Computer Science and Information Engineering
National Chi Nan University

Chapter 9
Connectivity

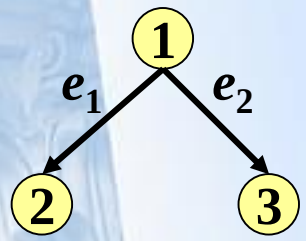
§ 9.5 Application

§ 9.5 Application

- **Technique 1**: $N = (V, E)$: digraph: $M = (a_{ij})_{|V|\times|E|}$ s.t.

  incidence matrix $a_{ij} = \begin{cases} 1, & \text{if } e_j = v_i v_i, \\ -1, & \text{if } e_j = v_i, v_i \\ 0, & \text{o.w.} \end{cases}$

- **Ex**:



$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cc} e_1 & e_2 \end{array} \\ \begin{pmatrix} 1 & 1 \\ -1 & 0 \\ 0 & -1 \end{pmatrix} \end{array}$$

§ 9.5 Application

- **Fact**: M **is totally unimodular**

  $\equiv \forall$ **square submatrix** A **of** M, $\det(A) = 0$ **or** 1 **or** -1.

  Proof.

  By induction.

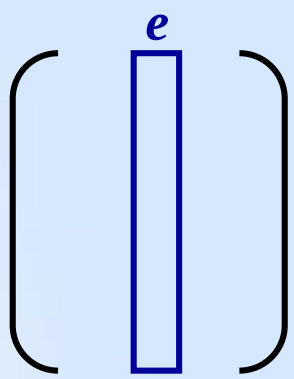
- **Technique 2**: ***s-t* path-edge matrix**: $A = (b_{ij})_{|\wp| \times |E|}$ **s.t.**

$$b_{ij} = \begin{cases} 1, \text{ if } e_j \in P_i \\ 0, \text{ o.w.} \end{cases}$$

- **Fact**: A **is totally unimodular.**

§ 9.5 Application

- **Technique 3**: Let $|\wp| = r$, $\sum_{P \in \wp, e \in P} a_P = (a_{P_1}, a_{P_2}, \ldots, a_{P_r})A_e$,

, where $A_e =$ 

means the e column vector of A.

- **Note**:

① $\sum_{P \in \wp, e \in P} a_P \leq \text{Cap}(e)$

$\Rightarrow (a_{P_1}, a_{P_2}, \ldots, a_{P_r})A \leq (\text{Cap}(e_1), \ldots, \text{Cap}(e_{|E|}))$.

§ 9.5 Application

- **Note:**

② 求 $\max_a(\sum_{P\in\wp} a_P)$

$$\Rightarrow \text{求 } \max_a(a_{P_1}, \ldots, a_{P_r}) \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} = \max_a 1 \cdot (a_{P_1}, \ldots, a_{P_r})^C.$$

③ Let $(a_{P_1}, \ldots, a_{P_r}) = X$, $(\text{Cap}(e_1), \ldots, \text{Cap}(e_r)) = C$.

求 $\max_a 1 \cdot X^C$

已知: $X \cdot A \le C$

$C \ge O$

: Linear Programming Problem 可用 F&F Algorithm 解.

$\because X \cdot A \le C \Rightarrow X \cdot A + Y \cdot I = C.$

§ 9.5 Application

- **Note:**

④ **Select suitable Y s.t.:**

求 $\max 1 \cdot X^C$

已知: $(X, Y) \cdot \begin{bmatrix} A \\ I \end{bmatrix} = C$

$X \geq O$

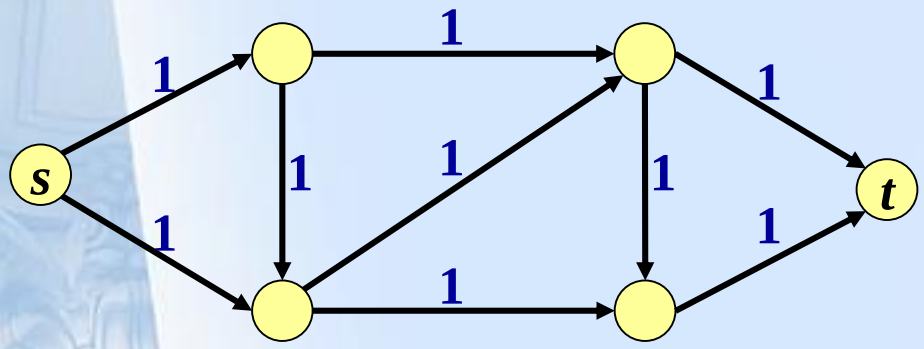
$Y \geq O$

§ 9.5 Application

- **Problem 1**: 找 max. # of edge-disjoint *s*-*t* directed paths in $G = (V, E)$: directed graph.

