More NPC Reduction

Frank

March 24, 2022

1 Hamiltonian Cycle (Directed/Undirected)

Hamiltonian Cycle: A cycle that goes through each node exactly once

1.1 DHC

Instance: $\langle G \rangle$, G is a digraph

Q: Does G have a HC?

1.2 NP Completeness of Hamiltonian Cycle

Thm 10.1: DHC $\in$ NPC

Thm 10.2: UHC $\in$ NPC

Proof of 10.1:

Need

1. DHC $\in$ NP (trivial)
2. VC $\leq_m^P$ DHC (will prove this)

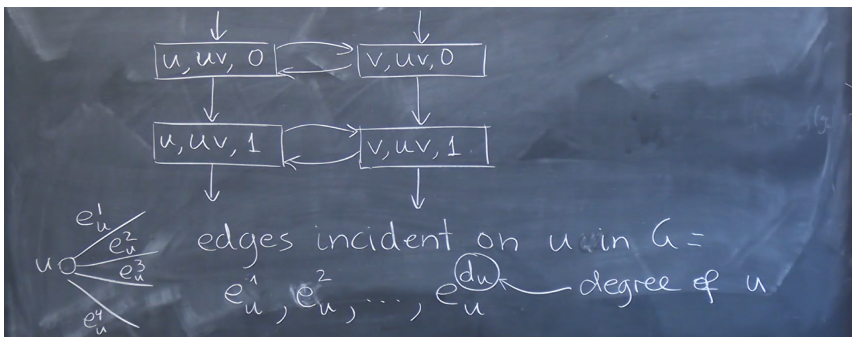
Given undirected graph $G = (V, E)$ and $b \in \mathbb{Z}^+$, construct (in polytime) digraph $G_D = (V_D, E_D)$ s.t. G has a v.c. of size $b \iff G_D$ has a HC.

$V_D = c_1, c_2, \ldots, c_b$ [will correspond to the b nodes of v.c. of G]

+ four nodes for every $\{u, v\} \in E$ [shorthand $\{u, v\}$ to uv or vu - (undirected)]

with the form $[u, uv, 0], [u, uv, 1]$ (point of view of u) , $[v, uv, 0], [v, uv, 1]$ (point of view of v)

and connect the nodes as follows:



$E_D =$

* $([u, e_u^i, 0], [u, e_u^i, 1])$, $\forall u \in V$, $\forall i \in 1..d_u$
* $([u, e_u^i, 1], [u, e_u^{i+1}, 0])$, $\forall u \in V$, $\forall i \in 1..d_u - 1$
* $([u, e_u^i, 0], [v, e_v^j, 0])$, $\forall u, v \in V$, $\forall i, j$ s.t. $e_u^i = e_v^j = uv = vu$
* $([u, e_u^i, 1], [v, e_v^j, 1])$, $\forall u, v \in V$, $\forall i, j$ s.t. $e_u^i = e_v^j = uv = vu$
* $(c_k, [u, e_u^1, 0]), ([u, e_u^{d_u}, 1], c_k)$ $\forall k = 1..b$, $\forall u \in V$

  These edges start from the c_i nodes and goes through the column of nodes and end up back at the start.

Now consider

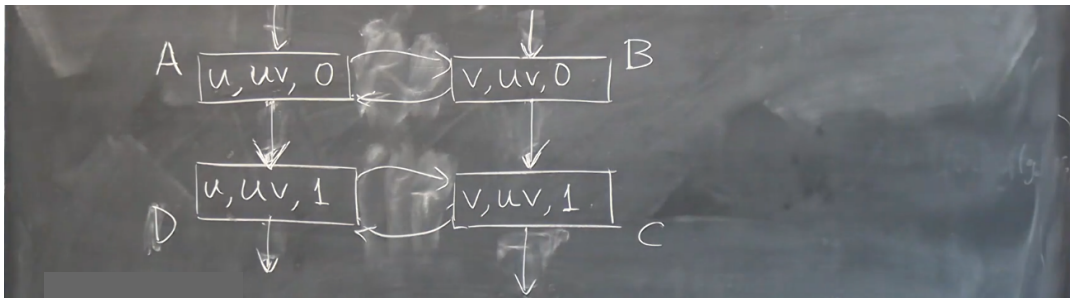


Figure 1: s102

This is a subgraph a part of a bigger graph. Observe that if we were to have a HC given that we start from A, we must exit at D, any other exits will result in missing a node.

