NP-Complete Problems

To show that L is NP-Complete

- show that every $A \in NP$ reduces to L & $L \in NP$

OR

(*) - show that $L' \leq_p L$ & $L \in NP$ where L' is a language known to be NP-Complete.

Proof

SAT is NP-Complete [Cook-Levin Theorem]

Once we know one problem to be NP-Complete we can show other problems to be NP-Complete using (*)

Defn: A Boolean formula over variables $u_1, u_2, \ldots, u_n$ consists of variables & logical operators $\wedge, \vee, \neg$

e.g. $(u_1 \vee \neg u_2) \wedge (u_2 \vee u_3 \vee \neg u_4) \wedge (\neg u_2 \vee \neg u_3)$ (also in 3-CNF)
satisfiable (satisfying assignment: $u_1 = 0, u_2 = 0, u_3 = 1, u_4 = 0/1$)

Defn: If ϕ is a Boolean formula over $u_1, u_2, \ldots, u_n$ & $z \in \{0,1\}^n$, then $\phi(z)$ denotes the value of ϕ when $u_1, u_2, \ldots, u_n$ are assigned values given by z.

Defn: A formula ϕ over n variables **satisfiable** if $\exists\, z \in \{0,1\}^n$ such that $\phi(z) = 1$. (z is called a satisfying assignment)
Otherwise ϕ is **unsatisfiable**.

Defn: A Boolean formula ϕ over $u_1, u_2, \ldots, u_n$ is in Conjunctive Normal Form (CNF) if it is an AND of ORs of variables or their negations.

literals - variables / their negations
clause - OR of literals

$$SAT = \{ \phi \mid \phi \text{ is a satisfiable CNF formula} \}$$

Defn: A Boolean formula ϕ is in k-CNF if ϕ is in CNF & every clause consists of at most k literals.

$$3SAT = \{ \phi \mid \phi \text{ is a satisfiable 3-CNF formula} \}$$

Cook-Levin Thm: SAT is NP-Complete

Every $L \in NP$ reduces to SAT & $SAT \in NP$

L is accepted by some poly-time NDTM M.
Encode the description of M, x into a Boolean formula ϕ s.t. M accepts $x \iff \phi$ is satisfiable.

certificate for a "yes" instance is a satisfying assignment

Thm: 3SAT is NP-Complete

Proof: 3SAT $\in$ NP : given an assignment, we can check in deterministic polynomial time whether the given formula is satisfied by the assignment.

3SAT is NP-Hard:

$SAT \leq_p 3SAT$

ϕ : instance of SAT

We map ϕ to ψ, an instance of 3SAT, as follows.

- Clauses in ϕ of size ≤ 3, remain as they are in ψ.
- Suppose C is a clause in ϕ of size $k > 3$.

[let $u_1, u_2, \ldots, u_n$ be ϕ's variables]

$C = x_1 \vee x_2 \vee \dots \vee x_{k-2} \vee x_{k-1} \vee x_k$ x_i's: literals.

introduce a new variable z.

$C_1 = x_1 \vee x_2 \vee \dots \vee x_{k-2} \vee z$

$C_2 = x_{k-1} \vee x_k \vee \neg z$

include C_1, C_2 in Ψ

Size of C_1 : $k-1$

$C \in SAT$	1	0		0/1	0
	0	1		1	1
$C \notin SAT$	0	0		0	0

C_2 is not satisfiable

C_1 is not satisfiable.

Claim: C is satisfiable iff $C_1 \wedge C_2$ is satisfiable.

If C is false, then no matter what we assign z, one of C_1, C_2 & hence $C_1 \wedge C_2$ will be false.

If C is satisfiable, the so is $C_1 \wedge C_2$.

Case 1 : $x_1 \vee x_2 \vee \dots \vee x_{k-2} = 0$. Then $x_{k-1} \vee x_k$ must be 1 & so $z = 1$ satisfies $C_1 \wedge C_2$

Case 2 : $x_1 \vee x_2 \vee \dots \vee x_{k-1} = 1$, then $z = 0$ satisfies $C_1 \wedge C_2$.

Repeatedly apply the procedure just outlined to reduce the sizes of all clauses to 3.

