

(A) $T(n) = 9\,T(n/3) + (n^2 - n - 1)$

(B) $T(n) = 2\,T(n/2) + \frac{1}{6}n(n+1)(2n+1)$

(C) $T(n) = 5\,T(n/32) + \log_2(n^2 + 1)$

(D) $T(n) = 8\,T(n/2) + n$

(E) 以上皆非。

背景觀念：主定理三情況

對 $T(n) = a\,T(n/b) + f(n)$（$a \geq 1, b > 1$），令臨界指數 $d^* = \log_b a$，把 $f(n)$ 與 n^{d^*} 比大小：

1. 情況 **1**（遞迴主導）：$f(n) = O(n^{d^*-\varepsilon})$（$f$ 嚴格較小）$\Rightarrow T(n) = \Theta(n^{d^*})$。
2. 情況 **2**（平手）：$f(n) = \Theta(n^{d^*}) \Rightarrow T(n) = \Theta(n^{d^*}\log n)$。
3. 情況 **3**（f 主導）：$f(n) = \Omega(n^{d^*+\varepsilon})$ 且滿足正則條件 $a\,f(n/b) \leq c\,f(n)$（某 $c < 1$）$\Rightarrow T(n) = \Theta(f(n))$。

逐選項計算　**(A)** $a = 9,\ b = 3$：$d^* = \log_3 9 = 2$，即 $n^{d^*} = n^2$。而 $f(n) = n^2 - n - 1 = \Theta(n^2)$——與 n^{d^*} 同階，情況 **2**：

$$T(n) = \Theta(n^2 \log n).$$

$n^2\log n$ 比 n^3 慢（$\log n = o(n)$），× 錯誤。

(B) $a = 2,\ b = 2$：$d^* = \log_2 2 = 1$，$n^{d^*} = n$。注意 $f(n) = \frac{1}{6}n(n+1)(2n+1)$ 正是平方和公式 $\sum_{i=1}^{n} i^2$，展開最高次項為 $\frac{2n^3}{6} = \frac{n^3}{3}$，故 $f(n) = \Theta(n^3)$。n^3 遠大於 $n^{d^*} = n^1$（取 $\varepsilon = 2$），檢查正則條件：

$$a\,f(n/b) = 2 \cdot \tfrac{1}{6} \cdot \tfrac{n}{2}\left(\tfrac{n}{2}+1\right)(n+1) \approx \tfrac{n^3}{12} \leq c \cdot \tfrac{n^3}{3} \quad (\text{取} c = \tfrac{1}{2} < 1)\ \checkmark$$

情況 **3**：$T(n) = \Theta(f(n)) = \Theta(n^3)$，✓ 正確。

(C) $a = 5,\ b = 32$：$d^* = \log_{32} 5 = \frac{\ln 5}{\ln 32} \approx \frac{1.609}{3.466} \approx 0.464$。$f(n) = \log_2(n^2+1) = \Theta(\log n)$。對數成長慢於任何正冪次：$\log n = O(n^{0.464-\varepsilon})$（例如取 $\varepsilon = 0.2$），情況 **1**：

$$T(n) = \Theta\left(n^{\log_{32} 5}\right) \approx \Theta(n^{0.464}).$$

連 n 都不到，離 n^3 差得遠，× 錯誤。

(D) $a = 8,\ b = 2$：$d^* = \log_2 8 = 3$，$n^{d^*} = n^3$。$f(n) = n = O(n^{3-\varepsilon})$（取 $\varepsilon = 1$），情況 **1**：

$$T(n) = \Theta(n^3),$$

✓ 正確。（這正是普通矩陣乘法分治的遞迴形狀。）

答案

(B) 與 **(D)**。兩條達到 $\Theta(n^3)$ 的路徑恰好相反：(B) 靠「每層做的工 $f(n) = \Theta(n^3)$ 主導」，(D) 靠「分支數 8 使葉子數量 $n^{\log_2 8} = n^3$ 主導」。

2.5 Q5：DPLL 應該先套用哪條規則？（單選，5 分）

題目

考慮下列命題子句集：

$$(\neg Q \vee \neg R \vee T),\quad (\neg P \vee Q \vee R \vee S),\quad (P \vee Q \vee \neg S),\quad (\neg Q \vee T),\quad (\neg P \vee R \vee \neg S)$$