Sol.

ex:



Let Cap(*e*) = 1, $\forall$ *e*.

$\Rightarrow$ 由 F&F Algo. 求出 *f*. (value(*f*) = 所求)

$\Rightarrow$ 由 *f* 找出 $\{a_P\}_{P \in \wp}$: a_P = 0 or 1 (由F&F 做法, *f*(*e*) = 0 or 1)

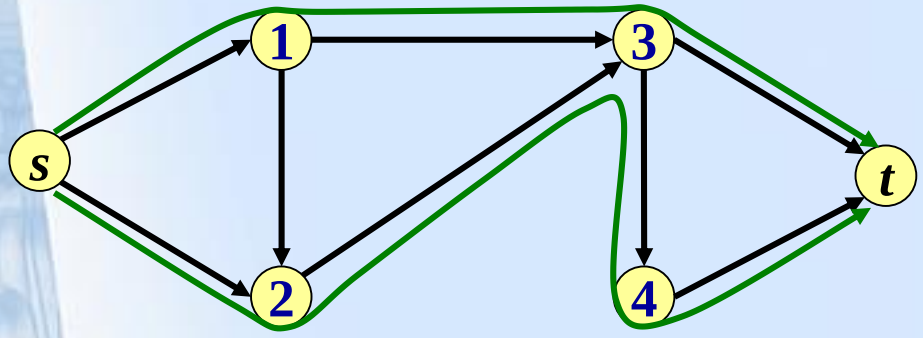
$\because \sum_{P \in \wp,\, e \in P} a_P \le \text{Cap}(e) = 1.$

§ 9.5 Application

- **Problem 2: 找 max. # of vertex-disjoint *s*-*t* directed paths in $G = (V, E)$: directed graph.**

- **Note: 不能用上法:**

ex:



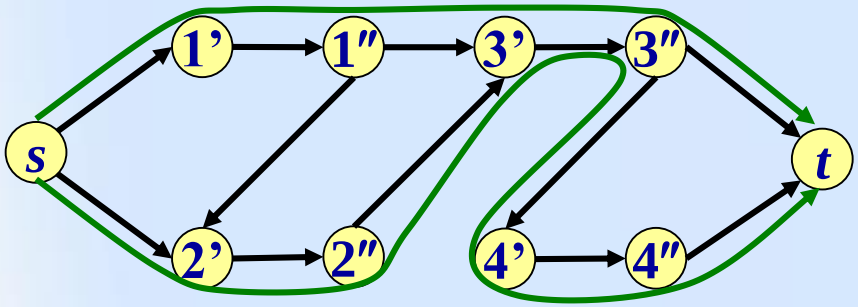
$\Rightarrow f = 2$,

but not exist 2 vertex-disjoint *s*-*t* directed paths. →←

§ 9.5 Application

- **Sol. (1/2)**

重畫一圖:



Step 1: Define $G' = (V', E')$ for given $G = (V, E)$ as:

$$V' = \{s, t\} \cup \{v', v'' : v \neq s, v \neq t, v \in V(G)\},$$

$$E' = \{sv' : sv \in E(G)\} \cup \{v''t : vt \in E(G)\} \cup$$

$$\{v_1''v_2' : v_1v_2 \in E(G) \ \forall\ e' \in E', \{v_1, v_2\} \cap \{s, t\} = \varnothing\} \cup$$

$$\{v'v'' : v \in V - \{s, t\}\}.$$

Step 2: Let $\text{Cap}(e') = 1, \forall\ e' \in E'$.

Step 3: Solve Problem 1 of G'.

§ 9.5 Application

- **Sol. (2/2)**

  Step 3: Solve Problem 1 of G'.

  $\because$ solve max. flow of G'

  $\Leftrightarrow$ Solve max. # edge-disjoint s-t di-paths in G'. ($\because$ Problem 1)

  $\Leftrightarrow$ Solve max. # vertex-disjoint s-t di-paths in G. (♦)

  Proof of (♦)

  If $\exists$ 2 di-paths is G s.t. $P_1 = s, i_1, i_2, \ldots, v, \ldots, i_r, t$

  $P_2 = s, j_1, j_2, \ldots, v, \ldots, j_m, t$

  then in G',

  $\exists$ 2 di-paths s.t. $P_1' = s, i_1', i_1'', i_2', i_2'', \ldots, v', v'', \ldots, i_r', i_r'', t$

  $P_2' = s, j_1', j_1'', j_2', j_2'', \ldots, v', v'', \ldots, j_m', j_m'', t$

  $\rightarrow\leftarrow$

§ 9.5 Application

- **Homework 3: (Due day: 12/19)**

  將Ford and Fulkerson Algorithm 實作出來，並用來解Problem 2

  Problem 2: 找 max. # of vertex-disjoint *s*-*t* directed paths in *G* = (*V*, *E*): directed graph.

