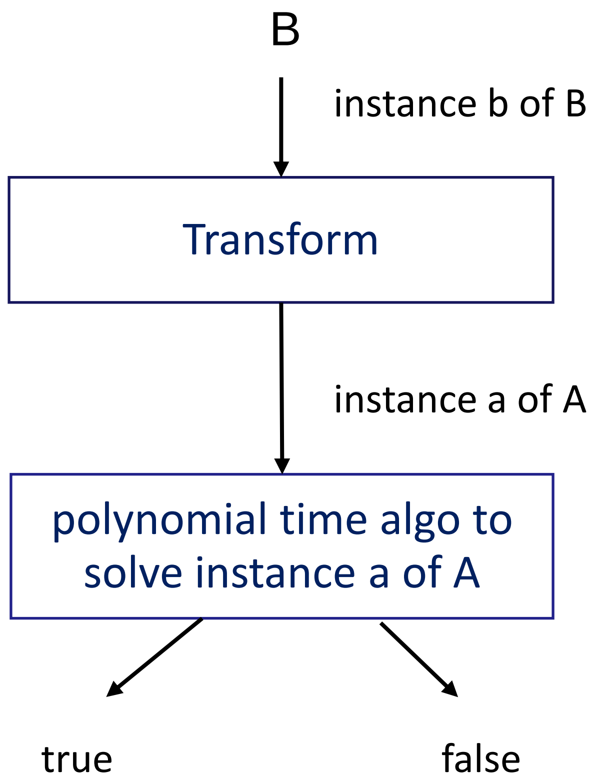
Reductions and NP-Completeness

Dianne Foreback

Goal: prove an unknown problem is in NP-Complete

- To prove an unknown (decision) problem U is in NP-Complete must:
  1. Prove U = NP
     - state U as a verification problem and provide an algorithm that verifies it in polynomial time)
  2. Prove it is in NP-Hard
     - given a problem H that is assumed to be in NP-complete, provide a reduction from H to U in polynomial time. Written $H \leq_P U$
       - ❑ the reduction is an "if and only if" (must prove in both directions)
       - ❑ A problem H is polynomial-time reducible to problem U, if any algorithm that can solve U can be used to solve H such that the increase in time is no more than polynomial

Reduction

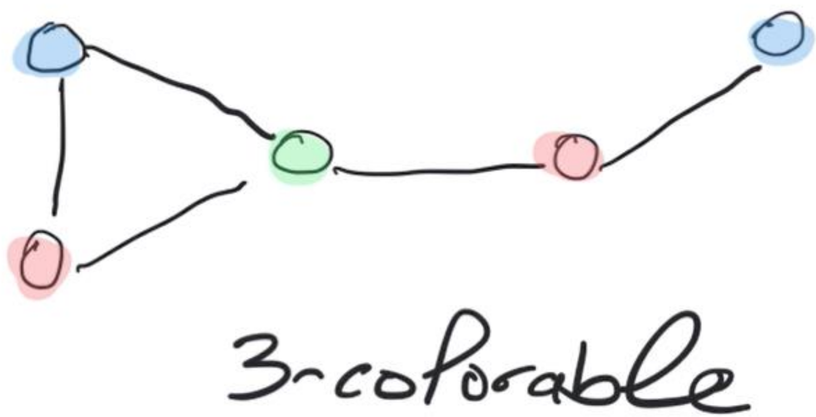

B
instance b of B
Transform
instance a of A
polynomial time algo to solve instance a of A
true
false

k-coloring of graph G

- k-coloring of a graph G =(V,E) is an assignment of a color to each $v \in V$ such that no 2 adjacent vertices share the same color.

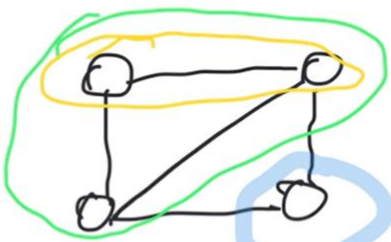
3-colorability Problem

- 3-colorability decision problem: Given a graph G, is G colorable with k= 3 colors or less?
- 3-colorability is widely believed not to be in P