對此子句集執行 DPLL 演算法時，應最先套用下列哪條規則？（選項：對 T 分支 / 其他皆非 / 以 $\neg T$ 做純文字消去 / 以單元子句 S 做單元傳播 / 以 T 做純文字消去 / 以單元子句 $\neg S$ 做單元傳播 / 矛盾，回溯）

背景觀念：DPLL 的規則優先順序

DPLL（Davis–Putnam–Logemann–Loveland）在每一步依下列優先序選擇動作：

1. 單元傳播（**Unit Propagation**）：若存在只含一個文字的子句（單元子句），該文字被迫為真，立即指派並化簡。
2. 純文字消去（**Pure Literal Elimination**）：若某變數在整個子句集中只以一種極性出現（只出現 X 或只出現 $\neg X$），直接把該文字設為真，刪除所有包含它的子句。
3. 分支（**Branching**）：以上皆不可用時，挑一個變數猜真值，之後可能回溯。

詳細解答　第一步：找單元子句。逐一檢查五個子句的長度：$3, 4, 3, 2, 3$——最短的是 $(\neg Q \vee T)$，有兩個文字，沒有單元子句，單元傳播不可用（所以「以 S / $\neg S$ 做單元傳播」都錯，S 與 $\neg S$ 出現的子句都不是單元子句）。

第二步：逐變數檢查純文字。列出每個變數出現的極性：

變數	正出現	負出現	判定
P	子句 3	子句 2, 5	兩種極性都有，不純
Q	子句 2, 3	子句 1, 4	不純
R	子句 2, 5	子句 1	不純
S	子句 2	子句 3, 5	不純
T	子句 1, 4	——	只有正出現：純文字！

T 是純文字（$\neg T$ 從未出現），所以正確動作是以文字 T 做純文字消去：令 $T = \text{True}$，刪除子句 1 與子句 4（它們已被滿足）。注意「以 $\neg T$ 做純文字消去」是錯的——純的是 T 這個正文字。

其他選項排除：不需要分支（還有更便宜的規則可用）；也沒有空子句，不需回溯。

答案

以文字 T 做純文字消去（**Pure Literal Elimination with literal** T）。

2.6 Q6：對 Q 做純文字消去的結果（單選，5 分）

題目

考慮下列子句集：

$$(P \vee R \vee \neg S \vee \neg T),\quad (P \vee Q),\quad (Q \vee R \vee \neg S),\quad (\neg R),\quad (P \vee Q \vee \neg R \vee \neg T)$$

對文字 Q 執行純文字消去後，結果為何？

詳細解答　確認 Q 是純文字：Q 出現在子句 2、3、5，全部是正出現；$\neg Q$ 不出現。✓ 純文字。

規則的效果：純文字消去令 $Q = \text{True}$。凡是包含 Q 的子句都因此被滿足，可整句刪除；不包含 Q 的子句完全不動（注意 $\neg Q$ 不存在，所以沒有任何子句需要「刪掉其中一個文字」）。

逐句處理：

子句	含 Q？	處理
$(P \vee R \vee \neg S \vee \neg T)$	否	保留
$(P \vee Q)$	是	刪除
$(Q \vee R \vee \neg S)$	是	刪除
$(\neg R)$	否	保留
$(P \vee Q \vee \neg R \vee \neg T)$	是	刪除

剩下的子句集為：

$$(P \vee R \vee \neg S \vee \neg T),\qquad (\neg R).$$

常見錯誤：選到「$(R \vee \neg S \vee \neg T)$, (Q), ...」那種把 Q 從子句裡拿掉的選項——那是把純文字消去跟單元傳播的化簡動作搞混了。純文字消去是整句刪除含該文字的子句，絕不會把文字從子句中剔除，也不會產生新單元子句。

答案

$(P \vee R \vee \neg S \vee \neg T)$ 與 $(\neg R)$ 兩句。

2.7 Q7：2SAT 蘊涵圖的強連通分量（複選，6 分）

題目

考慮 2SAT 實例：

$$(\neg P \vee S) \wedge (P \vee \neg R) \wedge (\neg Q \vee S) \wedge (\neg Q \vee \neg S) \wedge (\neg R \vee \neg S) \wedge (R \vee \neg S)$$