§ 9.5 Application

- **Problem 3: 找 max. # of edge-disjoint *s*-*t* path**
  in $G = (V, E)$: undirected graph.

Sol.

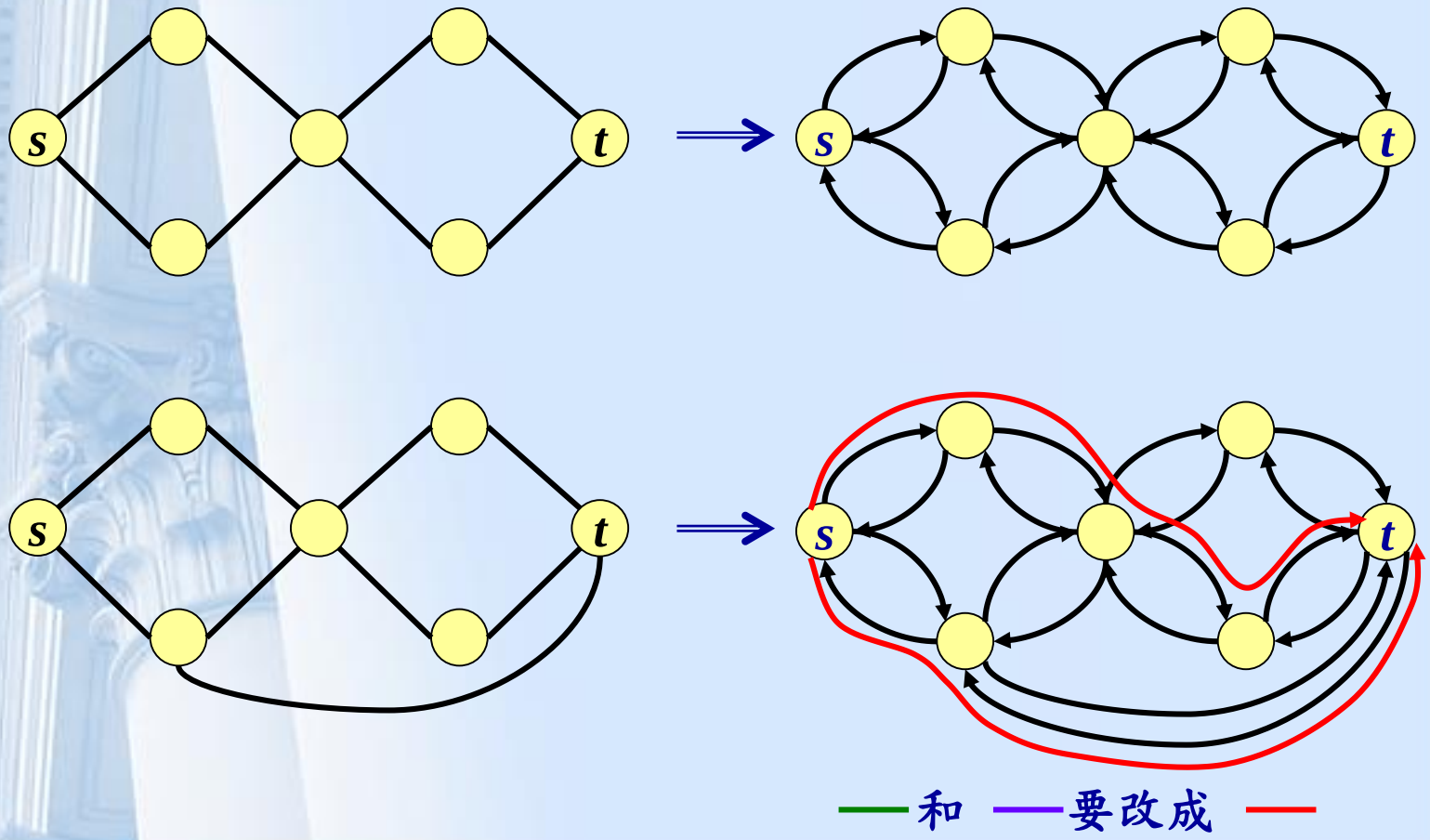
Step 1: Define $G' = (V, E')$: directed graph by:

$$E' = \{xy, yx : \{xy\} \in E(G)\}.$$

Step 2: Solve Problem 1 of G'.

