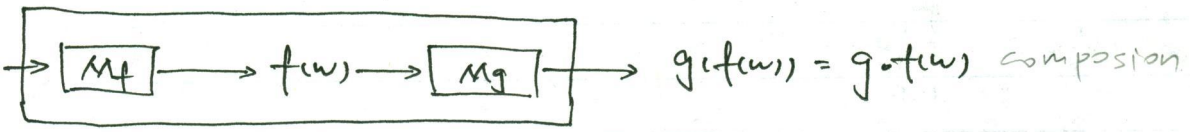
proof

(1) Just use the identity function $f(w) = w$ as the reduction

(2) Let $M_f : A \leq_p B$ and $M_g : B \leq_p C$.



$M_{g \circ f} = M_f ; M_g$

Then $M_{g \circ f} : A \leq_p C$.

The equivalence condition holds since

$w \in A \Leftrightarrow f(w) = B \Leftrightarrow g(f(w)) \in C$

By equivalence condition of f — By equivalence condition of g

That $W_{M_{g \circ f}}(n) = O(n^k)$ can be shown similarly to $W_A(n)$ in Theorem 2.

Theorem 5 If $A_1 \leq_p A_2 \leq_p A_3 \cdots \leq_p A_n$ and $A_n \in P$, then $A_i \in P$ for $\forall i : 1 \leq i \leq n-1$.

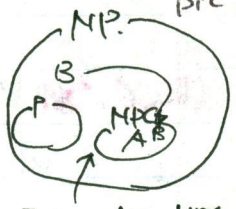
Proof By Theorem 2 and transitivity Theorem 4 (2)

Definition A language B is NP-Complete if

(1) $B \in NP$, and

(2) For all $A \in NP$, $A \leq_p B$ (NP hard condition)

widely believed pic



Intuitively, NPC is the class of most difficult NP problems.

Important **Definition** NPC = the class of all NP Complete languages.

Theorem (1) If $A \in NPC$, $B \in NP$, and $A \leq_p B$, then $B \in NPC$

proof Suppose $A \in NPC$, $A \leq_p B$, $B \in NP$

Since $A \in NPC$, for $\forall C \in NP$, $C \leq_p A$. But $A \leq_p B$,

so $C \leq_p B$ by transitivity of $\leq_p$

Hence $\forall C \in NP$, $C \leq_p B$.

But $B \in NP$, so $B \in NPC$

(2) If $\exists B \in P \cap NPC$, then $P = NP$ (unlikely)

means $(P \cap NPC \neq \emptyset)$

(3) If $\exists B \in NP - P$ (minus), then for all $A \in NPC$, $A \notin P$. (likely)

means $(NP - P \neq \emptyset)$.

Proof of (2) Suppose $B \in P \cap NPC$. Since $B \in NPC$, for $\forall C \in NP$, $C \leq_p B$

But $B \in P$. by Theorem 3, $C \in P$ we just showed $\forall C \in NP$, $C \in P$. So $NP \subseteq P$

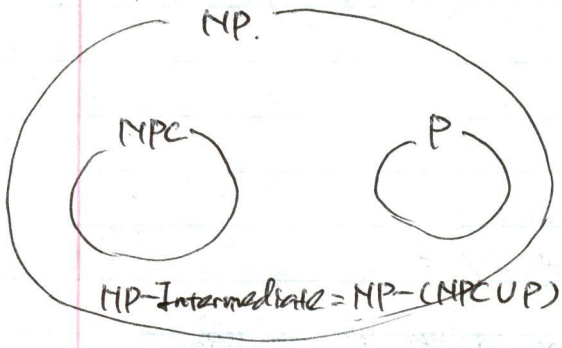
We showed ~~$NP \subseteq P$~~ already. So $P = NP$.

$P \subseteq NP$

proof(2)
proof(3)
] All NP-complete problems stand OR fall together w.r.t. existence of poly-time algorithm to solve them.

Proof of (3) suppose $B \in NP - P$. Let $A \in NPC$. Then for $\forall C \in NP$, $C \leq_p A$. Since $B \in NP$, $B \leq_p A$. But $B \notin P$ By theorem 3, $A \notin P$.

Widely believed picture.



Examples of problems believed to be in NP-Intermediate

- graph isomorphism: decide if two undirected graphs are isomorphic
- a version of integer factorization

Given integers $1 \leq M \leq N$. does ~~not~~ N have a factor d with $1 < d < M$?

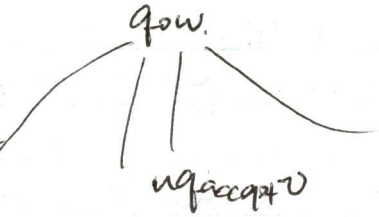
Theorem 7 (Cook-Levin) SAT $\in$ NPC (Boolean satisfiability problem.)

Proof idea $SAT \in NP$ has been shown already ✓ The proposal certificate is an assignment for the given Boolean expression. It remains to show: For $\forall A \in NP$, $A \leq_p SAT$ we will define a (poly-nomial time) reduction f s.t. $\forall w \in \Sigma^*$, $w \in A \Leftrightarrow f(w)$ is a satisfiable Boolean formula φ_w.

Since $A \in NP$, it is decided by a poly-time NTM, N and

☆ $w \in A \Leftrightarrow$ N accepts $w \Leftrightarrow$ $\exists$ accepting branch for w in the computation tree of N on w.

The formula φ_w is a "logic circuit" simulation of N's computation tree. For simplicity