下列哪些文字集合是其蘊涵圖（**implication graph**）的強連通分量（SCC）？選項：$\{R\}$、$\{P, R, S\}$、$\{\neg P, P, \neg Q, \neg R, R, S, \neg S\}$、$\{\neg Q\}$、$\{Q\}$、$\{\neg P\}$、以上皆非。

背景觀念：蘊涵圖的建法

每個子句 $(a \vee b)$ 等價於兩條蘊涵：$\neg a \to b$ 與 $\neg b \to a$（「其中一個必須真：若 a 假則 b 真」）。蘊涵圖以全部 $2n$ 個文字為頂點、以這些蘊涵為有向邊。強連通分量 = 彼此可互相到達的最大頂點集。著名結論：公式可滿足 $\iff$ 沒有變數 X 使 X 與 $\neg X$ 落在同一個 SCC。

第一步：列出全部 **12** 條邊

子句	蘊涵一	蘊涵二
$(\neg P \vee S)$	$P \to S$	$\neg S \to \neg P$
$(P \vee \neg R)$	$\neg P \to \neg R$	$R \to P$
$(\neg Q \vee S)$	$Q \to S$	$\neg S \to \neg Q$
$(\neg Q \vee \neg S)$	$Q \to \neg S$	$S \to \neg Q$
$(\neg R \vee \neg S)$	$R \to \neg S$	$S \to \neg R$
$(R \vee \neg S)$	$\neg R \to \neg S$	$S \to R$

第二步：找環（環上的點屬於同一 **SCC**） 正文字側：$P \to S$（子句 1）、$S \to R$（子句 6）、$R \to P$（子句 2）——三條邊構成環

$$P \to S \to R \to P,$$

所以 $\{P, R, S\}$ 兩兩互達，屬於同一個 SCC。

負文字側（鏡像）：$\neg S \to \neg P$（子句 1）、$\neg P \to \neg R$（子句 2）、$\neg R \to \neg S$（子句 6）構成環 $\neg S \to \neg P \to \neg R \to \neg S$，故 $\{\neg P, \neg R, \neg S\}$ 也是一個 SCC。（這不是巧合：蘊涵圖有「斜對稱」性——把每條邊 $u \to v$ 對映成 $\neg v \to \neg u$ 仍是圖中的邊，所以 SCC 總是成對出現。）

Q 與 $\neg Q$：檢查 Q 的入邊——邊 $\cdots \to Q$ 只會來自含正文字 Q 的子句，但沒有任何子句含正的 Q，所以 Q 沒有入邊（只有出邊 $Q \to S$、$Q \to \neg S$），它自成單點 SCC $\{Q\}$。同理 $\neg Q$ 沒有出邊（只有入邊 $\neg S \to \neg Q$、$S \to \neg Q$），自成單點 SCC $\{\neg Q\}$。

第三步：逐選項判定

- $\{R\}$：× 錯誤。R 屬於較大的 SCC $\{P, R, S\}$；SCC 是「極大」集合，真子集不算。
- $\{P, R, S\}$：✓ 正確（上述環）。
- 全部七個文字一組：× 錯誤。例如 Q 無入邊，別的點到不了 Q，不可能與其他點同屬一個 SCC。
- $\{\neg Q\}$：✓ 正確（單點 SCC）。
- $\{Q\}$：✓ 正確（單點 SCC）。
- $\{\neg P\}$：× 錯誤。$\neg P$ 屬於 SCC $\{\neg P, \neg R, \neg S\}$（選項中沒有列出這一組，但這不影響 $\{\neg P\}$ 單獨成立與否的判定）。

加碼：這個公式可滿足嗎？

四個 SCC 為 $\{Q\}$、$\{\neg Q\}$、$\{P, R, S\}$、$\{\neg P, \neg R, \neg S\}$，沒有任何變數與自己的否定同組，故可滿足。例如 $P = R = S = \text{True}$、$Q = \text{False}$ 即為一組解（可逐子句驗證）。

