Reduction to Hamiltonian Cycle

Jenny Chen, Sanjit Basker

HamCycle is in NP

- Certificate:
  - Ordered list of vertices
- Verifier:
  - First check if there are duplicated vertices
  - Then check all consecutive pairs are joined by some edges belong to the graph
  - Polytime

Side note

Common NP-complete questions in your toolbox

- Independent Set – include more is harder
- Vertex Cover/Set Cover – include less is harder
- Hamiltonian Cycle – sequencing problems
- Subset Sum – numerical problems (be careful of pseudopolynomial time reduction and backward reduction)
- SAT/3SAT – if nothing above works

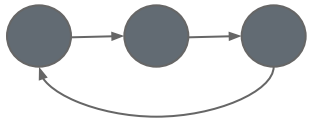
Reduce 3SAT to Hamiltonian Cycle

3SAT:

- Input variables: $x_1, x_2, \ldots, x_n$
- Clauses: $C_1, C_2, \ldots, C_m$
- Example: $(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee \overline{x_3})$
- Yes instance if some assignment of variables can satisfy the formula

HamCycle

- Input: directed graph G = (V, E)
- Yes if contains cycle such that visits each vertex exactly once



Reduction Part I

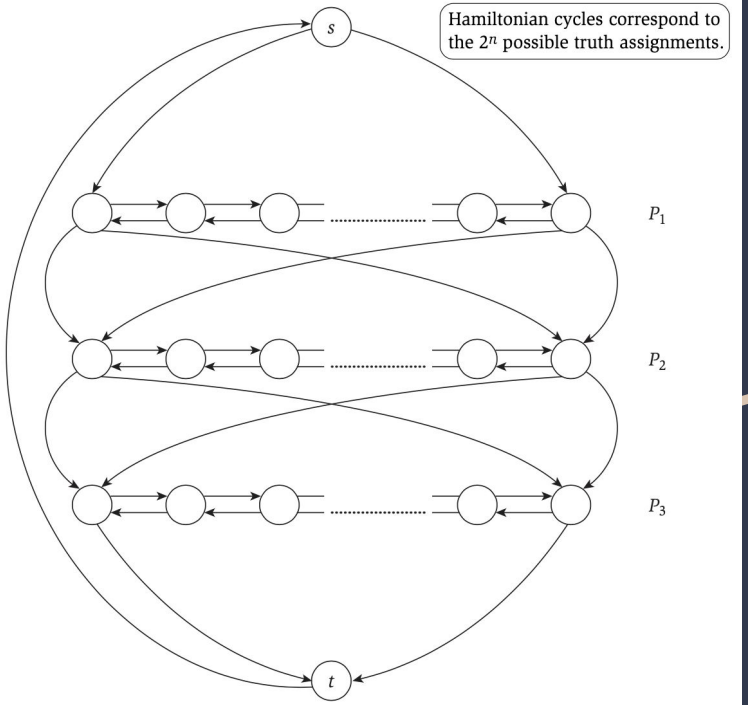


Figure 8.7 The reduction from 3-SAT to Hamiltonian Cycle: part 1.

- Each *P* corresponds to a variable
- Each time we can go left or right
  - Left to right means x_i is true
  - Right to left means x_i is false
- 2^n different possible truth assignments correspond to 2^n distinct possible cycles of the form
  $s \rightarrow$ down left/right $\rightarrow$ right/left $\rightarrow \ldots \rightarrow t \rightarrow$ s
- Not enough, why?
  - The No instance of 3SAT maps to Yes instance of HamCycle

Accounting for Clauses

- By design all variable nodes can be reached no matter the assignment to the variables
- But some assignments do not satisfy the clauses, so we have to eliminate such cycles
- Idea: we'll add a node for each clause
- So we are adding *m* new nodes
  - Each node can be included in the path if its clause is made true by the assignment
  - A Ham Cycle must include all *m* nodes
  - So a Ham Cycle will correspond to an assignment that satisfies c_1, …, c_m
  - Because we are adding new edges, we have to be careful to maintain that all Ham Cycles are of the form
    $s \rightarrow$ down left/right $\rightarrow$ right/left $\rightarrow \ldots \rightarrow t \rightarrow s$
    (no "cheating" Ham Cycles")

How to enforce a single clause?