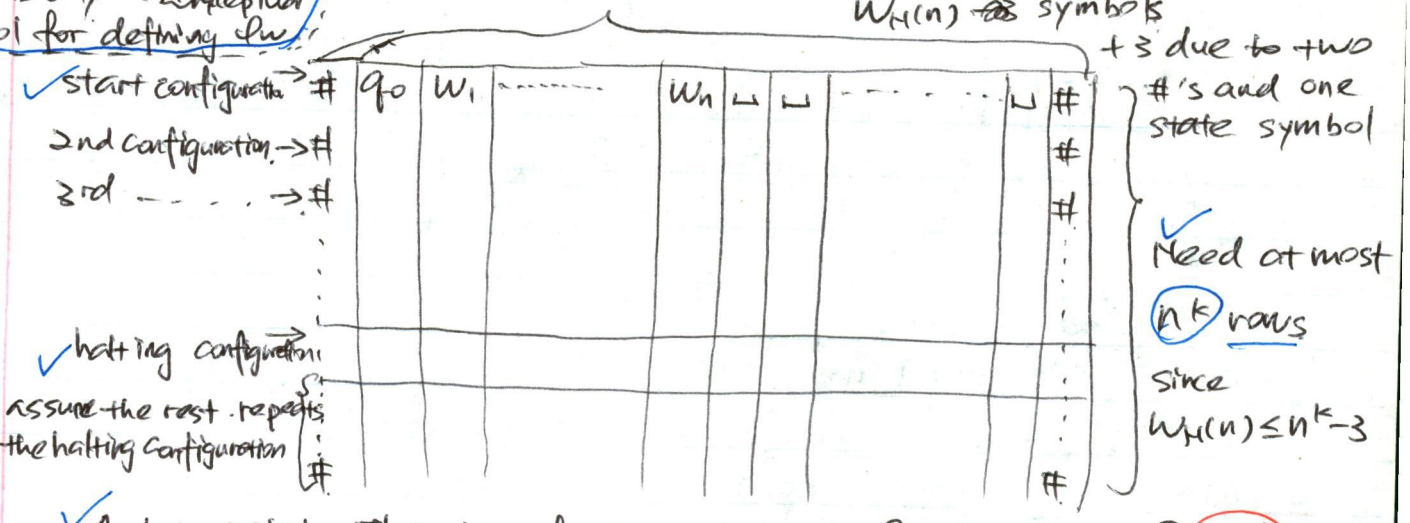
proof ↳ Assume $W_N(n) \leq n^k - 3$ for some constant $k \geq 0$

First, we represent a branch of N's computation tree on $w = w_1 w_2 \cdots w_n$ by a tableau for N on w, an $n^k \times n^k$ table in the following format:

☹ The reduction does not construct any tableau — it is only a conceptual tool for defining φ_w.

✓ The number of columns $\le W_M(n)+3 \le n^k - 3 + 3 = n^k$

n^k columns are sufficient since M can write at most $W_M(n)$ symbols



✓ Need at most n^k rows since $W_M(n) \le n^k - 3$

✓ A key point: The size of a tableau is $O(n^k \times n^k) = O(n^{2k})$, polynomial bounded.

✓ A tableau is accepting if it represents a branch leading to an accepting configuration.

☺ We denote the $n^k \times n^k$ entries by cell $[i,j]$, $1 \le i,j \le n^k$

Each cell $[i,j]$ contains a symbol in $C = Q \cup T \cup \{\#\}$ where Q is the set of control states of M, T is the tape alphabet of M.

($|C|$ is a constant determined by M, independent of $n = |w|$)

Description of φ_w

The variables of φ_w are $\{x_{i,j,s} \mid 1 \le i,j \le n^k, s \in C\}$.

The interpretation is: $x_{i,j,s} = 1 \iff$ cell $[i,j]$ has symbol s.

$\varphi_w = \varphi_{cell} \wedge \varphi_{start} \wedge \varphi_{move} \wedge \varphi_{accept}$.

✓ φ_{cell}: ensures each cell $[i,j]$ has exactly one symbol.

φ_{start}: ensures 1st row represents start configuration.

φ_{move}: ensures that $(i+1)^{th}$ row follows from i^{th} row according to M's transition function.

φ_{accept}: ensures that the tableau contains q_{accept} somewhere.

$$\varphi_{cell} = \bigwedge_{1 \le i,j \le n^k} [(\text{cell}[i,j] \text{ has at least one symbol in } C) \wedge (\text{cell}[i,j] \text{ does not have two distinct symbols in } C)]$$

$$= \bigwedge_{1 \le i,j \le n^k} \Big[\Big(\bigvee_{s \in C} x_{i,j,s} \Big) \wedge \Big(\bigwedge_{\substack{s,t \in C \\ s \ne t}} (\overline{x_{i,j,s}} \wedge \overline{x_{i,j,t}}) \Big) \Big]$$

$$= \bigwedge_{1 \le i,j \le n^k} \Big[\Big(\bigvee_{s \in C} x_{i,j,s} \Big) \wedge \Big(\bigwedge_{s,t \in C, s \ne t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \Big) \Big]$$

($\bigvee_{s \in C}$: constant # of literals; $\bigwedge_{s,t \in C, s \ne t}$: constant # of literals)

The size of φ_{cell} = the total # of literals $= c \times n^k \times n^k = O(n^{2k})$

$\varphi_{start} = X_{1,1,\#} \wedge X_{1,2,q_0} \wedge X_{1,3,w} \wedge \cdots \wedge X_{1,n+2,w_n} \wedge X_{1,n+3,\sqcup} \wedge \cdots \wedge X_{1,n^k-1,\sqcup} \wedge X_{1,n^k,\#}$

The size of $\varphi_{start} = n^k = O(n^k)$

$\varphi_{accept} = \bigvee_{1 \le i,j \le n^k} X_{i,j,q_{accept}}$

The size of $\varphi_{accept} = n^k \times n^k = O(n^{2k})$

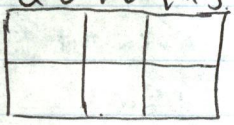
10/20 補上.

φ_{move} ensures that $(i+1)$-th row follows from i-th row by N's δ

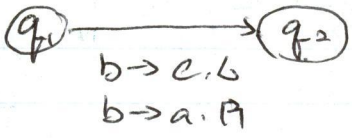
key observation: legal transitions only change at most 3 columns involving a state symbol.

$\# a a a b a b \cdots a b a b q_i a b c a \cdots \sqcup \sqcup \sqcup \cdots \sqcup \#$

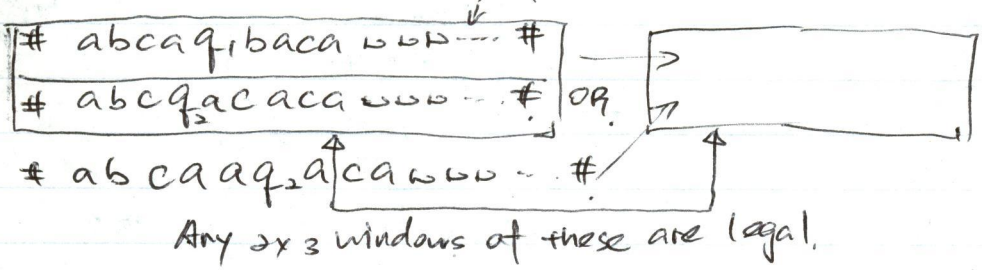
✓ A window is any 2×3 contiguous cells containing symbols in $C = Q \cup T \cup \{\#\}$



✓ A window is legal if it is a portion of some two consecutive rows representing a_1 and a_2. $a_1 \vdash a_2$



suppose to have more.