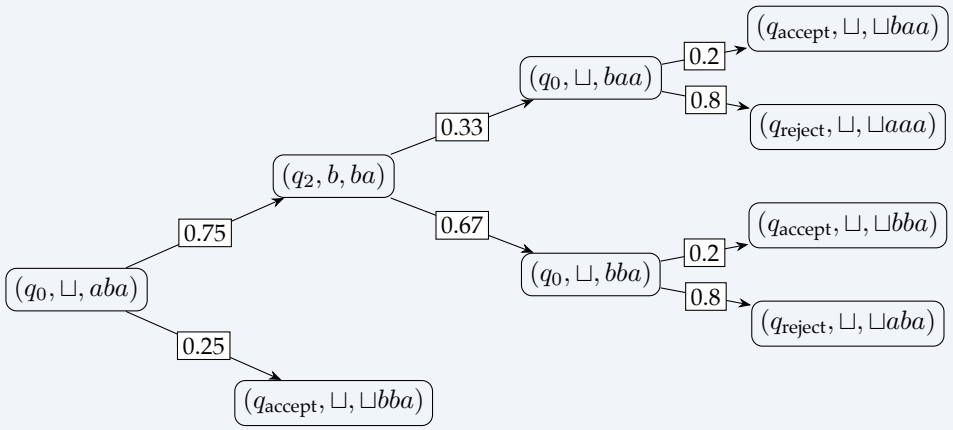
答案

$\{P, R, S\}$、$\{\neg Q\}$、$\{Q\}$。

2.8 Q8：機率圖靈機——拒絕機率與期望終止時間（填空，16 分）

題目

給定機率圖靈機 M 在輸入字 w 上的組態樹（configuration tree）：



(i) M 拒絕 w 的機率 $\text{Prob}(M(w) = 0)$ 是多少？

(ii) M 在輸入 w 上的期望終止時間 $\mathbb{E}[T_M(w)]$ 是多少？

背景觀念：怎麼讀組態樹

每個內部節點是機器的一個組態，每條邊標注該步隨機轉移的機率（同一節點的出邊機率和為 1）。一條「根到葉」的路徑就是一次可能的執行：其機率是沿路邊機率的乘積，其執行步數是路徑的邊數。

第一步：列出所有根到葉的路徑

路徑（葉節點）	機率計算	機率	步數
$q_{\text{accept}}\,(\sqcup bba$，捷徑）	0.25	0.25	1
$q_{\text{accept}}\,(\sqcup baa)$	$0.75 \times 0.33 \times 0.2$	0.0495	3
$q_{\text{reject}}\,(\sqcup aaa)$	$0.75 \times 0.33 \times 0.8$	0.198	3
$q_{\text{accept}}\,(\sqcup bba)$	$0.75 \times 0.67 \times 0.2$	0.1005	3
$q_{\text{reject}}\,(\sqcup aba)$	$0.75 \times 0.67 \times 0.8$	0.402	3
合計		1.000 ✓	

（機率總和為 1，這是很好的驗算習慣。）

（i）拒絕機率　把兩條落在 q_{reject} 的路徑機率相加：

$$\text{Prob}(M(w)=0) = 0.75\times0.33\times0.8+0.75\times0.67\times0.8 = 0.75\times0.8\times(0.33+0.67) = 0.75\times0.8 = \boxed{0.6}$$

（提出公因數 0.75×0.8 後括號恰為 1，可以少算很多。相應地接受機率 $= 0.25+0.0495+0.1005 = 0.4$，兩者和為 1 ✓。）

（ii）期望終止時間　終止時間 $T_M(w)$ = 該次執行的步數（路徑邊數）。只有兩種深度：捷徑接受路徑 1 步（機率 0.25），其餘四條路徑都是 3 步（總機率 0.75）：

$$\mathbb{E}[T_M(w)] = 1\times0.25+3\times0.75 = 0.25+2.25 = \boxed{2.5}$$

答案

（i）$\text{Prob}(M(w)=0)=0.6$；　（ii）$\mathbb{E}[T_M(w)]=2.5$ 步。

2.9 Q9：單純形法——樞軸行列與一次迭代（10 分）

題目

考慮下列單純形表（tableau）：

$T_1 =$

x	y	s_1	s_2	s_3	C	
-9	-5	1	0	0	0	9
2	-6	0	1	0	0	8
-3	10	0	0	1	0	7
-9	-7	0	0	0	1	0