§ 9.5 Application

- **ex:**



§ 9.5 Application

- **Note: Denote** $\{\{x, y\} \in E : xy \in P\} \equiv \tilde{E}(P)$, $\forall$ P **in** G**'.**

  P_1, P_2 **are 2 edge-disjoint di-paths in** G**'**

  s.t. $\tilde{E}(P_1) \cap \tilde{E}(P_2) \neq \emptyset$.

  $\Rightarrow \exists$ **2 edge-disjoint paths in** G **s.t.**

$$\begin{cases} \tilde{E}(P_1') \cap \tilde{E}(P_2') = \emptyset \\ \tilde{E}(P_1') \cup \tilde{E}(P_2') \subseteq \tilde{E}(P_1) \cup \tilde{E}(P_2) \end{cases}$$

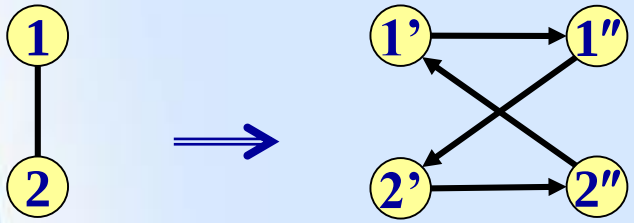
§ 9.5 Application

- **Problem 4: 找 max. # of vertex-disjoint *s*-*t* path**
  in $G = (V, E)$: undirected graph.

Sol.

結合 Problem 2 & 3:

ex:



Step 1: Define $G' = (V', E')$: directed graph by:

$$V' = \{s, t\} \cup \{v', v'' : v \in V - \{s, t\}\},$$

$$E' = \{sv', v''s : sv \in E(G)\} \cup \{v''t, tv' : tv \in E(G)\} \cup$$

$$\{v_1''v_2', v_2''v_1' : v_1v_2 \in E(G)\} \cup \{v'v'' : v \in V - \{s, t\}\}.$$

Step 2: Solve Problem 1 of G'.

Computer Science and Information Engineering
National Chi Nan University

Chapter 10
NP-completeness

§ 10.1 P, NP-problem

Chapter 10 NP-completeness

- **Book:**

◎ **Graph Algorithms**

Shimon Even © 1979

◎ **Introduction to the Theory of Computation**

Michael Sipser © 1997

§ 10.1 P, NP-problem

- **Def: nondeterministic Turing machine (NDTM)**
  deterministic Turing machine (DTM)

- **Def:**

① **decision problem**

ex: input: $G = (V, E)$ graph, $k \in N$.
output: "yes", if $\exists$ dominating set of size $\leq k$.
"no", if not.

② **optimal problem**

ex: input: $G = (V, E)$ graph.
output: $r(G) = ?$

§ 10.1 P, NP-problem

- **Ex: $S = \{i_1, i_2, \ldots, i_k\}$ given.**

  Does there $\exists\ S = S_1 \cup S_2$ s.t. $\sum_{i \in S_1} i = \sum_{j \in S_2} j$?

  Sol.

  NDTM: ① guessing stage: “猜” 一個答案 S_1

  ② checking stage: “驗證” $\sum_{i \in S_1} i = \sum_{j \notin S_1} j$

  DTM: ? (2^k) 次!

§ 10.1 P, NP-problem

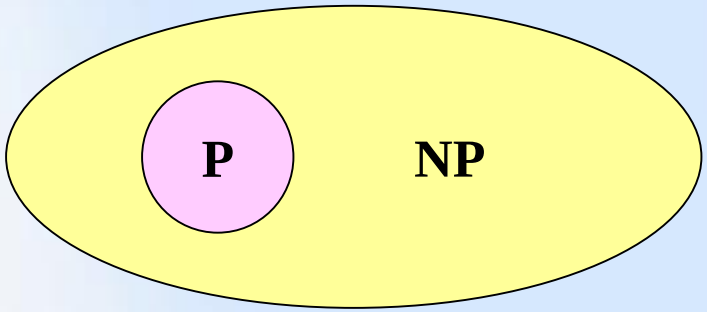
- **Def:**

  P problem: Γ problem 有 DTM 用 polynomial time 程式可解,
  則稱 $\Gamma \in$ P. (Γ is P problem)