The special legal windows are any portions of two halting configurations

S_1	q_{accept}	S_2
S_1	q_{accept}	S_2

etc

∴ the tableau repeats the halting configurations to the end.

Examples of illegal windows.

Illegal format

b	q_1	b
q_2	c	q_2

#	#	a
q	b	#

etc.

a	b	a
a	a	a

The control is not reading b, and cannot be changed to a.

a	q_1	b
q_1	a	a

$\delta(q_1, b) = (q_1, a, b)$ No such transition.

Claim 1 For any legal window

s_1	s_2	s_3
t_1	t_2	t_3

where s_1, s_2, s_3 are not state symbols, $s_2 = t_2$.

proof Sufficient to consider the cases where s_1 or s_3 is next to a state symbol

q	s_1	s_2	s_3	q
	t_1	t_2	t_3	

In all such cases $s_2 = t_2$.

As defined for Turing machine M so has exactly one state symbol.

Claim 2 If the i-th row represents a configuration C_i other than the halting configuration and every window over the i-th and (i+1)-th rows are legal, then the (i+1)-th row represents a configuration C_{i+1} s.t. $C_i \vdash C_{i+1}$

proof In the i-th row, consider any consecutive symbols $s_1 s_2 s_3$ not including a state symbol.

By claim 1, s_2 moves/changes in the (i+1)-th row. So all position of the i-th row, possibly except the position with the state symbol, q, and the adjacent positions, s, t, have the same symbols in the (i+1)-th row.

C_i	#	A	s	q	t	B	#
C_{i+1}	#	A				B	#

This is the reason for using # — to apply claim 1 uniformly

Now the window

s	q	t

is legal, it must be a portion of two configurations $D_i \equiv D_{i+1}$ s.t. $D_i \vdash D_{i+1}$.

D_i, D_{i+1} may be different from C_i, C_{i+1}

identical with a window in $C_i C_{i+1}$

D_i	#	E		S	q	t.	F.	#
D_{i+1}	#	E.					F.	#

Now replace E by A. F by B. and we get C_i, C_{i+1} s.t $C_i \vdash C_{i+1}$

Define (i,j)-window to be

	$j-1$	j	$j+1$
i	a_1	a_2	a_3
$i+1$	a_4	a_5	a_6

where $a_1, \dots a_6$ are symbols in C

asserts ALL windows in the tableau are legal

$\varphi_{move} = \bigwedge_{1 \le i < n^k, 1 < j < n^k}$ legal_window (i,j)

→ asserts (i,j)-window is legal

where legal_window$(i,j) = \bigvee$ for all $a_1 \dots a_6$ forming a legal (i,j)-window $(X_{i,j-1,a_1} \wedge X_{i,j,a_2} \wedge X_{i,j+1,a_3} \wedge X_{i+1,j-1,a_4} \wedge X_{i+1,j,a_5} \wedge X_{i+1,j+1,a_6})$

size of φ_{move} $O(n^k \times n^k) = O(n^{2k})$

All such legal windows can be enumerated by inspection of M's. Q, T, δ.

It is independent of $n = |w|$. Hence the size of legal-window(i,j) is a constant.

claim 3 If the 1st row represents the start configuration on input w and every window in the tableau is legal, the tableau represents a valid computation branch on w.

If in addition q_{accept} appears in the tableau, the tableau represents an accepting branch

Proof By claim 2 and properties of tableaus.

This justifies the def of φ_{move}.

The size of φ_w is thus size(φ_{cell}) + size(φ_{start}) + size(φ_{move}) + size(φ_{accept})

$= O(n^{2k}) + O(n^k) + O(n^{2k}) + O(n^{2k})$

$= O(n^{2k})$

Hence φ_w can be constructed in polynomial time.

claim 4

satisfiable means if it is possible to find an interpretation (model) makes the formula true.

$X_{i,j,s} = 1 \Leftrightarrow$ cell $[i,j]$ contains S

$\Rightarrow$,

φ_w is satisfiable $\Leftrightarrow \exists$ accepting tableau on w.

proof ($\Rightarrow$) Suppose φ_w is satisfied by an assignment α.
True φ_{cell} determines the symbols in all cell $[i,j]$, hence determines a certain tableau T. True φ_{start} ensures T has the start config. in the 1st row. True φ_{move} ensures all windows T are legal. So T represents a valid branch on w. True φ_{accept} ensures q_{accept} appears in T. So T represents an accepting branch on w. By Claim 3 applied to T.

($\Leftarrow$) Suppose T is an accepting tableau on w. φ_{cell} obviously satisfied by an assignment defined by $X_{i,j,s}=1 \Leftrightarrow$ cell $[i,j]$ contains S.
φ_{start} obviously satisfied. T represents a valid branch, so all windows of T are legal, satisfying φ_{move}. φ_{accept} is obviously satisfied.

$\bigvee (X_{i,j-1,a_1} \wedge \cdots \wedge)$ forall

φ_w is satisfiable.

Now, $\exists$ accepting tableau on $w \Leftrightarrow M$ accepts $w \Leftrightarrow w \in A$, proving the equivalence condition of the reduction.
This concludes Theorem 7 ! ☆

in miscellous Notes

10/25 3 conjunctive normal form 3-cnf.
$(l_1^1 \vee l_1^2 \vee l_1^3) \wedge (l_2^1 \vee l_2^2 \vee l_2^3) \wedge \cdots \wedge (l_n^1 \vee l_n^2 \vee l_n^3)$
3 SAT is the satisfiability problem for 3-cnf. Boolean formulas.
A general cnf can have any # of literals in clause.