Thus $SAT \leq_p 3SAT$.

Integer Programming (IP)

A set of linear inequalities with rational coefficients over variables $u_1, u_2, \ldots, u_n$ is in IP if there is an assignment of integers to $u_1, u_2, \ldots, u_n$ that satisfies it.

Thm: IP is NP-Complete

Proof: IP $\in$ NP (show this!)

$x_1, x_2, x_3, \ldots, x_n$

$x_i \in \mathbb{Z}$

$|x_i|$ should be poly-bounded.

IP $\in$ NP-Hard

$SAT \leq_p IP$

$\phi \longmapsto$ instance of IP.

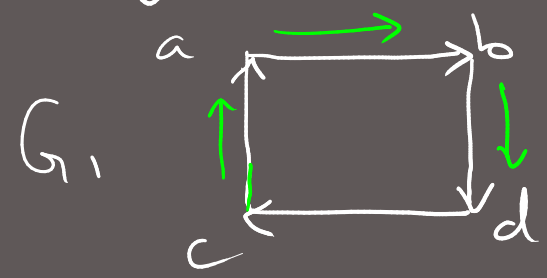
each clause in $\phi \longmapsto$ a linear inequality

→ for every variable u_i of ϕ, add constraints $0 \leq u_i \leq 1$

→ clause $u_1 \vee u_2 \vee \neg u_3 \vee \neg u_4$ would map to $u_1 + u_2 + (1-u_3) + (1-u_4) \geq 1$

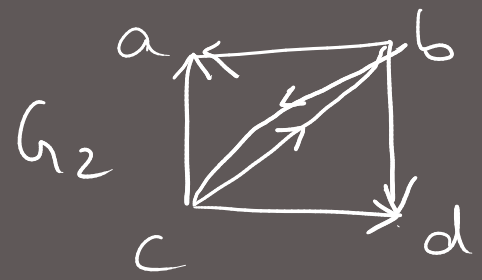
Directed Hamiltonian Path (dHAMPATH)

A directed graph $G = (V, E)$ is in dHAMPATH if $\exists$ a directed path in G visiting every vertex in V exactly once.



(d, c, a, b), (c, a, b, d) Hamiltonian paths

$G_1 \in$ dHAMPATH



No Hamiltonian path exists.

$G_2 \notin$ dHAMPATH

dHAMPATH $\in$ NP

Certificate : seq. of vertices in a Hamiltonian path.