  NP problem: Γ problem 有 NDTM 用 polynomial time 程式可解,
  則稱 $\Gamma \in$ NP. (Γ is NP problem)

  (NP is the class of languages that have poly. time verifiers)

- **Note: P $\subseteq$ NP:**



Computer Science and Information Engineering
National Chi Nan University

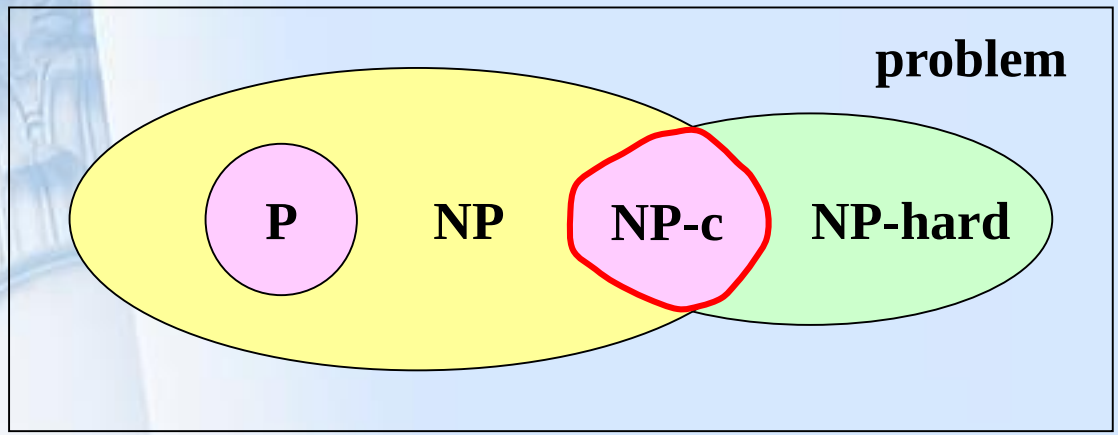
Chapter 10
NP-completeness

§ 10.2 NP-complete

§ 10.2 NP-complete

- **Def: polynomial-time reducible:**
  problem $\Gamma_1 \propto$ problem Γ_2: Γ_1 在 polynomial-time 之內交給 Γ_2

- **Def: Γ is NP-complete ≡ ① $\Gamma \in$ NP.**
  ② $\forall \Gamma' \in$ NP, $\Gamma' \propto \Gamma$. (Γ is NP-hard)



§ 10.2 NP-complete

- **Def: SAT problem (Satisfiability problem)**

  input: A boolean expression $f(x_1, x_2, \dots, x_n) = 1$.

  output: Yes, if we can given $x_1, x_2, \dots, x_n$ s.t. $f(x_1, x_2, \dots, x_n) = 1$.

  No, o.w..

- **Ex:**

① $f(x, y, z) = (x \vee y) \rightarrow (z \wedge x)$

$f(x, y, z) \Rightarrow f(1, 1, 1) = 1.$ $\Rightarrow$ **Yes!**

② $f(x, y) = x \wedge (\neg x) \wedge y$

$f(1, y) = 1 \wedge 0 \wedge y = 0$

$f(0, y) = 0 \wedge 1 \wedge y = 0$

} **always = 0** $\Rightarrow$ **No!**

§ 10.2 NP-complete

- **Theorem (Cook):**

  SAT is NP-complete.

- **Property:** $\Gamma' \propto \Gamma$ **and** $\Gamma \in \mathbf{P} \Rightarrow \Gamma' \in \mathbf{P}$.

- **Cor:** Γ^* **is NP-complete,** $\Gamma^* \in \mathbf{P} \Rightarrow \mathbf{P} = \mathbf{NP}$.

§ 10.2 NP-complete

- **Property**: Γ^* is NP-complete, and $\Gamma \in$ NP, and $\Gamma^* \propto \Gamma$

  $\Rightarrow \Gamma$ is NP-complete.

  Proof.

  $\forall \Gamma' \in$ NP, $\Gamma' \propto \Gamma^*$ (by def. of Γ^* is NP-complete)

  $\because \Gamma^* \propto \Gamma$ (已知)

  $\Rightarrow \Gamma' \propto \Gamma$ ($\because$ "$\propto$" is transitive)

  $\therefore$ By ① $\Gamma \in$ NP and ② $\forall \Gamma' \in$ NP, $\Gamma' \propto \Gamma$

  $\Rightarrow \Gamma$ is NP-complete.