Theorem 8: 3SAT $\in$ NPC

proof It is sufficient to convert $\varphi_w = \varphi_{cell} \wedge \varphi_{start} \wedge \varphi_{move} \wedge \varphi_{accept}$ to a 3-cnf $\varphi_w^{3\text{-cnf}}$ s.t. φ_w is satisfiable $\Leftrightarrow \varphi_w^{3\text{-cnf}}$ is satisfiable and show that $\varphi_w^{3\text{-cnf}}$ can still be constructed in $O(n^{3k})$ time.
φ_{cell}, φ_{start}, φ_{accept} are already in conjunctive normal form (cnf).
$\varphi_{move} = \bigwedge_{1 \le i < n^k,\ 1 < j < n^k}$ legal-window (i,j).

legal-window $(i,j) \Leftrightarrow \bigvee_{\text{forall } a_1,\cdots a_6 \text{ forming a legal window}} (X_{i,j-1,a_1} \wedge \cdots \wedge X_{i+1,j+1,a_6})$, which is in disjunction of conjunction. 6 literals

	j-1	j	j+1
i	a_1	a_2	a_3
i+1	a_4	a_5	a_6

which is in disjunctive normal form. (dnf)

例3 fact 1

Apply Fact1 to legal-window (i,j) and get an equivalent cnf whose size is still constant (although a larger constant). Apply this to all legal-window (i,j)'s in φ_{move}, and let φ_{move}^{cnf} be the resulting cnf equivalent to φ_{move}

Since size$(\varphi_{move}) = O(n^{2k})$.

So size(φ_{move}^{cnf}) is still $O(n^{2k})$

if3fact2

Apply Fact2 to φ_w^{cnf} and let $\varphi_w^{3\text{-}cnf}$ be the resulting formula such that φ_w^{cnf} is satisfiable $\Leftrightarrow$ $\varphi_w^{3\text{-}cnf}$ is satisfiable

Each clause's size in φ_w^{cnf} increases about 3 times. So size$(\varphi_w^{3\text{-}cnf})$ is about 3 times size$(\varphi_w^{cnf}) = O(n^{2k})$

So size$(\varphi_w^{3\text{-}cnf}) = O(n^{2k})$. This concludes Theorem 8. ∎

✓

Theorem 9 3SAT $\leq_p$ VERTEX-COVER

Proof Let $\varphi = C_1 \wedge \cdots \wedge C_n$,

$C_i = l_i^1 \vee l_i^2 \vee l_i^3$, be any 3-cnf

We will define a poly-time reduction $f: \varphi \mapsto (G_\varphi, k)$ s.t. φ is satisfiable $\Leftrightarrow$ G_φ has a k-vertex cover and G_φ and k can be generated in poly time of size$(\varphi) = 3n$ = # of literals

Let $x_1, \ldots, x_m$ be the variables of φ. The reduction generates G_φ consisting of m variable pairs, n clause triples and cross edges between var pairs and clause triples.

Also, set $k = m + 2n$

Input: an undirected Graph G and an integer $k \geq 1$

Decide: if G has a k-vertex cover

A vertex cover: containing k vertices

A set of vertices covering all edges of G

SAT∈NPC

3SAT∈NPC

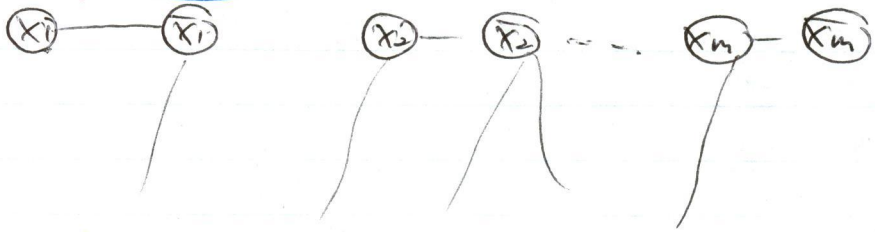
By theorem 8 (1)

if $A_1 \leq_p A_2 \leq_p \cdots \leq_p A_k$,

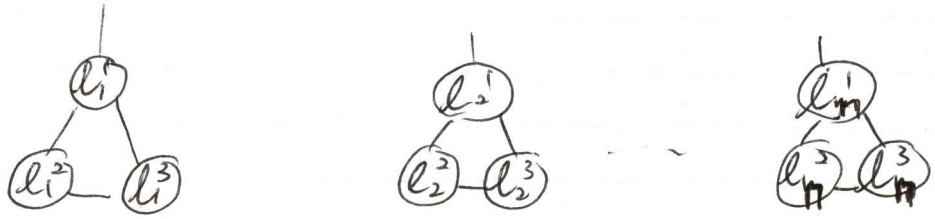
$A_1 \in NPC$

$A_2 \in NP$ $A_k \in NP$

Then $A_2 \in NPC$ ⋯ $A_k \in NPC$



create across edge between a vertex in a VAR pair, and a vertex in a clause triple labeled by the same literal



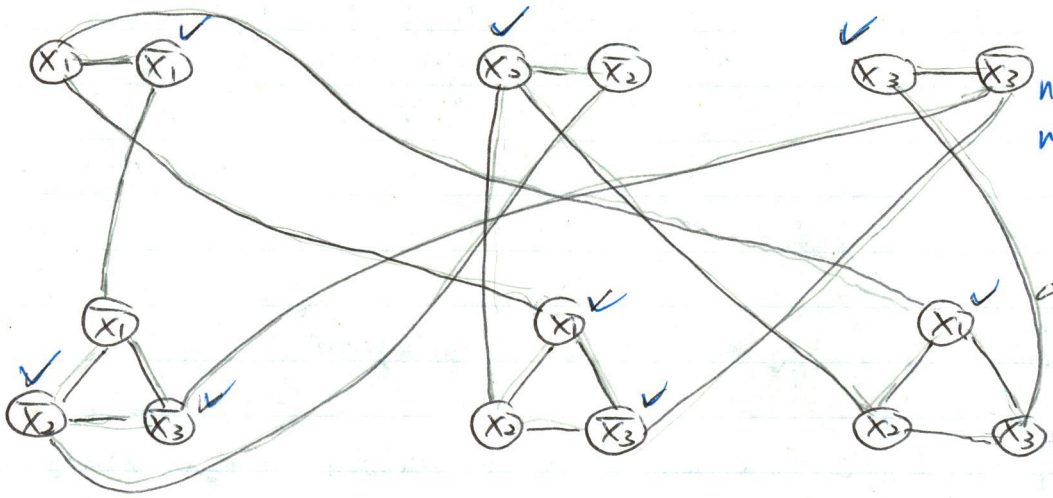
Mid-term ups to 3SAT. theorem

including: Cook-Levin Theorem: 3SAT ∈ NPC.

SAT ∈ NPC.

e.g. significance of nk×nk tableau. $\varphi_w = \varphi_{cell} \wedge \varphi_{start} \wedge \varphi_{move} \wedge \varphi_{accept}$.