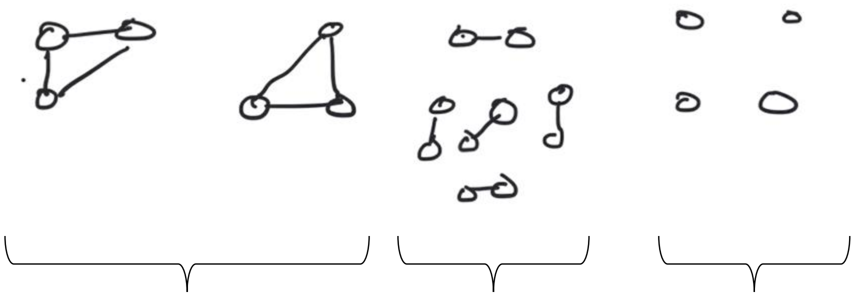


Clique & Complete Graph

- a clique of a graph G is a subgraph of G that is a complete graph
- a complete graph has an edge between every pair of vertices



G



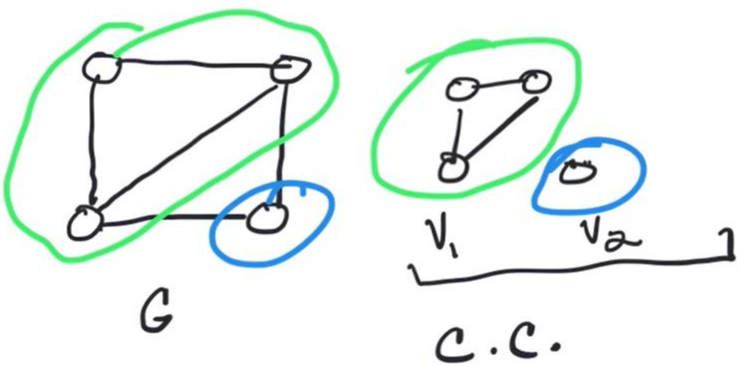
cliques of size 3

cliques of size 2

cliques of size 1

Clique Cover

- a clique of a graph G is a subgraph of G that is a complete graph
- a complete graph has an edge between every pair of vertices
- a clique cover of a graph G = (V, E) is a partition of vertices V of G into $V_1, V_2, \ldots V_k$ such that V_j is a clique



An example of a clique cover of graph G. More clique covers exist for this graph as well.

Clique Cover Problem (CC)

- Clique Cover (decision) Problem: Given a graph G, is there a clique cover of G with k cliques or less?

- We want to show this an NP-Complete Problem (our UNKNOWN problem U)
  1. Prove U = NP
     - state U as a verification problem and provide an algorithm that verifies it in polynomial time
  2. Prove it is in NP-Hard
     - given a problem H that is assumed to be in NP-complete, provide a reduction from H to U in polynomial time. Written $H \leq_P U$
     - 3-colorability $\leq_P$ Clique Cover Problem

Clique Cover Verification

Clique Cover Verification Problem: Given a graph G, a set of Cliques C, and an integer k, verify that C is a clique cover of G with no more than k cliques.

CC Verification Algorithm: CCVerify(G, C, k)

Input: A graph G = (V, E) and a certificate as a set of cliques C with each clique containing vertices from graph G, and an integer k

Output: True if the certificate has k or fewer cliques and it is a CC, false otherwise

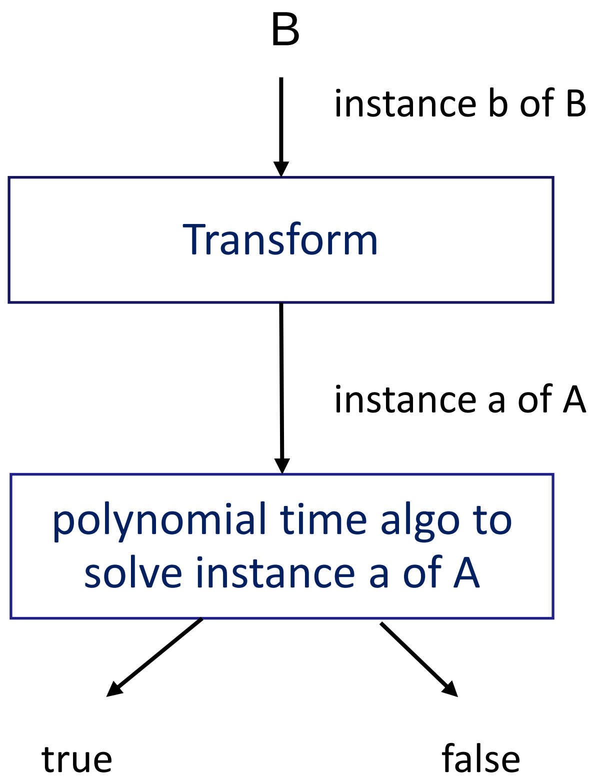
- Count the number of cliques in C, if it is greater than k, returning false if the count increases past k. This takes time k and will be less then or equal to |V|.
- Keep track of the number of times a vertex is in a clique. (The vertex should be in one and only one clique c.) If the vertex is in more than one clique return false. This is can be done in linear time iterating through vertices in each clique c in C.
- For each clique c in C, and for each vertex v in c verify that v is adjacent to all other vertices u in c. (Doing this for all vertices in all cliques can be done by examining the vertices adjacency list once, thus, a linear runtime with respect to our input)