Here is an example of the first two type of edges. Notice our VC consist of vertex 1 and 3

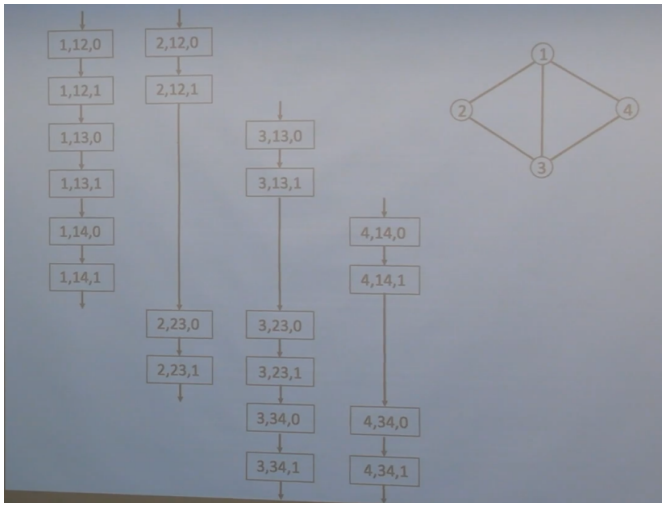


Figure 2: s103

Now the next two type of edges. Note that theres only an edge if $e_u^i = e_v^j$

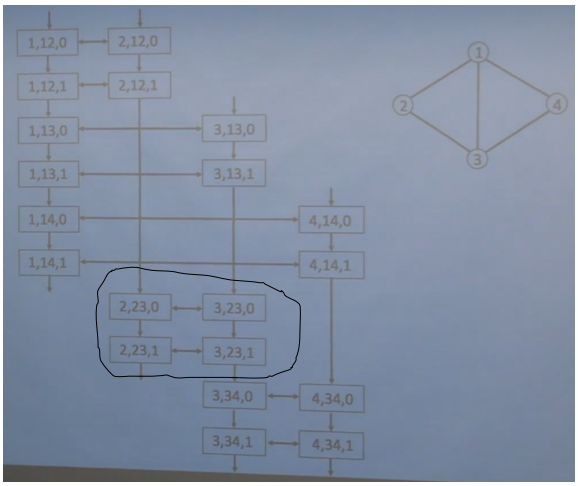


Figure 3: s104

Note that above encircled area represents a structure in figure 1.

Now, connecting using the last type of edges

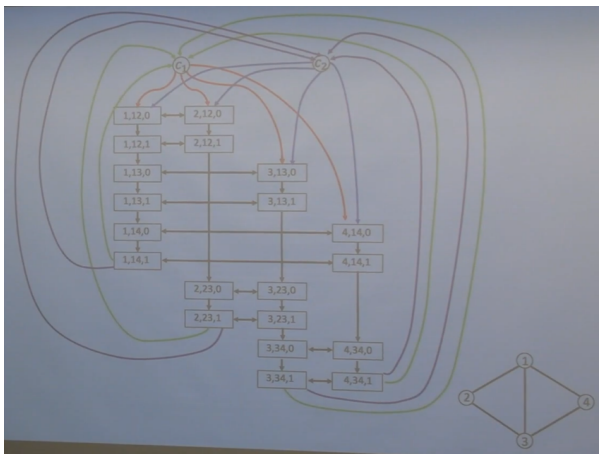


Figure 4: s105

Then starting from c_1, going down the column, and doing the $\supset$ maneuver when we encounter structures from column 2 and 4, going to c_2, going down the column, and doing the $\subset$ maneuver when we encounter structures from column 2 and 4 (not in the vc), then going back to c_1. We achieve a HC.

Note that we can percisely do such maneuver because c_1 and c_2 are vc, and thus covers every edge in the graph, this means that each column of a vertex cover can "sidetrack" and "visit" the other nodes by doing a maneuver. And cycling through each v.c. in this manner will allow us to sidetrack and visit all other nodes in the graph, and in the end going back to the starting v.c. node, thus achieving a HC.