Example: $\varphi = (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (x_1 \vee x_2 \vee \bar{x}_3) \wedge (x_1 \vee x_2 \vee x_3)$



eg: (同理例子)

n = # of clauses = 3

m = # of VARs = 3

k = m + 2n = 3 + 2×3 = 9

are close to.

这里也对应选 x3.

去掉 x1, x2.

VAR triple.

$k = m + 2n = 3 + 2\times 3 = 9$

3·size(φ) ≤ 3n+6n = 9n

证明: # of vertices in G_φ = 2m+3n

# of edges = m + 3n + 3n = m + 6n

# of VARs ≤ # of literals, so m ≤ 3n.

# of vertices = 2m+3n ≤ 3n+3n = 6n

= 3·(3n) = 3·size(φ)

So, G_φ can be generated in linear time of size(φ).

defined before.

10/17. continue.

回顾上次的 example

n = # of clauses = 3

m = # of vars = 3

Proof of equivalence condition: φ is satisfiable ⇔ G_φ has a k-vertex cover

(⇒) Suppose φ is satisfied by an assignment A. A makes exactly one of x_i — $\bar{x}_i$ true, $1 \le i \le m$. Let $L_1, \dots L_m$ be the true nodes in the m var pairs.

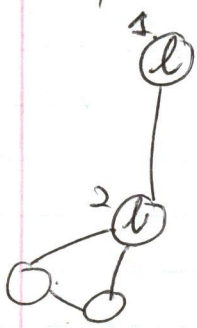
A makes each clause C_i, $1 \le i \le n$, true. In each clause triple for C_i, select a true node. select a true node from the triple, let the other two be l_i', l_i''

Let VC = $\{L_1, \dots L_m, l_1', l_1'', l_2', l_2'', l_n', l_n''\}$ containing m+2n nodes

We show VC is a vertex cover. $L_1, \dots L_m$ cover all connecting edges in the VAR pairs. x_i — $\bar{x}_i$

The nodes l_i' and l_i'' in clause triple for C_i cover the 3 edges in the triple

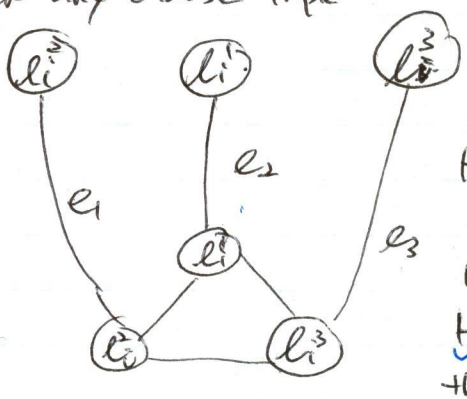
Consider any cross edge.



By def. of G_Φ, the end nodes have the same literal.
If ℓ is true under A, the node ① must be one of L_1-L_m and is covered by it
If ℓ is false under A, the node ② must be one of the two nodes selected in that triple and is covered by it. (l_i, l_i', l_i'')

Example $A = \{x_1=0, x_2=1, x_3=1\}$ (前页图上打勾).

(⇐) Suppose VC is a vertex cover containing $m+2n$ nodes. The connecting edge x_i–$\bar{x}_i$ can only be covered by x_i or $\bar{x}_i$.
The clause triple edges can only be covered by at least 2 nodes in the triple. Thus, VC must contain m nodes from the var pairs, one from each pair, and $2n$ nodes from the n clause triples, 2 from each triple.
Let VC be $\{L_1 \cdots L_m, l_1' l_1'' \cdots l_n' l_n''\}$
(from var pairs; from triples)
Let A be the assignment that makes $L_1 \cdots L_m$ true.
We show A satisfies Φ
Consider any clause triple



Exactly 2 nodes in this triple is VC.
Covering the two incident e_j edges
Hence at least one e_j must be covered by the end node in a VAR pair.,
which is in VC, which is made true by A.
Hence, the other end node labeled by the same literal, is true, making that clause true. Thus every clause has a true literal under A.

All NP-complete problems are polynomially equivalent.
For any A, B ∈ NPC. $A \leq_p B$ and $B \leq_p A$.

Theorem 3SAT $\leq_p$ SUBSET-SUM. (#)

Input: a multi-set $S = \{t_1, \ldots, t_n\}$ of integers t_i and an integer t. S may contain multiple copies of the same integer.

Decide if S has a sub-multi-set that sum to t.

$\varphi = (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (x_1 \vee x_2 \vee \bar{x}_3) \wedge (x_1 \vee \bar{x}_2 \vee x_3)$

(C_1, C_2, C_3)

	var control 1	2	3	C_1	C_2	C_3
x_1	1	0	0	0	1	1
$\bar{x}_1$	1	0	0	1	0	0
x_2		1	0	0	1	1
$\bar{x}_2$		1	0	1	0	0
x_3			1	0	0	1
$\bar{x}_3$			1	1	1	0
padding integers g_1				1	0	0
h_1				1	0	0
g_2					1	0
h_2					1	0
g_3						1
h_3						1
t:	1	1	1	3	3	3

All numbers are decimal.

tells which literal appears in which clause

# of literals

to variable clause number 无关

Mid. Understand proof ideas for big proofs

eg. If $A \leq_p B$ and $B \in P$, $A \in P$.

- ✓ $SAT \leq_p CLIQUE$.
- ✓ Theorem V to M [w | c] — tell the difference, also proof.
  Theorem M to V [w | #1 c]
- ✓ Definition of DTM. NTM. Configuration.
- ✓ Simulations. How to simulate multi-tape DTM by one step DTM.
- ✓ PAM $\Leftrightarrow$ DTM
- ✓ 10 Questions

Page on the Book.

Theorem $3SAT \leq_p SUBSET\text{-}SUM$.

Let $\varphi = c_1 \wedge \dots \wedge c_k$ be any 3-cnf-formula, and let the vars of φ be $x_1, \dots x_\ell$. The reduction produces $S = \{x_1, \bar{x}_1, \dots x_\ell, \bar{x}_\ell, g_1, h_1, \dots, g_k, h_k\}$ containing $2k+2\ell$ integers and $t = \underbrace{11\dots1}_{\ell}\underbrace{33\dots3}_{k}$ arranged in the following sum table.