- Suppose the clause is $c_i = x_1 \vee x_2 \vee x_3$
- The Ham Cycle should be allowed to traverse in any of the following:
  - $P_1 \rightarrow, P_2 \rightarrow, P_3 \rightarrow$
  - $P_1 \rightarrow, P_2 \leftarrow, P_3 \rightarrow$
  - …
- Total 8 ways to assign, and only one fails ($P_1 \leftarrow, P_2 \leftarrow, P_3 \leftarrow$)
- We'll add edges oriented so that traversing everything right-to-left doesn't allow you to include the node for c_i

Reduction Part II

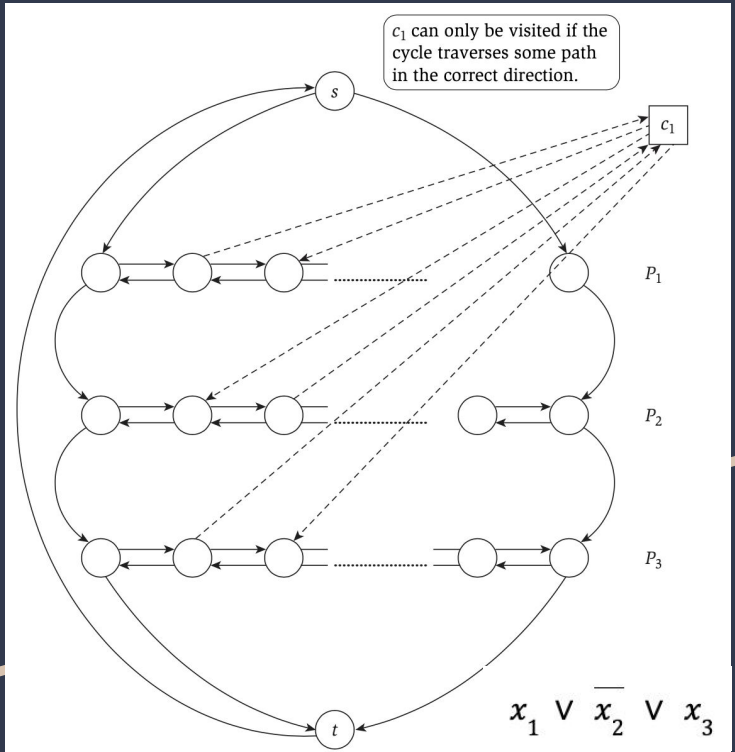


Figure 8.8 The reduction from 3-SAT to Hamiltonian Cycle: part 2.

Main idea: encourage traversing in the way that will satisfy the clause

Two cases:

- If in the form of x
  - C_i is true if x is true -> traverse from left to right
- If in the form of $\bar{x}$
  - C_i is true if x is false -> traverse from right to left

To enforce the condition:

- If traverse from left to right, left node connect to clause
- If traverse from right to left, right node connect to clause
- Total number of nodes need for one path: $3m + 3$, where m is the number of clauses