Theorem 3-colorability $\leq_P$ CC

Theorem: If 3-colorability cannot be decided in polynomial time, neither can clique cover (CC)

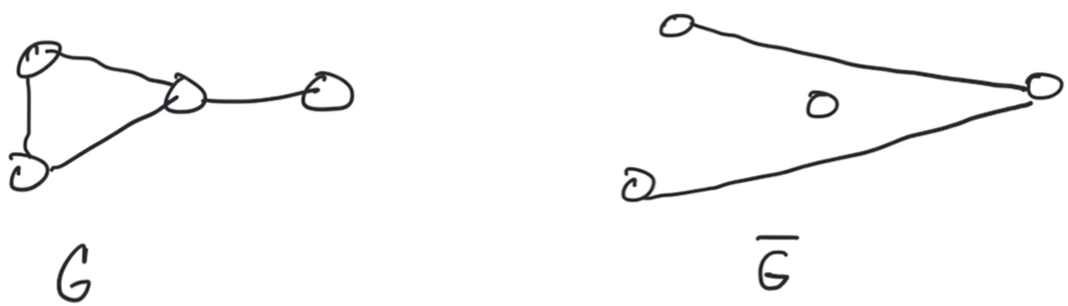
Proof by contradiction: Assume clique covering can be solved in polynomial time. This means there is a polynomial time algorithm A that given a graph G and integer k, can decide if G has a clique cover with at most k cliques.

Reduction


B
instance b of B
Transform
instance a of A
polynomial time algo to solve instance a of A
true
false

Compliment of a Graph G

- Given a graph G = (V,E), the complement of G is a graph $\overline{G}$ that has the same set of vertices as G. However, edges between vertices u and v exist in $\overline{G}$ only if the edge does not exist in G.



Algorithm: 3-colorability $\leq_P$CC

Algo: 3ColorToCC(G)

Input: Graph G

Output: a clique covering of graph G

1. Construct from G its compliment graph $\overline{G}$
2. Invoke algorithm A to find the clique cover of the compliment of graph $\overline{G}$ (Recall: in our theorem 3-colorabilty we assumed this algorithm A existed)
3. If the clique cover has 3 sets or less, make all vertices in each set have the same color (this color should be unique across the sets of clique covers)
4. return the clique cover

Theorem 3-colorability $\leq_P$ CC

Theorem: If 3-colorability cannot be decided in polynomial time, neither can clique cover (CC)

Proof by contradiction: Assume clique covering can be solved in polynomial time. This means there is a polynomial time algorithm A that given a graph G and integer k, can decide if G has a clique cover with at most k cliques.

Algorithm: 3-colorability $\leq_P$CC

Algo: 3ColorToCC(G)

Input: Graph G

Output: a clique covering of graph G

1. Construct from G its compliment graph $\overline{G}$
2. Invoke algorithm A to find the clique cover of the compliment of graph $\overline{G}$ (Recall: in our theorem 3-colorabilty we assumed this algorithm A existed)
3. If the clique cover has 3 sets or less, make all vertices in each set have the same color (this color should be unique across the sets of clique covers)
4. return the clique cover

Algo Proof of Correctness: 3-colorability $\leq_P$ CC (part 1 →)

Prove correctness of reduction in 2 parts (2 directions)

(→) Part 1: If G is 3-colorable, then $\overline{G}$ has a clique cover with 3 sets or less. This means, if G is 3-colorable there does not exist an edge between 2 vertices u and v that have the same color. Thus, in $\overline{G}$, there will be an edge between u and v. That is, colors in G represent vertices that are in the same clique in $\overline{G}$. There are 3 colors for a graph that is 3-colorable and thus 3 cliques in $\overline{G}$.