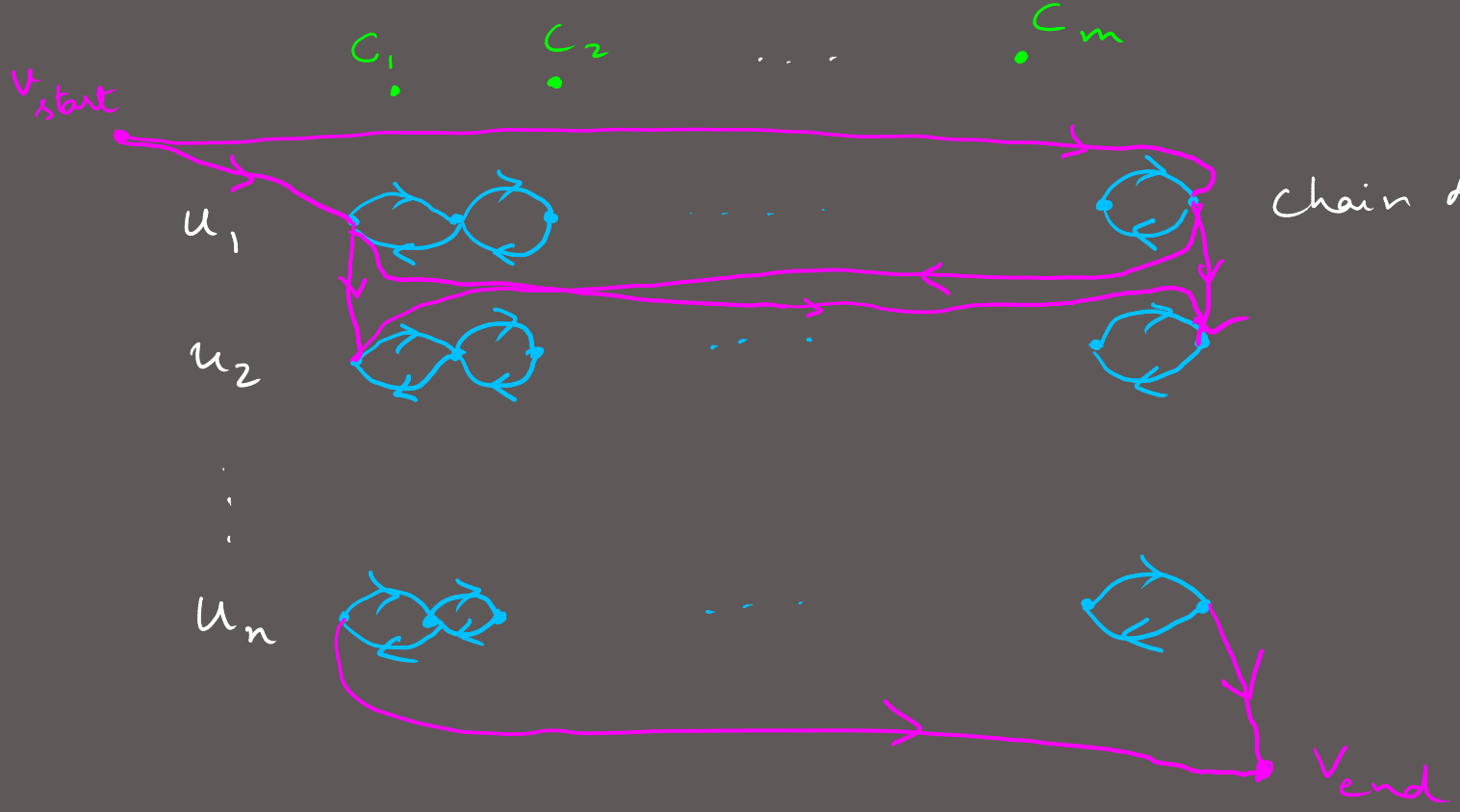
dHAMPATH $\in$ NP-Hard

$SAT \leq_p$ dHAMPATH

$\phi \longmapsto G = (V, E)$ s.t. $\phi \in SAT$ iff $G \in$ dHAMPATH

Construction of G

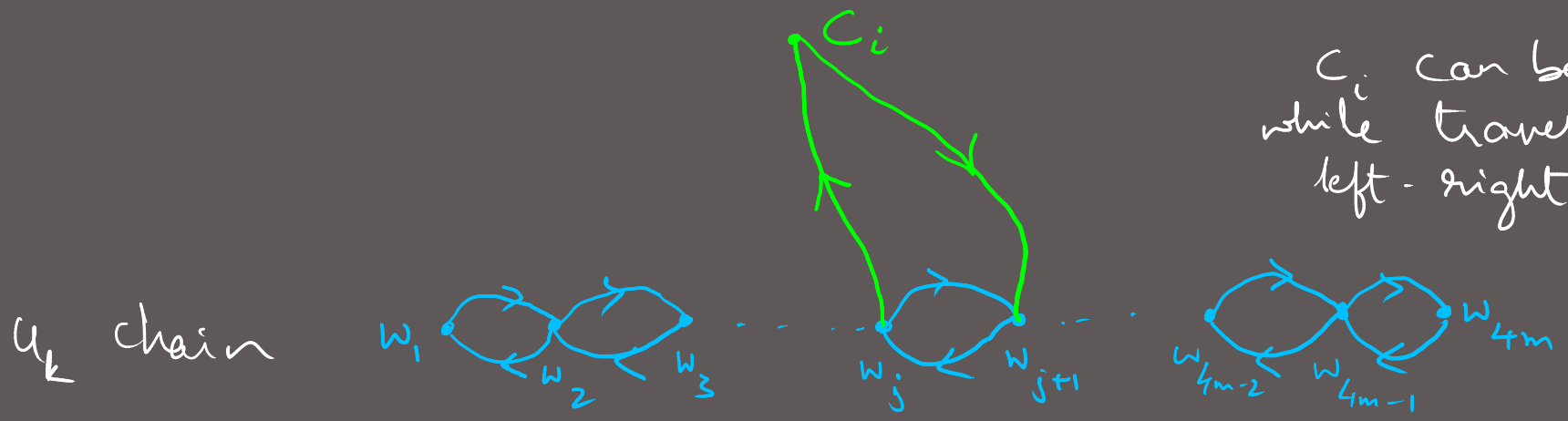
$\phi = C_1 \wedge C_2 \wedge \dots \wedge C_m$ over variables $u_1, u_2, \dots, u_n$.