	1 2 $\dots$ ℓ	c_1 c_2 $\dots$ c_k
x_1	$1\ 0\ \dots\ 0$	$[x_i, c_j] = 1$ iff x_i occurs in c_j
$\bar{x}_1$	$1\ 0\ \dots\ 0$	$[\bar{x}_i, c_j] = 1$ iff $\bar{x}_i$ $\dots$
x_2	$1\ 0\ \dots\ 0$	$1 \le i \le \ell$, $1 \le j \le k$.
$\bar{x}_2$	$1\ 0\ \dots\ 0$	all others $= 0$.
$\vdots$	$\vdots$	Each c_j column has at most three 1's
x_ℓ	$[x_i, i] = [\bar{x}_i, i] = 1$	
$\bar{x}_\ell$	$1 \le i \le \ell$. all other $= 0$.	
g_1		$1\ 0\ \dots\ 0$
h_1		$1\ 0\ \dots\ 0$
g_2		$1\ 0\ \dots\ 0$
h_2	empty	$1\ 0\ \dots\ 0$
$\vdots$		$\vdots$
		$[g_i, c_i] = [h_i, c_i] = 1$
g_k		$1 \le i \le k$
h_k		all others $= 0$.
t	$1\ 1\ \dots\ 1$	$3\ 3\ \dots\ 3$

In the book, y_i z_i are used for x_i $\bar{x}_i$

# of vars $= \ell \le$ # of literals $3k$

Example. $\varphi = (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (x_1 \vee x_2 \vee \bar{x}_3) \wedge (x_1 \vee x_2 \vee x_3)$

	1	2	3	c_1	c_2	c_3	k.
x_1	1	0	0	0	1	1	
$\bar{x}_1$	1	0	0	1	0	0	
x_2		1	0	0	1	1	
$\bar{x}_2$		1	0	1	0	0	
x_3			1	0	0	1	
$\bar{x}_3$			1	1	1	0	
g_1				1	0	0	
h_1				1	0	0	
g_2					1	0	
h_2					1	0	
g_3						1	
h_3						1	
	1	1	1	3	3	3	

2ℓ rows; $2k$ rows.

The total # of digits in the table $\le$

$(2\ell + 2k + 1)(\ell + k) \le$

$(6k + 2k + 1)(3k + 4k)$

$= (8k+1)(7k+4)$

$= 56k^2 + 4k$

$= 32\left(\frac{n}{3}\right)^2 + 4\left(\frac{n}{3}\right) = O(n^2)$

Proof

Equivalence condition: φ is satisfiable $\Leftrightarrow$ S has a subset summing to t.

($\Rightarrow$) Suppose φ is satisfied by assignment A.

Notation: $A(x_i) = 0$ or 1. $A(\bar{x}_i) = 0$ or 1

Let $S' = \{x_i \mid A(x_i) = 1\} \cup \{\bar{x}_i \mid A(\bar{x}_i) = 1\}$

Since S' includes exactly one of x_i or $\bar{x}_i$ for each $1 \le i \le \ell$, the initial ℓ digits of the integer in S' sum to the initial ℓ digits of $t = 11\cdots1\cdots$ (k 1's)

Consider any C_j column, $1 \le j \le k$, restricted to rows $x_1, \bar{x}_1, \cdots x_\ell, \bar{x}_\ell$ (upper right segment). Since C_j has exactly 3 literals, C_j column has at most three 1's in this segment. Since A satisfies φ, $A(x_i) = 1$, or $A(\bar{x}_i) = 1$ for at least one x_i or $\bar{x}_i$ occurring in C_j.

If $A(x_i) = 1$, $x_i \in S'$, and $[x_i, C_j] = 1$ since x_i occurs in C_j.

If $A(\bar{x}_i) = 1$, $\bar{x}_i \in S'$, and $[\bar{x}_i, C_j] = 1$ since $\bar{x}_i$ occurs in C_j.

So, the C_j column has at least one 1 in this segment.

If the number of 1 = 1, add g_j and h_j in S'. Since $[g_j, C_j] = [h_j, C_j] = 1$, the sum of C_j column = 3 in the entire column.

If the number of 1 = 2, add g_j in S', the sum of C_j column = 3

If the number of 1 = 3, add g_j or h_j. The resulting S' row add to t.

$A(\bar{x}_1) = 1$	$A(x_1) = 1$
$A(x_2) = 1$	$A(\bar{x}_2) = 1$
$A(x_3) = 1$	$A(\bar{x}_3) = 1$
$S' = \{\bar{x}_1, x_2, x_3\}$	$S' = \{x_1, \bar{x}_2, \bar{x}_3\}$
$S' = \{\bar{x}_1, x_2, x_3, g_1, h_1, g_2, h_2, g_3\}$	$S' = \{x_1, \bar{x}_2, \bar{x}_3, g_1, g_2, g_3, h_3\}$

($\Leftarrow$) Suppose $S' \subseteq S$ add to $t = 11\cdots1\,33\cdots3$ (k 3's)

From the def. of the upper left segment, S' must have exactly one of x_i or $\bar{x}_i$ for each $1 \le i \le \ell$ in order to add to 1.

Let A be the assignment defined by:

$A(x_i) = 1$ if $x_i \in S'$

$A(\bar{x}_i) = 1$ if $\bar{x}_i \in S'$

We show A satisfies φ. Consider any column C_j, $1 \le j \le k$, limited to the integers in S'

The digits in this column add up to 3.

	C_j
x_i	1
$\bar{x}_i$	OR
	1
	⋮

But at most 2 can come from g_j and/or h_j.

So at least one 1 must be in the upper right segment. Let x_i or $\bar{x}_i$ be the number in S' that has this 1.

So $[x_i, C_j] = 1$ OR $[\bar{x}_i, C_j] = 1$

If $[x_i, C_j] = 1$, $A(x_i) = 1$, and x_i occurs in C_j. So C_j is satisfied by A

If $[\bar{x}_i, C_j] = 1$, $A(\bar{x}_i) = 1$, and $\bar{x}_i$ So

Hence, φ is satisfied by A.

back to [3] eg.

red.mark $S' = \{x_1, \bar{x}_2, \bar{x}_3, g_1, g_2, g_3, h_3\}$ add to t.