- **Theorem** (Cook): 也可看成是:

  SAT $\in$ P iff P = NP.

§ 10.2 NP-complete

- **Def: A boolean-expression $f(x_1, x_2, \ldots, x_n)$ in CNF(conjunctive normal form):**

  literals; and; or

  $f(x_1, x_2, \ldots, x_n) = (x_{11} + x_{12} + \ldots + x_{1n_1}) \cdot (x_{21} + x_{22} + \ldots + x_{2n_2}) \cdot \ldots \cdot (x_{r1} + x_{r2} + \ldots + x_{rn_r})$ where $x_{ij} \in \{x_1, x_2, \ldots, x_n, \neg x_1, \neg x_2, \ldots, \neg x_n\}$.

  clause

- **Note:**

① $p \rightarrow q \Leftrightarrow \neg p \vee q$

② $p \leftrightarrow q \Leftrightarrow (\neg p \vee q) \wedge (\neg q \vee p)$

③ $p \vee (q \wedge s) = p + (q \cdot s) \Leftrightarrow (p \vee q) \wedge (p \vee s) = (p + q) \cdot (p + s)$

④ $\neg(p \wedge q) \Leftrightarrow \neg p \vee \neg q$

$\therefore f(x_1, x_2, \ldots, x_n) = \prod_{1 \le i \le r} (\sum_{j=1}^{n_i} x_{ij})$.

§ 10.2 NP-complete

- **Def:**

  SAT Problem:

  input: Given a CNF $f = \prod_{1 \le i \le r} (\sum_{j=1}^{n_i} x_{ij})$,

  where $x_{ij} \in \{x_1, x_2, \ldots, x_n, \neg x_1, \neg x_2, \ldots, \neg x_n\}$.

  output: { **"yes", if *f* is satisfiable or**
  (∃ assignment of $x_1, x_2, \ldots, x_n$ **s.t. *f* is true)**
  "no", o.w..

  3 SAT Problem:

  input: Given $f' = \prod_{1 \le i \le r} (x_{i1} + x_{i2} + x_{i3})$ **over** $\{x_1, \ldots, x_n\}$.

  output: { **"yes", if *f* ' is satisfiable**
  "no", o.w..

  (3 SAT ≡ if all the clauses have three literals.)

§ 10.2 NP-complete

- **Note: 所有的 boolean formula 皆可寫成 3 CNF-formula.**

  Proof.

  1. $x \equiv x + x + x$.

  2. $x + y \equiv x + y + x \equiv x + y + y$.

  3. $p + q \equiv (p + x) \cdot (q + (\neg x))$, x: **new variable.**

- **Ex:**

① $(x_{i1} + x_{i2} + x_{i3} + x_{i4}) = (x_{i1} + x_{i2} + y_1) \cdot (x_{i3} + x_{i4} + (\neg y_1))$

(p: $x_{i1} + x_{i2}$; q: $x_{i3} + x_{i4}$)

y_1: **new variable**

② $(x_{i1} + x_{i2} + x_{i3} + x_{i4} + x_{i5})$

$= (x_{i1} + x_{i2} + z_1) \cdot (x_{i3} + x_{i4} + x_{i5} + (\neg z_1))$

$= (x_{i1} + x_{i2} + x_{i3}) \cdot (x_{i3} + x_{i4} + z_2)) \cdot (x_{i5} + (\neg z_1) + (\neg z_2))$

where z_1, z_2: **new variables.**

§ 10.2 NP-complete

- **Note: variable 增加 poly. 倍.**

- **Theorem: 3 SAT is NP-complete.**

  Proof.

  ① 3 SAT ∈ NP

  ② SAT ∝ 3 SAT:

  Given SAT with $f = \prod_{i=1}^{r}(\sum_{j=1}^{n_i} x_{ij})$ over $\{x_1, x_2, \ldots, x_n\}$

  Consider 3 SAT with f' = Product of some $z_{k1} + z_{k2} + z_{k3}$ over $\{x_1, \ldots, x_n\} \cup \{\text{new variables}\}$

  claim: f is satisfiable $\Leftrightarrow$ f' is satisfiable

  (i) ∵ variable 增加 poly. 倍, ∴ 為 poly. time reduce.

  (ii) f' sat. $\Rightarrow$ f sat. 需證.