Now, given a HC, simply reverse the process

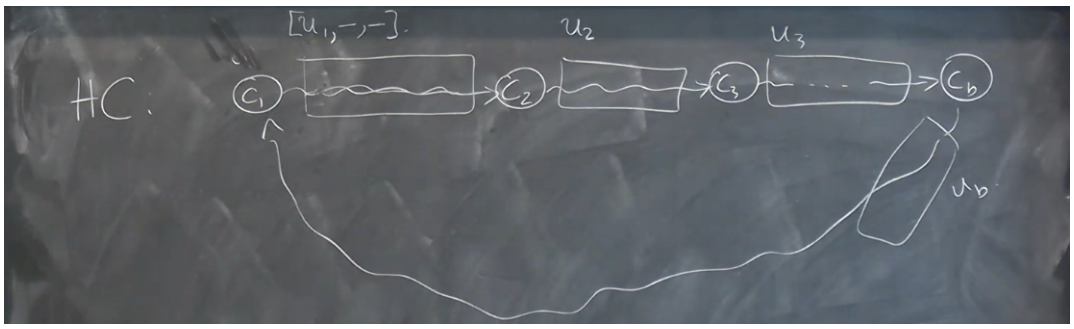


Figure 5: s106

We will choose $u_1, \ldots, u_b$ to be v.c. in G.

1.3 Undirected Variation

Proof of 10.2: DHC $\leq_m^P$ UHC

Given directed $G = (V, E)$, construct (in polytime) undirected $G' = (V', E')$ s.t. G has HC $\iff$ G' has HC

Simply split each node u into 3 nodes, one representing an "in-node", one representing a "node", and one representing an "out-node", to represent in-edges, out-edges in the undirected graph. As follows:

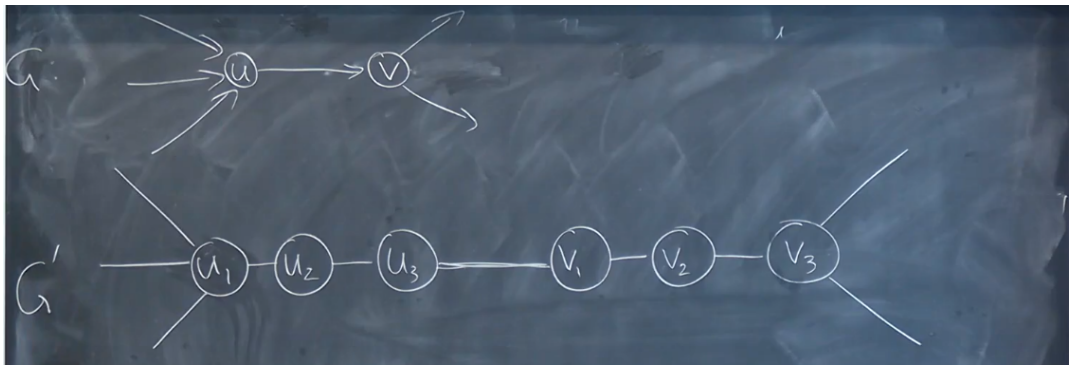


Figure 6: s107

Note that we have 3 nodes instead of 2. Why? I suppose it's to deal with the "across-edges" in the construction of the HC. We should not attach it to the in-node or out-node, but the "middle-node".