（a）樞軸行（pivot column）是哪一欄？（b）樞軸列（pivot row）是哪一列？（c）執行一次單純形迭代得到 T_2，讀出指定位置的值 X（第 3 列 x 欄）、Y（目標列 y 欄）、Z（第 1 列 s_2 欄）、W（第 2 列最右端常數）。

背景觀念：單純形法一次迭代的三個步驟

1. 選樞軸行：看最底下的目標函數列，挑最負的係數所在欄（改善目標最快的方向）。
2. 選樞軸列：對樞軸行中係數為正的每一列做比值檢定（ratio test）：$\dfrac{\text{最右端常數}}{\text{樞軸行係數}}$，取比值最小的列（走最遠又不違反任何限制式）。係數 ≤ 0 的列不參加。
3. 列運算：把樞軸元素縮放成 1，再用列運算把樞軸行的其他元素全消成 0。

（a）樞軸行　目標列係數：$x:-9$、$y:-7$（其餘非負）。最負的是 -9，故樞軸行為 x 欄。

（b）樞軸列　檢查 x 欄各列係數：第 1 列 -9（負，跳過）、第 2 列 2（正 ✓）、第 3 列 -3（負，跳過）。唯一可用的是第 2 列，比值 $= 8/2 = 4$。故樞軸列為第 **2** 列，樞軸元素為 2。

（c）一次迭代的完整列運算　記三條限制列為 R_1, R_2, R_3，目標列為 R_4。

步驟 **1**：$R_2 \leftarrow R_2/2$（樞軸化為 1）：

$$R_2' = \begin{pmatrix} 1 & -3 & 0 & 0.5 & 0 & 0 & | & 4 \end{pmatrix}$$

步驟 **2**：消去其他列的 x 欄：

- $R_1 \leftarrow R_1 + 9R_2'$：$(-9{+}9,\ -5{-}27,\ 1,\ 0{+}4.5,\ 0,\ 0 \mid 9{+}36) = (0,\ -32,\ 1,\ 4.5,\ 0,\ 0 \mid 45)$
- $R_3 \leftarrow R_3 + 3R_2'$：$(-3{+}3,\ 10{-}9,\ 0,\ 0{+}1.5,\ 1,\ 0 \mid 7{+}12) = (0,\ 1,\ 0,\ 1.5,\ 1,\ 0 \mid 19)$
- $R_4 \leftarrow R_4 + 9R_2'$：$(-9{+}9,\ -7{-}27,\ 0,\ 0{+}4.5,\ 0,\ 1 \mid 0{+}36) = (0,\ -34,\ 0,\ 4.5,\ 0,\ 1 \mid 36)$

得到第二張表：

$T_2 =$

x	y	s_1	s_2	s_3	C	
0	-32	1	**4.5** (Z)	0	0	45
1	-3	0	0.5	0	0	**4** (W)
0 (X)	1	0	1.5	1	0	19
0	**-34** (Y)	0	4.5	0	1	36

對照題目模板的位置：Z 在第 1 列 s_2 欄 $= 4.5$；W 是第 2 列最右端常數 $= 4$；X 在第 3 列 x 欄 $= 0$；Y 在目標列 y 欄 $= -34$。

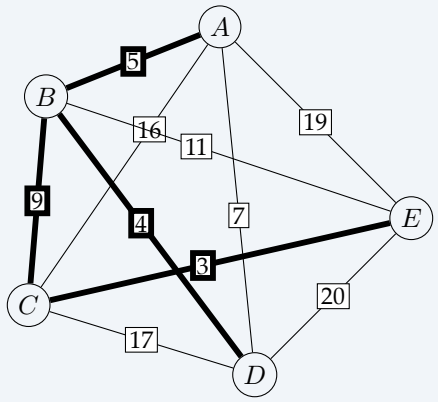
答案

（a）x 欄；（b）第 2 列；（c）$X = 0,\ Y = -34,\ Z = 4.5,\ W = 4$。

2.10 Q10：TSP 的 2-opt 演算法（作答題，15 分）

題目

考慮五個城市的 TSP 實例（下圖），邊上數字為距離，粗線為最小生成樹（**MST**）。以 MST 的前序走訪（**preorder traversal**）作為初始路線，套用 2-opt 演算法求出（局部）最佳路線。第一行寫初始路線，之後每套用一次 2-opt 交換寫一行，最後一行是（局部）最佳路線；路線以 A 開頭並回到 A。