§ 10.2 NP-complete

- **Note: 2 SAT is polynomially solvable. (2 SAT $\in$ P)**

Computer Science and Information Engineering
National Chi Nan University

Chapter 10
NP-completeness

§ 10.3 Some NP-Complete Problems

§ 10.3 Some NP-Complete Problems

- **Def:**

  Clique Problem:

  input: Graph G and $k \in N$.

  output: $\begin{cases} \text{“yes”, if } G \text{ has a clique of size } k; \\ \text{“no”, o.w..} \end{cases}$

§ 10.3 Some NP-Complete Problems

- **Theorem: SAT $\propto$ Clique (i.e. Clique is *NP*-complete)**

  Proof. (1/4)

  Given $f = \prod_{i=1}^{r}(\sum_{j=1}^{n_i} x_{ij})$ over $\{x_1, \ldots, x_n\}$

  Consider G as: $V(G) = \{(i, j): 1 \le i \le r \text{ and } 1 \le j \le n_i\}$

  $$E(G) = \{(i, j)(i', j'): i \ne i' \text{ and } x_{ij} \ne (\neg x_{i'j'})\}$$

  Let $k = r$

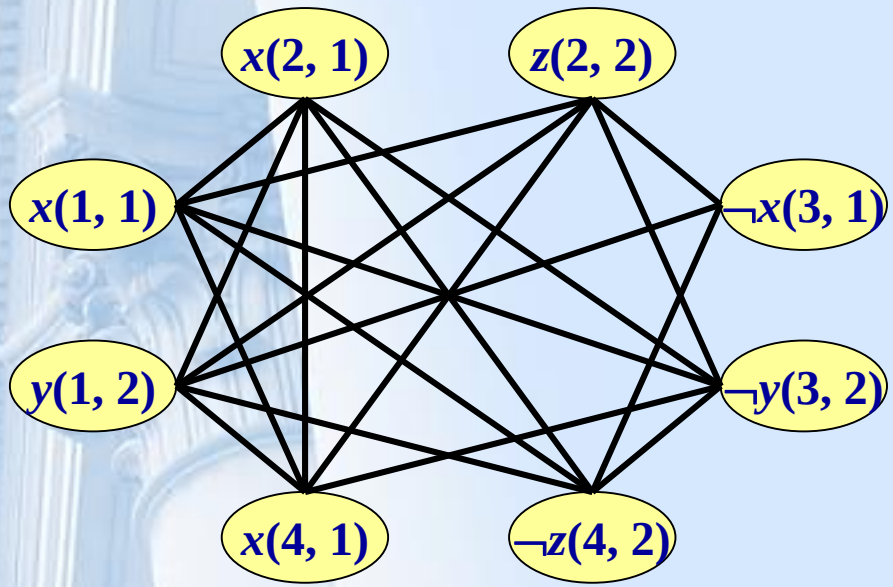
  Claim: f is satisfiable $\Leftrightarrow$ G has a clique of size k.

§ 10.3 Some NP-Complete Problems

- **Theorem: SAT ∝ Clique (i.e. Clique is *NP*-complete)**

  Proof. (2/4)

  Ex: $(x + y)(x + z)(\neg x + \neg y)(x + \neg z)$



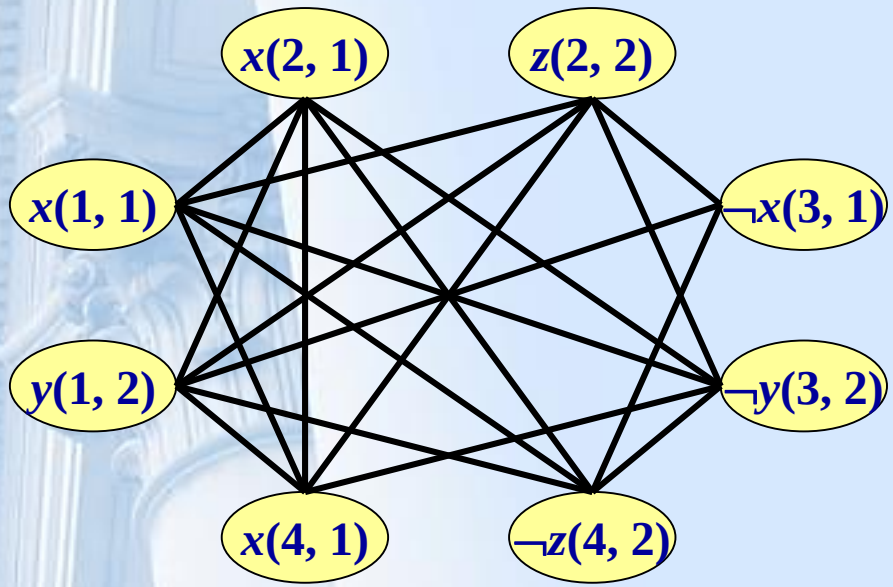
① $x = 1, \neg y = 1, z = 0$
$x = 1, \neg y = 1, z = 1$
} 皆可