2 So far...

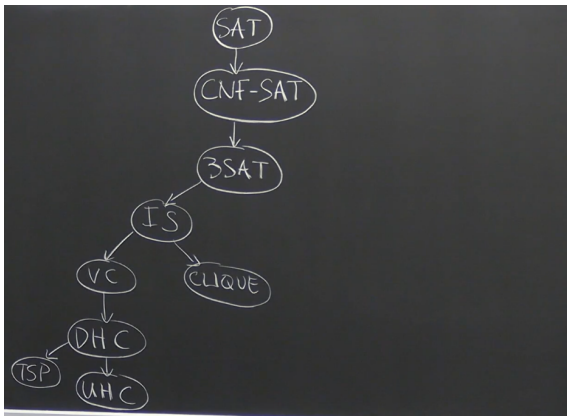


Figure 7: s108

In this lecture, we will do these

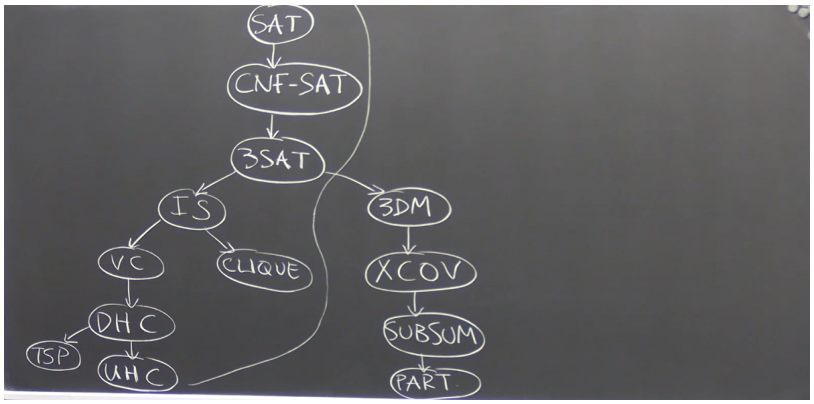


Figure 8: s109

3 3 Dimensional Matching

Instance: $\langle A, B, C, M \rangle$ where $|A| = |B| = |C| = n$ and $M \subseteq A \times B \times C$ $[(a, b, c) \in M]$

Question: Does there exist matching $M' \subseteq M$, $|M'| = n$ s.t. $\forall (a, b, c) \neq (a', b', c') \in M'$, $a \neq a'$, $b \neq b'$ and $c \neq c'$

Basically an extension of bipartite matching

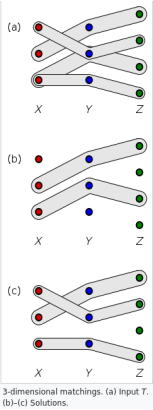


Figure 9: s110

Thm 10.3: 3DM $\in$ NPC

Proof:

1. 3DM $\in$ NP (trivial)
2. 3SAT $\leq_m^P$ 3DM (will do this)

Given 3CNF formula F, construct (in polytime) (A, B, C, M) s.t. F is satisfiable $\iff$ (A, B, C, M) has a matching

Consider $F = C_1 \wedge \ldots \wedge C_m$ and $C_j = l_j^1 \vee l_j^2 \vee l_j^3$ for vars $x_1, \ldots, x_n$

Then set C for our 3DM, $C = \{x_{ij}^0, x_{ij}^1 : 1 \leq i \leq n, 1 \leq j \leq m\}$. $|C| = 2mn$

Then for each x_{ij}^0, x_{ij}^1 have 2 triples (triangles) that share a node, as such

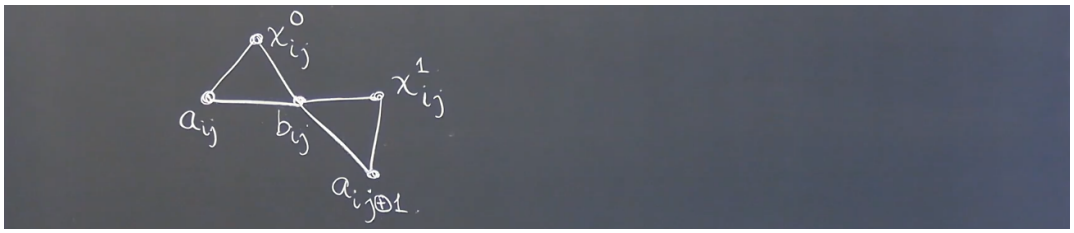


Figure 10: s111

The idea is that the triangles represent true or false of x_i, so that we can only choose one triangle. What we want is to connect the x_{ij}'s so that when one x_i is true for clause j, then it is true for all clauses. So we want to tie these triangles together.

Consider this example for x_1 in 4 clauses. Note that we can either choose all black triangles or all white triangles to satisfy both 3DM (no matching can share a node) and 3SAT (if x_{ij} is true, then x_{ij} cannot be false). Note that choosing black triangle means setting the variable to true, and choosing white triangle means setting the variable to false.