$A(x_1) = 1$

$A(\bar{x}_2) = 1$

$A(\bar{x}_3) = 1$

Theorem 3SAT $\leq_p$ HAMPATH

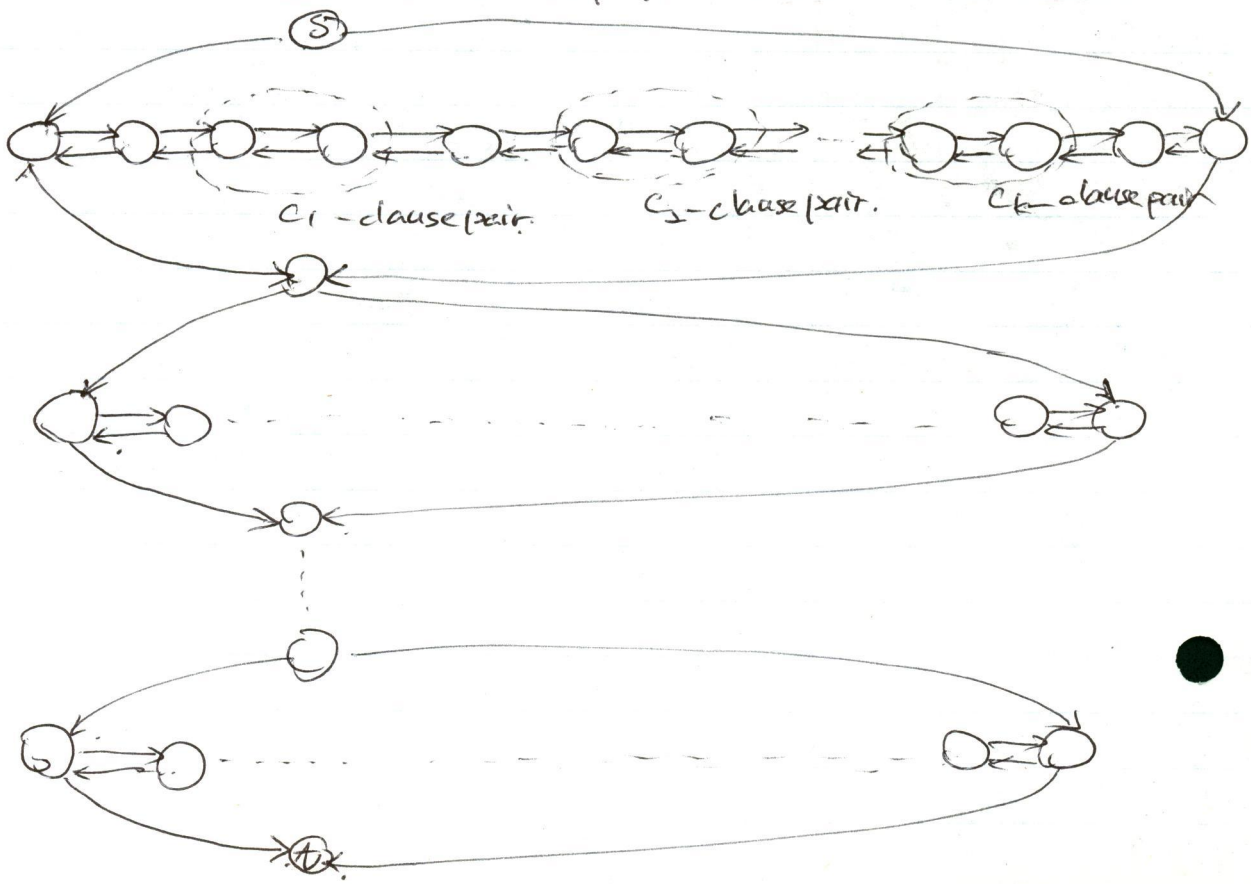
Input: directed Graph G, s, t. nodes

decide if G has a HAMpath from s to t

visit each node exactly one time

$\varphi = C_1 \wedge C_2 \wedge \cdots C_k$

Let $x_1, \cdots x_l$ be the vars of φ.



x_1 diamond

Each x_i - diamond has the same format.

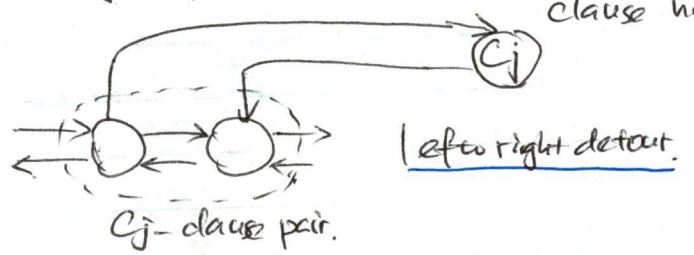
x_2 diamond.

x_l diamond.

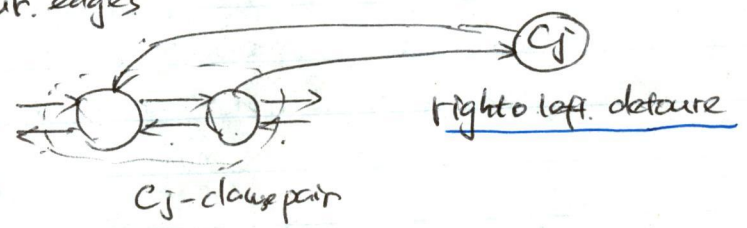
k clause nodes

The 'detours edges' shows which literal occurs in which clause.

- If x_i occurs in C_j - create the following detour edges from/to the x_i - diamond.



- If $\bar{x}_i$ occurs in C_j, create the following detour edges



11/8.