chain of $4m$ vertices

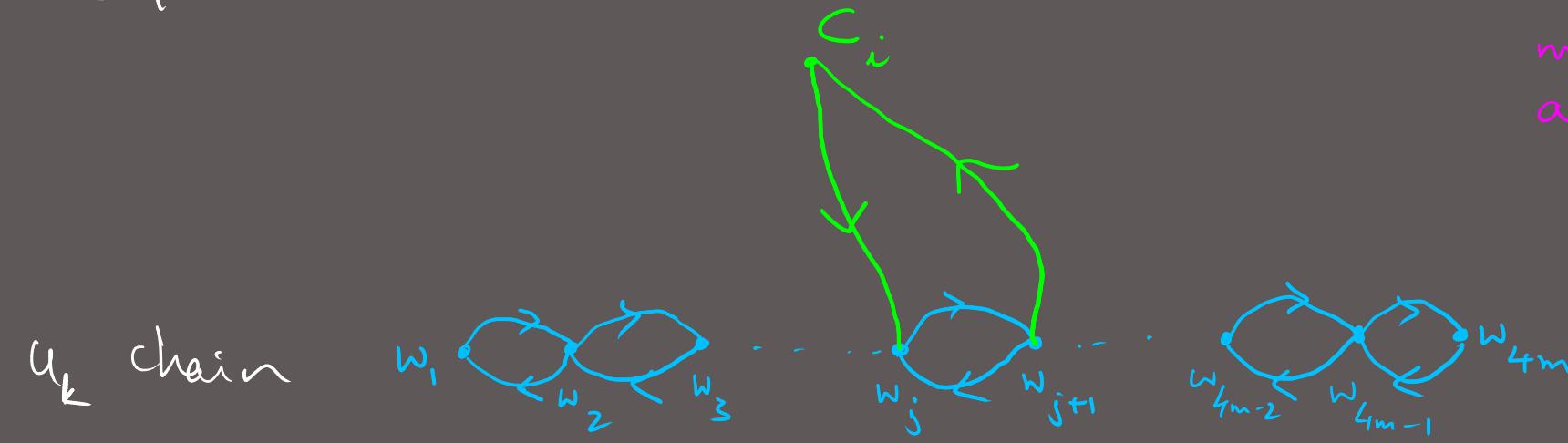
If we ignore vertices $c_1, c_2, \dots, c_m$, then $\exists$ a Hamiltonian path starting at v_{start} & ending at v_{end}.

C_i contains literal u_k



C_i can be visited while traversing the chain left - right

C_i contains literal $\neg u_k$



Choice of the link (w_j, w_{j+1}) made s.t. the adjacent links are unused.

C_i can be visited while traversing the chain right — left.

$\phi \in SAT \Rightarrow G_\phi \in dHAMPATH$

Path will start at v_{start}, traverse all the chains corresponding to $u_1, u_2, \ldots, u_n$ & end in v_{end}.

Consider a satisfying assignment for ϕ.

Chain u_k is traversed in left to right order if $u_k = 1$
in right to left order if $u_k = 0$.

Each clause has at least one literal $= 1$, each vertex c_i can be visited exactly once.

$G \in$ dHAMPATH $\Rightarrow \phi \in$ SAT

Any Hamiltonian path in G must start & end in v_{start} & v_{end} respectively.

If the chain corresponding to u_k is traversed left-right, then set $u_k = 1$.
Otherwise set $u_k = 0$.

Clauses visited while traversing u_k-chain will be satisfied.

Every c_i is visited exactly once & hence every clause is satisfied i.e., ϕ is satisfiable.