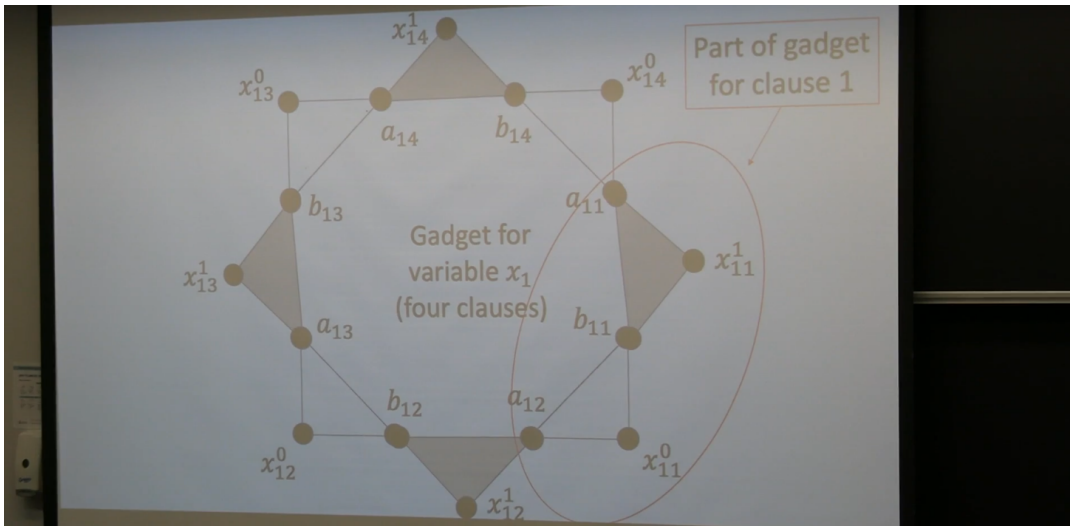


Figure 11: s112

Now we can define the first group of triples.

Group I (variables)

- $(a_{ij}, b_{ij}, x_{ij}^0), (a_{ij}, b_{ij}, x_{ij}^1)$ for $1 \leq i \leq n$ and $1 \leq j \leq m$

Group II (clauses)

connects variables with clauses

Consider clause $C_j = (l_j^1 \vee l_j^2 \vee l_j^3)$

- choose (a_j, b_j, x_{ij}^0) if $l_j^t =$ positive (makes the clause true)
- and choose (a_j, b_j, x_{ij}^1) if $l_j^t =$ negative (makes the clause false)

This is because if we choose x_{ij}^0, then we make it so that group I has to choose the black triangles (otherwise there will be overlap of nodes), and the same thing applies to choosing x_{ij}^1 (then we have to choose the white triangles)

Consider for clause 1, we draw the following triangles according to our specified triples

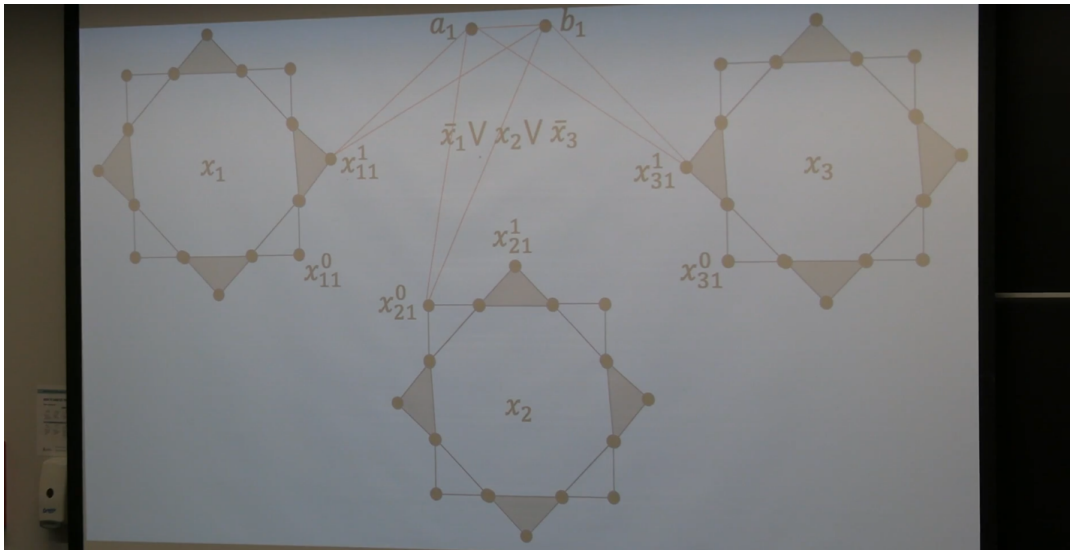


Figure 12: l113

Group III (gap fillers)