§ 10.3 Some NP-Complete Problems

- **Theorem: SAT ∝ Clique (i.e. Clique is *NP*-complete)**

  Proof. (2/4)

  Ex: $(x + y)(x + z)(\neg x + \neg y)(x + \neg z)$



① $x = 1, \neg y = 1, z = 0$
$x = 1, \neg y = 1, z = 1$
} 皆可

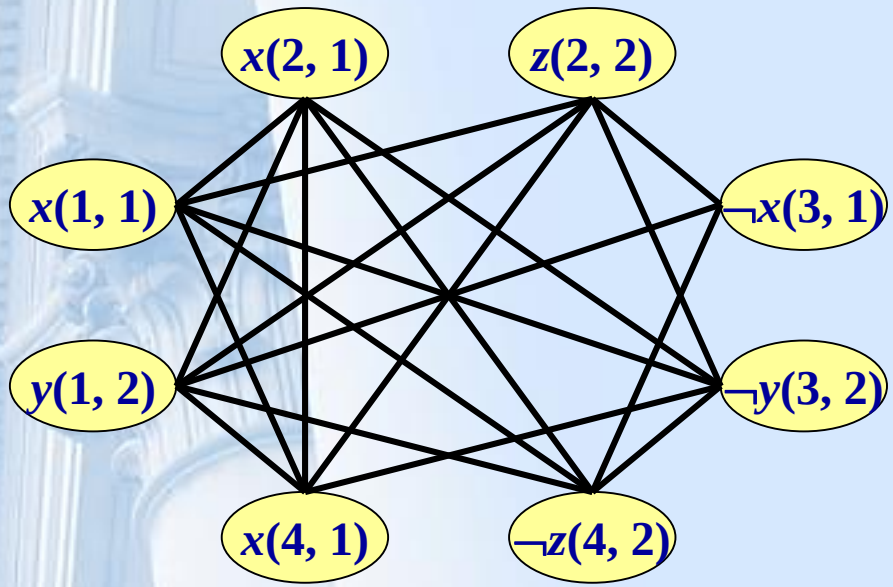
② $x = 1, \neg y = 1, z = 1$

§ 10.3 Some NP-Complete Problems

- **Theorem: SAT ∝ Clique (i.e. Clique is *NP*-complete)**

  Proof. (2/4)

  Ex: $(x + y)(x + z)(\neg x + \neg y)(x + \neg z)$



① $x = 1, \neg y = 1, z = 0$
$x = 1, \neg y = 1, z = 1$ } 皆可

② $x = 1, \neg y = 1, z = 1$

③ $x = 1, \neg y = 1, z = 0$

§ 10.3 Some NP-Complete Problems

- **Theorem: SAT $\propto$ Clique (i.e. Clique is *NP*-complete)**

  Proof. (3/4)

  Claim: f is satisfiable $\Leftrightarrow$ G has a clique of size k.

  Proof.

  ($\Rightarrow$) f is sat. $\Rightarrow$ we can assign $x_1, x_2, \ldots, x_n$ s.t. f is true.

  i.e. $\forall\, i, \exists\, j_i$ s.t. $x_{ij_i} = 1$.

  Let $C = \{(i, j_i): 1 \le i \le r\}$. Then $|C| = r = k$.

  It is impossible that $\exists\, i \neq i'$, $x_{ij_i} = \neg x_{i'j_{i'}}$

  So C is a clique (size = $k = r$)

§ 10.3 Some NP-Complete Problems

- **Theorem: SAT $\propto$ Clique (i.e. Clique is *NP*-complete)**

  Proof. (4/4)

  Claim: f is satisfiable $\Leftrightarrow$ G has a clique of size k.

  Proof.

  ($\Leftarrow$) Suppose G has a clique D of size $k = r$.

  Then $D = \{(i, p_i): 1 \le i \le r\}$ for some $p_1, p_2, \ldots, p_r$.

  We can assign $x_1, x_2, \ldots, x_n$ property

  s.t. $x_{ip_i} = 1 \; \forall \; 1 \le i \le r$ (Since $x_{ip_i} \neq \neg x_{i'p_{i'}} \; \forall \; i \neq i'$)

  $\Rightarrow$ f is sat.