（邊權整理：$AB=5,\ AC=16,\ AD=7,\ AE=19,\ BC=9,\ BD=4,\ BE=11,\ CD=17,\ CE=3,\ DE=20$；MST $=\{AB, BC, BD, CE\}$，總重 $5+9+4+3=21$。）

背景觀念：2-opt 交換

2-opt 是 TSP 的局部搜尋法：從目前路線中挑兩條不相鄰的邊拆掉，把中間的一段路線反轉後重新接起來（這是唯一能重新接回一條迴路的方式）。若新路線更短就採用，重複直到沒有任何 2-opt 交換能再改善——此時稱為（局部）最佳。

第一步：由 **MST** 前序走訪取得初始路線　MST 的邊為 $\{AB,\ BC,\ BD,\ CE\}$。以 A 為根畫出樹：A 的子節點是 B；B 的子節點是 C 與 D；C 的子節點是 E。深度優先前序走訪（子節點按字母序）：

$$A \ \to\ B \ \to\ C \ \to\ E\,(\text{回溯}) \ \to\ D$$

得初始路線 **A,B,C,E,D,A**，成本：

$$AB+BC+CE+ED+DA=5+9+3+20+7=44.$$

第二步：第一次 **2-opt** 交換　檢視目前路線的五條邊 $\{AB, BC, CE, ED, DA\}$，最貴的是 $ED=20$。嘗試拆掉 ED 與 AB（成本合計 25）：反轉中段 B, C, E，新路線為 A, E, C, B, D, A——同一條迴路也可寫成 **A,D,B,C,E,A**。新邊為 $AD=7$ 與 EB⋯逐邊列出：

$$AD+DB+BC+CE+EA=7+4+9+3+19=42 \quad (\text{改善}44\to 42\ \checkmark\ \text{採用})$$

（等價的看法：拆 $\{AB=5,\ ED=20\}$ 換成 $\{AE=19,\ BD=4\}$，$25\to 23$，總成本下降 2。）

第三步：第二次 **2-opt** 交換　目前路線 A, D, B, C, E, A，五條邊為 $AD=7$、$DB=4$、$BC=9$、$CE=3$、$EA=19$。最貴的是 $EA=19$。拆掉 EA 與 BC（合計 28），反轉中段 C, E，新路線 A, D, B, E, C, A——同一條迴路即 **A,C,E,B,D,A**：

$$AC+CE+EB+BD+DA=16+3+11+4+7=41 \quad (\text{改善}42\to 41\ \checkmark\ \text{採用})$$

第四步：驗證已達（局部）最佳　對路線 A, C, E, B, D, A（邊集 $\{AC, CE, EB, BD, DA\}$）檢查全部 5 種合法的 2-opt 交換：

拆掉的兩邊	得到的新路線	新成本
$AC,\ EB$	A, E, C, B, D, A	$19 + 3 + 9 + 4 + 7 = 42$
$AC,\ BD$	A, B, E, C, D, A	$5 + 11 + 3 + 17 + 7 = 43$
$CE,\ BD$	A, C, B, E, D, A	$16 + 9 + 11 + 20 + 7 = 63$
$CE,\ DA$	A, C, D, B, E, A	$16 + 17 + 4 + 11 + 19 = 67$
$EB,\ DA$	A, C, E, D, B, A	$16 + 3 + 20 + 4 + 5 = 48$

全部 $\geq 42 > 41$，沒有任何交換能改善——A, C, E, B, D, A（成本 41）是 2-opt 局部最佳。（事實上把 12 條相異迴路全列出來可驗證 41 就是全域最佳，與官方說明「此題設計成全域最佳可從任何合法初始路線在 2–3 步內到達」一致。）

答案（依題目要求逐行書寫）

`A,B,C,E,D,A`　（初始路線，成本 44）
`A,D,B,C,E,A`　（第一次 2-opt，成本 42）
`A,C,E,B,D,A`　（第二次 2-opt，成本 41，局部 = 全域最佳）