Reduction Part III

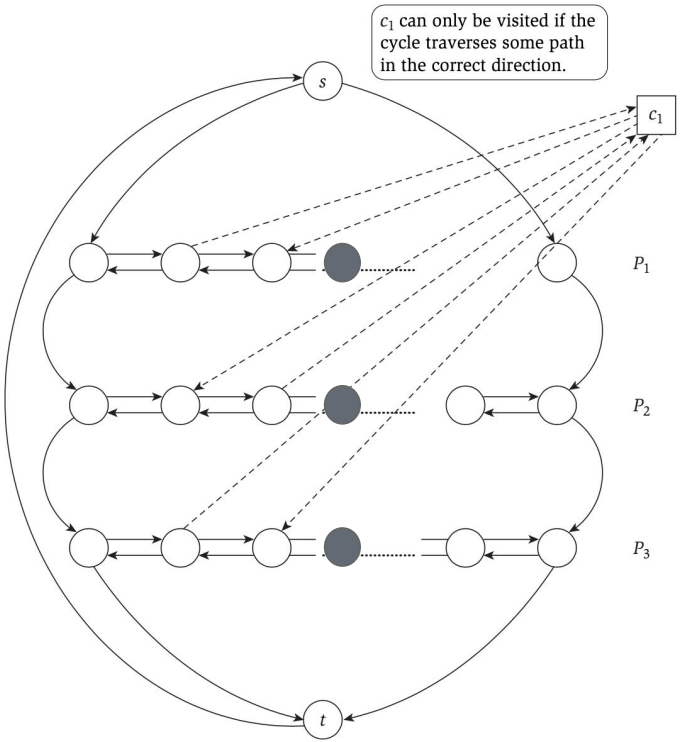


Figure 8.8 The reduction from 3-SAT to Hamiltonian Cycle: part 2.

Why 3 nodes for each clause?

- Avoid the possibility of jumping around different variables and make the proof easier :)

Since this extra node is “uniquely tagged” by this particular clause and variable, if we ever get out of the variables paths to reach the clause node, we are forced to come back. Otherwise, this node will never be reached. Hence we don’t need to argue the case where we jump around using different clauses.

Proof, First Direction

Want to show that suppose there exists some satisfying assignment to the 3SAT instance, then there exists some Ham Cycle in the reduction graph

We claim that the following is a valid cycle:

- If x_i is true, traverse P_i from left to right. Reverse for false
- When traversing, if C_j is already reached then continue, otherwise go touch C_j

Why:

- All nodes along P_i's will be traversed
- By construction, C_j is reachable if at least one of its literals is evaluated to true -> C_j is satisfied, and we know all clauses are satisfied by assumption
- Node *s* and *t* are always reachable

Proof, Second Direction

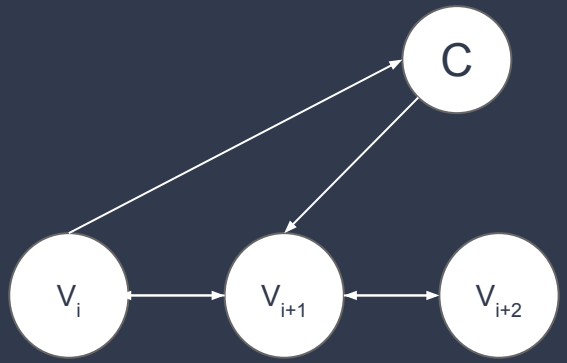
It's easier to prove that if the 3SAT is unsatisfiable, then there exists no Ham Cycle in the graph. To do this, we assume that any Ham Cycle is of the form
$s \rightarrow$ down left/right $\rightarrow$ right/left $\rightarrow \ldots \rightarrow t \rightarrow s$ (Lemma). Then, we can reason about truth assignments.

Since the formula is unsatisfiable, there must exists some clauses such that all literals are evaluated to false, regardless of what assignment we choose.

By construction, variables evaluated to false traverse in the direction that cannot reach the clause node -> the clause "missed" all three of its chances to be reached/satisfied

No possible Ham Cycle in the graph.

Outline of Proof of Lemma



Claim: Any Ham Cycle in the graph is of the form $s \rightarrow$ down left/right $\rightarrow$ right/left $\rightarrow \ldots \rightarrow t \rightarrow s$.

Proof Sketch:

Consider a Ham Cycle. It must start with s and then contain the first/last vertex of P_1. After that, it must traverse the path at least partially in some direction, and possibly visit a clause node. If it doesn't visit any clause nodes, it goes to the opposite end of P_1 and continues. If it does visit a clause node, it cannot transfer into some other path P_j because then it would be unable to include both $v_{1,\, i+1}$ and the unconnected dummy node $v_{1,\, i+2}$ (we've already used c, so there's a dead end and you can't make it to t anymore).

We can prove by induction that it must traverse every path P_i in this way as well.