To see why we need gap fillers, note that $|C| = 2mn$, $|A|$ and $|B|$ has so far $mn+m$ (for a_{ij} and a_j). So we have $(2mn-mn-m = mn-m = m(n-1))$ more elements in C than in either A or B

- $\hat{a}_k \in A$ for $1 \leq k \leq m(n-1)$
- $\hat{b}_k \in B$ for $1 \leq k \leq m(n-1)$

New elements for A and B

To cover uncovered elements x_{ij}^b, we declare

- $(\hat{a}_k, \hat{b}_k, x_{ij}^0), (\hat{a}_k, \hat{b}_k, x_{ij}^1)$

Then our 3DM consists of triples:

- $A = \{a_{ij} : 1 \leq i \leq n, 1 \leq j \leq m\} \cup \{a_j : 1 \leq j \leq n\} \cup \{\hat{a}_k : 1 \leq k \leq m(n-1)\}$
- $B = \{b_{ij} : 1 \leq i \leq n, 1 \leq j \leq m\} \cup \{b_j : 1 \leq j \leq n\} \cup \{\hat{b}_k : 1 \leq k \leq m(n-1)\}$
- $C = \{x_{ij}^0, x_{ij}^1 : 1 \leq i \leq n, 1 \leq j \leq m\}$

Where $|A| = |B| = |C| = 2mn$

$M =$ triples in groups I, II, and III

Note that $|M| = 2mn + 3m + 2mn \cdot m(n-1) = \Theta(m^2n^2)$ which is poly in size of F

Finally, we show that F is satisfiable $\iff (A, B, C, M)$ has a matching

$(\implies)$

Let τ be a t.a. that satisfies $F \implies \tau$ satisfies each C_j, $1 \leq j \leq n$

Define $M' \subseteq M$ based on τ:

* Group I triples in M': $\forall i,\ 1 \le i \le n$,
  if $\tau(x_i) = 1 \implies$ put $(a_{ij}, b_{ij}, x_{ij}^1)$ into M' $\forall j$
  if $\tau(x_i) = 0 \implies$ put $(a_{ij}, b_{ij}, x_{ij}^0)$ into M' $\forall j$
* Group II triples in M': $\forall j,\ 1 \le j \le n$, $C_j = l_j^1 \vee l_j^2 \vee l_j^3$
  Let $t_j \in \{1, 2, 3\}$ be s.t. $\tau(l_j^{t_j}) = 1$
  if $l_j^{t_j} = x_i$ then put (a_j, b_j, x_{ij}^0) in M'
  if $l_j^{t_j} = \bar{x}_i$ then put (a_j, b_j, x_{ij}^1) in M'
* Group III triples in M': For any uncovered $x_{ij}^{0/1}$ by triples
  if Group I or II, put $(\hat{a}_k, \hat{b}_k, x_{ij}^{0/1})$ in M' $\forall k \in 1..m(n-1)$

Exercise: check that M' is a matching

$(\impliedby)$

Assume $\exists M'$ matching of (A, B, C, M)

Define τ:

$$\tau(x_i) = \begin{cases} 1 & \text{if } M' \text{ contains } (a_{ij}, b_{ij}, x_{ij}^1) \\ 0 & \text{if } M' \text{ contains } (a_{ij}, b_{ij}, x_{ij}^0) \end{cases}$$

Exercise: check that τ satisfies each $C_j \implies$ satisfies F

4 Exact Cover (XCOV)

Instance: $\langle U, C \rangle$, where U is a universe of elements of a finite size $U = \{u_1, \ldots, U_n\}$, and $C = \{A_1, \ldots, A_m\}$ set of covers s.t. $A_j \subseteq U$

Q: Does $\langle U, C \rangle$ have an exact cover: $C' \subseteq C$ s.t.