2.11 Q11：Metric TSP 是 2-approximable 是什麼意思？（簡答，10 分）

題目

說 Metric TSP（滿足三角不等式的 TSP）是 *2-approximable*，是什麼意思？

詳細解答　定義：Metric TSP 是 2-approximable，意思是——

存在一個多項式時間的近似演算法 M，對**每一個**問題實例，M 回傳的路線長度至多是該實例全域最佳解長度的**兩倍**：

$$\text{cost}(M(I)) \ \leq\ 2 \cdot \text{OPT}(I) \qquad \text{對所有實例} I.$$

作答時必須涵蓋的三個要點（官方答案特別點名）：

1. 存在性：只要求「存在」這樣一個演算法即可（不是說每個演算法都達標）。
2. 上界為兩倍最佳：回傳解 $\leq 2\times$ 全域最佳。它是保證的上界，實務上往往更好；且注意是與「最佳解」比，不是與其他演算法比。
3. 對所有輸入實例成立：這個保證是最壞情況保證，不能只對某些實例成立、也不是平均而言。

另外隱含的重要條件是多項式時間——如果允許指數時間，直接窮舉就能拿到最佳解，「近似」就沒有意義了。

加碼：為什麼 Metric TSP 做得到 2 倍？（經典 MST 演算法）

1. 求最小生成樹 MST（多項式時間）。注意 cost(MST) $\leq$ OPT：把最佳迴路拿掉一條邊就是一棵生成樹，而 MST 是最便宜的生成樹。
2. 沿 MST 做深度優先走訪，每條邊來回各走一次，得到成本 $\leq 2 \cdot$ cost(MST) 的封閉巡行。
3. 用「抄捷徑」跳過重複造訪的城市（前序走訪的順序）。三角不等式保證抄捷徑不會變長——這正是需要 Metric 假設的地方。

合計：回傳路線 $\leq 2 \cdot$ cost(MST) $\leq 2 \cdot$ OPT。（這也解釋了上一題為何用「MST 前序走訪」當 2-opt 的初始路線——它本身就已是不錯的 2-近似起點。）順帶一提：一般（非 Metric）TSP 則不存在任何常數倍近似演算法（除非 P = NP）。

答案（範例作答）

「存在一個多項式時間的近似演算法 M，使得對此問題的每一個實例，M 回傳的解之長度至多為該實例全域最佳解長度的兩倍。」——關鍵字：存在這樣的演算法、兩倍最佳解的上界、對所有可能輸入實例皆成立。

附錄：本卷重點速查表

主題	一句話重點
不可判定的例子	停機問題、圖靈機空語言問題（皆已被證明）；SAT、TSP 只是「難」，仍可判定。
不可判定 $\Rightarrow$	無窮語言、NP-hard、任何健全完備演算法必有卡住的輸入；但絕非 NP-complete。
補集的不可判定性	判定器輸出可翻轉且保留終止性，故可判定性對取補封閉。
辨識器的不對稱	$w \in L$ 終將得到 True；$w \notin L$ 可能永遠沒有答案。
Entscheidungsproblem	德文直譯「判定問題」；問謂詞邏輯公式是否恆真；不可判定。
NP 成員證明	非決定性「猜」一個解（證書），再用多項式時間「驗」。
SAT $\to$ CLIQUE	子句 $=$ 頂點群；跨群且不矛盾才連邊；$k =$ 子句數。
主定理	比較 $f(n)$ 與 $n^{\log_b a}$：小 $\Rightarrow$ 葉子主導；平 $\Rightarrow$ 乘 log；大 $\Rightarrow$ f 主導。
DPLL 優先序	單元傳播 $>$ 純文字消去 $>$ 分支；純文字消去是「整句刪除」。
2SAT 蘊涵圖	$(a \vee b) \rightsquigarrow \neg a \to b,\ \neg b \to a$；$X$ 與 $\neg X$ 同 SCC $\iff$ 不可滿足。
機率圖靈機	路徑機率 $=$ 邊機率乘積；期望時間 $= \sum$(步數 $\times$ 機率)。
單純形法	樞軸行取目標列最負；樞軸列取正係數中比值最小；列運算清空樞軸行。
2-opt	拆兩條不相鄰邊、反轉中段重接；無法再改善即局部最佳。
2-approximable	存在多項式時間演算法，對所有實例回傳 $\leq 2\cdot$OPT 的解。