Algo Proof of Correctness: 3-colorability $\leq_P$CC (part 2 ←)

Prove correctness of reduction in 2 parts (2 directions)

(←) Part 2: If $\overline{G}$ has a clique covering with 3 sets or less, then G is 3 colorable. If V_1, V_2, V_3 are clique covers of G, then there are k = 3 sets of cliques in $\overline{G}$. For any vertices u and v in the same clique in $\overline{G}$, there is an edge $(u, v) \in \overline{G}$ and edge $(u, v) \notin G$. Therefore, if u and v are assigned the same color, there is no conflict.

Algo Proof of Correctness: 3-colorability $\leq_P$CC

Prove correctness of reduction in 2 parts (2 directions)

($\rightarrow$) Part 1: If G is 3-colorable, then $\overline{G}$ has a clique cover with 3 sets or less. This means, if G is 3-colorable there does not exist an edge between 2 vertices u and v that have the same color. Thus, in $\overline{G}$, there will be an edge between u and v. That is, colors in G represent vertices that are in the same clique in $\overline{G}$. There are 3 colors for a graph that is 3-colorable and thus 3 cliques in $\overline{G}$.

($\leftarrow$) Part 2: If $\overline{G}$ has a clique covering with 3 sets or less, then G is 3 colorable. If V_1, V_2, V_3 are clique covers of $\overline{G}$, then there are k = 3 sets of cliques in $\overline{G}$. For any vertices u and v in the same clique in $\overline{G}$, there is an edge $(u, v) \in \overline{G}$ and edge $(u, v) \notin G$. Therefore, if u and v are assigned the same color, there is no conflict.

Consequently, G is 3-colorable based on the provided 3 cliques with 3 different colors in G'.

Reduction Proof of Correctness Pitfalls

1. Omitting the reverse direction ($\leftarrow$) in the proof of correctness
   - e.g., there could be a graph G such that its compliment graph $\overline{G}$ has a clique cover of size 3 or less but graph G is not 3-colorable. You must prove in both directions
2. Implicitly assuming the output is "know". You are giving an algorithm to find the output so you cannot implicitly assume you have it to use
   - e.g. Giving an algorithm for solving 3-colorablebilty. Input is graph G, ..., then your pseudocode reads "if color of vertex us is orange then,…". If color of vertex is green then…
     - You are implicityly assuming you know the color
     - usually happens if the reduction is in the wrong order. I.e., instead of 3-color $\leq_P$ CC you are doing the reduction CC $\leq_P$3-color

Theorem 3-colorability $\leq_P$ CC

Theorem: If 3-colorability cannot be decided in polynomial time, neither can clique cover (CC)

Proof by contradiction: Assume clique covering can be solved in polynomial time. This means there is a polynomial time algorithm A that given a graph G and integer k, can decide if G has a clique cover with at most k cliques.

Next: we gave a reduction of 3-color $\leq_P$ CC and proved its correctness (in both directions)

We assumed that algorithm A exists. However, if 3-colorabiity is reducible to CC (as we proved), then we could use this algorithm A to decide 3-colorability. But 3-colorability is NP-complete, therefore there is no such algorithm.

Thus, by contradiction we proved CC does not have such an algorithm and it is NP-Hard.

Earlier we gave a polynomial verification algorithm and proved its runtime meaning the CC is in NP. Thus, CC is in NP and in NP-Hard so it is in NP-Complete

CC is in NP-Complete Proven

- Clique Cover (decision) Problem: Given a graph G, is there a clique cover of G with k cliques or less?

- We want to show this an NP-Complete Problem (our UNKNOWN problem U)
  1. Prove U = NP
     - state U as a verification problem and provide an algorithm that verifies it in polynomial time)
  2. Prove it is in NP-Hard
     - given a problem H that is assumed to be in NP-complete, provide a reduction from H to U in polynomial time. Written $H \leq_P U$
     - 3-colorability $\leq_P$ Clique Cover Problem

Thank You !

Questions ?