1. $\bigcup_{A \in C'} A = U$ [Cover]
2. $\forall A \neq B \in C'$, $A \cap B = \emptyset$ [Exact]

3XCOV: Special case where $|A_j| = 3$, $\forall j \in 1..m$

Thm 10.4: 3XCOV $\in$ NPC

Proof:

1. 3XCOV $\in$ NP
2. 3DM $\leq_m^P$ 3XCOV [will show]

These problems are actually quite similar, consider that instances of 3DM is a set of triples (a, b, c), while instances of 3XCOV is a set of 3 element sets $\{a, b, c\}$.

We can convert instances of 3DM to instances of 3XCOV by simply "ignoring order of triples".

Cor 10.5 XCOV $\in$ NPC (general case of 3XCOV)

5 Subset Sum (SUBSUM)

Instance: $A = a_1, a_2, \ldots, a_n \in \mathbb{Z}^+$, $k \in \mathbb{Z}^+$

Q: Is there $1 \leq i_1 < i_2 < \ldots < i_m \leq n$ s.t. $\sum_{l=1}^{m} a_{i_l} = k$ (Is there any subseq of A whose sum is equal to k)

Note that since we have a sequence of numbers, we can have repeats of numbers (as opposed to sets)

Ex. $A = 3, 2, 17, 11, 3, 9$

$k = 15 \implies$ yes

$k = 24 \implies$ no

Thm 10.6: SUBSUM $\in$ NPC

Proof:

1. SUBSUM $\in$ NP (trivial)
2. XCOV $\leq_m^P$ SUBSUM (will show)

Lets look at an example

$U = \{u_1, u_2, u_3, u_4\}$

$A_1 = \{u_1, u_2\}$, $A_2 = \{u_2, u_3, u_4\}$, $A_3 = \{u_3, u_4\}$, $A_4 = \{u_1, u_4\}$, $A_5 = \{u_2, u_4\}$

Each set has a "characteristic" encoding representing whether an element is in the set.

$A_1 = 1100$, $A_2 = 0111$, $A_3 = 0011$, $A_4 = 1001$, $A_5 = 0101$

So $U = \{u_1, \ldots, u_n\}$ and $C = \{A_1, \ldots, A_j, \ldots, A_m\}$ where we associate each A_j with a bit-string $b_j[1..n]$ where the bit at position i is 1 if $u_i \in A_j$, and 0 otherwise (encode sets as binary numbers)

Now observe that if we have an exact cover, then the bit strings of that cover sum up to $1111\ldots1$ (covers all elements). However this isn't the complete picture, we can have non-covers that add up to $1111\ldots1$ with carry-overs.

To avoid the problem of carry-overs, we must think of the numbers in base $m+1$, where $1111\ldots1 = \sum_{j=0}^{n-1}(m+1)^j$. This is to create sufficiently large gaps in between numbers so that we can never have carry-overs.

Imagine adding m sets together (the most we can add together), and all numbers in the last position is 1, then we would need the second last position to be at least $m+1$ so that we cannot carry over from the last position, similarly, the third last position is at least $(m+1)^2$, so that we cannot carry-over from the second last position.

So the numbers are still encoded in binary, but our base is changed to disallow carry-overs

Using the above transformation, we can perform the reduction

6 Partition Problem (PART)

Instance: $A = a_1, \ldots, a_n$, $a_i \in \mathbb{Z}^+$

Q: Does there exist a sub list $1 \leq i_1 < i_2 < \ldots < i_m \leq n$ s.t. $a_{i_1} + a_{i_2} + \ldots + a_{i_m} = \frac{1}{2}\sum_{i=1}^{n} a_i$ (or can we partition A in such a way that one sub list sums to the same number as the other sub list, which is equivalent to saying if you can find a sub list that sums to half of the total sum of the list, then the other sub list is also half)

Thm 10.7: PART $\in$ NPC

Proof:

1. PART $\in$ NP (trivial)
2. SUBSET $\leq_m^P$ PART (will show)

Given $A = a_1, \ldots, a_n \in \mathbb{Z}^+$, and $k \in \mathbb{Z}^+$, construct new list in (polytime) $A' = a_1, a_2, \ldots, a_n, a_{n+1}$ s.t. $\langle A, k\rangle \in$ SUBSUM $\iff \langle A'\rangle \in PART$

Note that we need a_{n+1} to be as such that when we add it to k, we can get the total $(\sum_{i=1}^{n} a_i)\ T - k$, so we can balance the two sides of the partition.