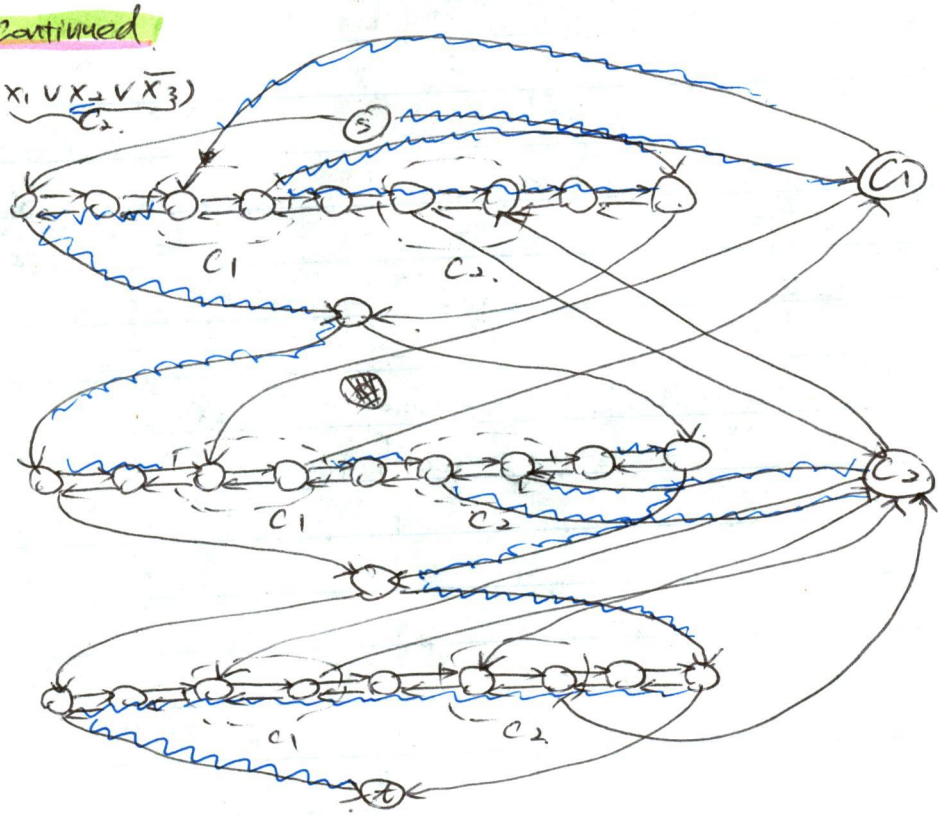
3SAT $\leq_p$ HAMPATH Continued

$\varphi = (\underbrace{\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3}_{C_1}) \wedge (\underbrace{x_1 \vee x_2 \vee \bar{x}_3}_{C_2})$

x_1 var diamond

x_2 var diamond

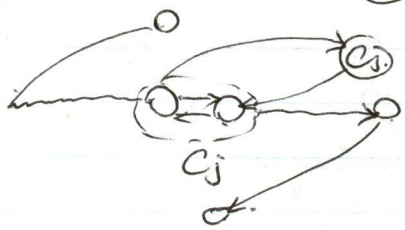
x_3 var diamond



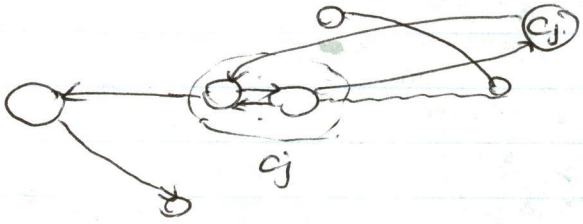
Observations and intuitions

(1) Each Hampath from s to t must traverse each x_i diamond in zig-zag way ↙↗ OR zag-zig way ↘↗

(2) If x_i appears in C_j,
then zig-zag traverse in x_i-diamond can take a left-to-right detour to the clause node (C_j)



If $\bar{x}_i$ appears in C_j
then zag-zig traverse in x_i-diamond can take a right to left detour to the clause node (C_j)



(3) Assignment: $x_i = 1$ will correspond to a zig-zag traverse in the x_i-diamond
$x_i = 0$ zag-zig

(4) A HAM must have exactly one detour path to/from each clause node.

# of nodes in the produced graph =

$(3k+3)\ell + (\ell+1) + k$ ← # clause nodes

- k is # of clauses
- ℓ is # of VARs
- $(3k+3)$: # of nodes in each 2-way horizontal row
- ℓ: # of diamond
- $(\ell+1)$: # of "apex" nodes including s, t.

Input size: $n = 3k$, $k = \frac{n}{3}$

$\ell \le 3k$.

$\le (3k+3)(3k) + (3k+1) + k$

$9k^2 + 9k + 3k + 1 + k$

$9k^2 + 13k + 1$

$9\left(\frac{n}{3}\right)^2 + 13\left(\frac{n}{3}\right) + 1$

$n^2 + \frac{13}{3}n + 1 = O(n^2)$

# of edges $\le m(m-1)$, m = # of nodes

Hence, the graph can be generated in poly time.

Proof:

φ is satisfiable $\Leftrightarrow$ G has a Ham-path from s to t.

($\Rightarrow$) Suppose φ is satisfied by A. Define the following path:

If $x_i = 1$, the x_i-diamond is traversed zig-zag

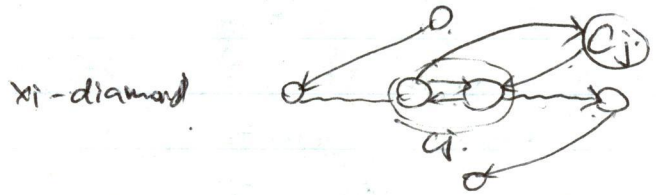
If $\bar{x}_i = 1$ zag-zig.

This path traverses all nodes from s to t exactly once except the clause nodes.

We add the following detour paths to/from each clause nodes $c_1, \ldots, c_k$

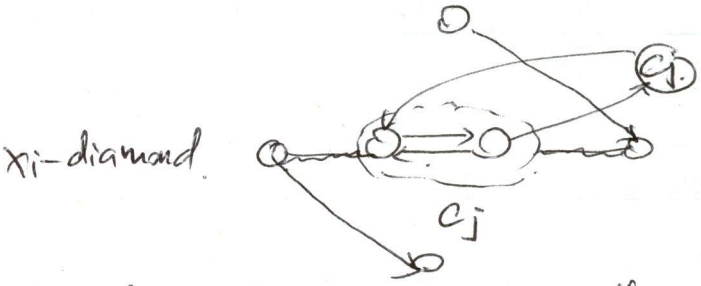
In each c_j, $1 \le j \le k$, select one true literal under A.

If it is $x_i = 1$, the detour is from the clause pair for c_j in the x_i-diamond to the clause node c_j



this detour is consistant with the zig-zag traversal assigned to the x_i-diamond

If it is $\bar{x}_i = 1$, the detour is from the clause pair for c_j in the x_i-diamond to the clause node c_j using right-to-left



This detour is consistant with the zag-zig traversal assigned to the x_i-diamond.

(If neither x_i nor $\bar{x}_i$ is selected for c_j, the x_i-diamond has no detour in this path)

The resulting path is a HAM-PATH from s to t.

(舉例) ✓ $\bar{x}_1 = 1$, $x_2 = 1$, $x_3 = 0$, $\bar{x}_3 = 1$

Blue line (前圖)

先判定 x_i diamond

再看 C_j 裡面 detour

Before proving ($\Leftarrow$), we show any HAM-path from s to t in G must traverse the diamonds in order of $x_1, \ldots, x_k$ and visit to each clause node c_j must come back to the same clause pair for c_j from which it started.