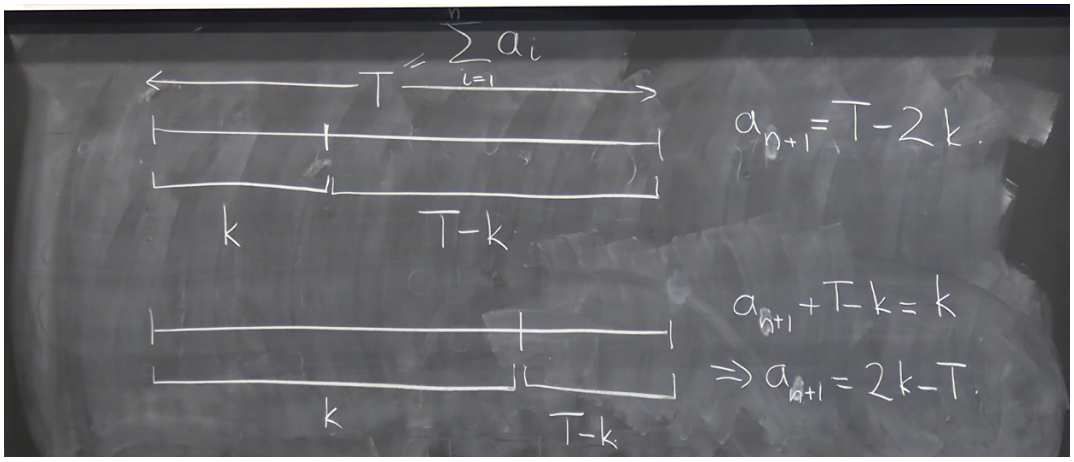


Figure 13: s118

Now, depending on the relationship between k and $T - k$, we might choose either one.

Note that if $k = T - k$, then both sides are already balanced, so we do nothing.

$$a_{n+1} = \begin{cases} T - 2k & \text{if } T > 2k \\ 2k - T & \text{if } T < 2k \\ \text{nothing} & \text{if } T = 2k \end{cases}$$

Applying the above transformation gives us the reduction.

7 Now our graph looks like

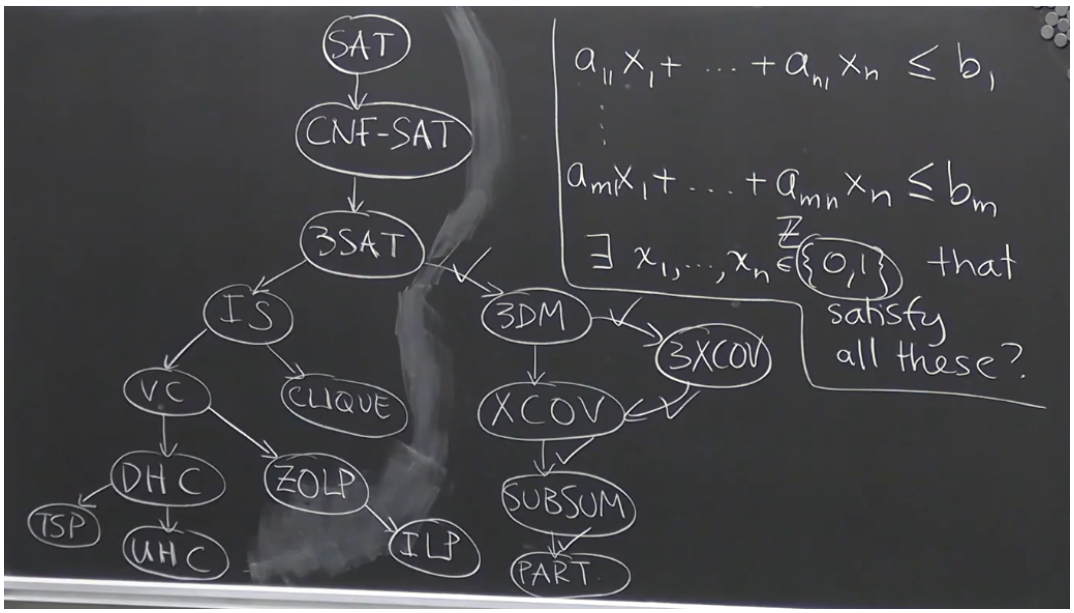


Figure 14: s119