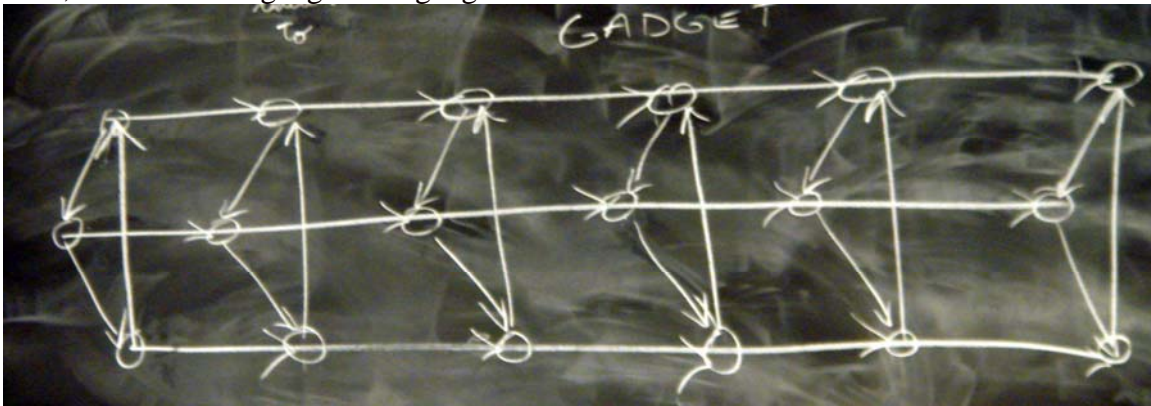


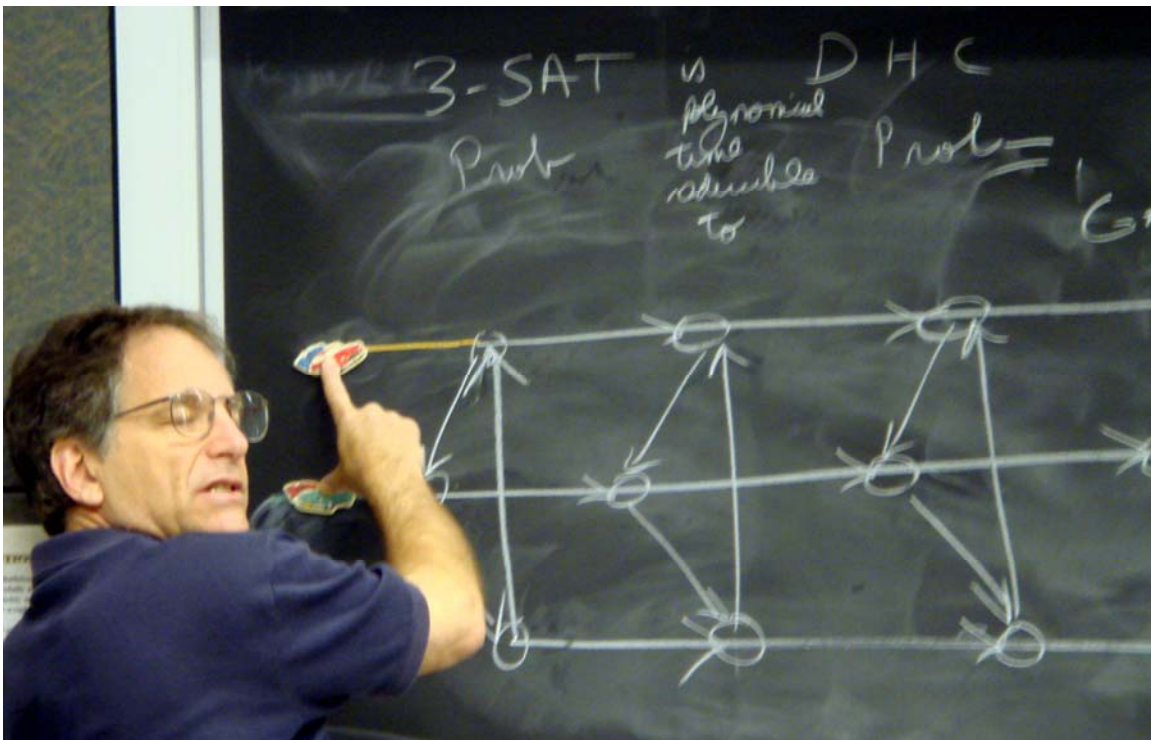
CSCI 303 Introduction to Algorithms
Spring 2007
March 7th, 2007 class notes

The 3-SAT problem is polynomial-time reducible to the Direct Hamiltonian Cycle problem.

First, we discuss a “gadget.” A gadget is an 18 node construction:



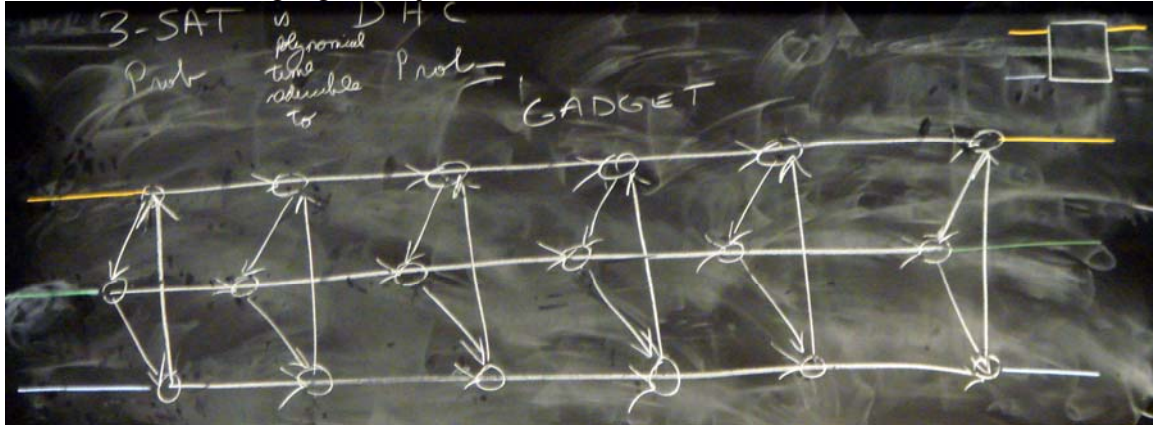
Suppose 3 cars arrive at the 3 left-most nodes: a green, a red, and a blue car. Can these cars traverse this graph using only available edges such that each node is visited by exactly 1 car and each car exits on the same row it entered?



The answer is yes. Further, each car must exit on the same row it entered and cannot exit on a different row.

The same statement is true if only 2, or even just 1 car arrive on the left side.

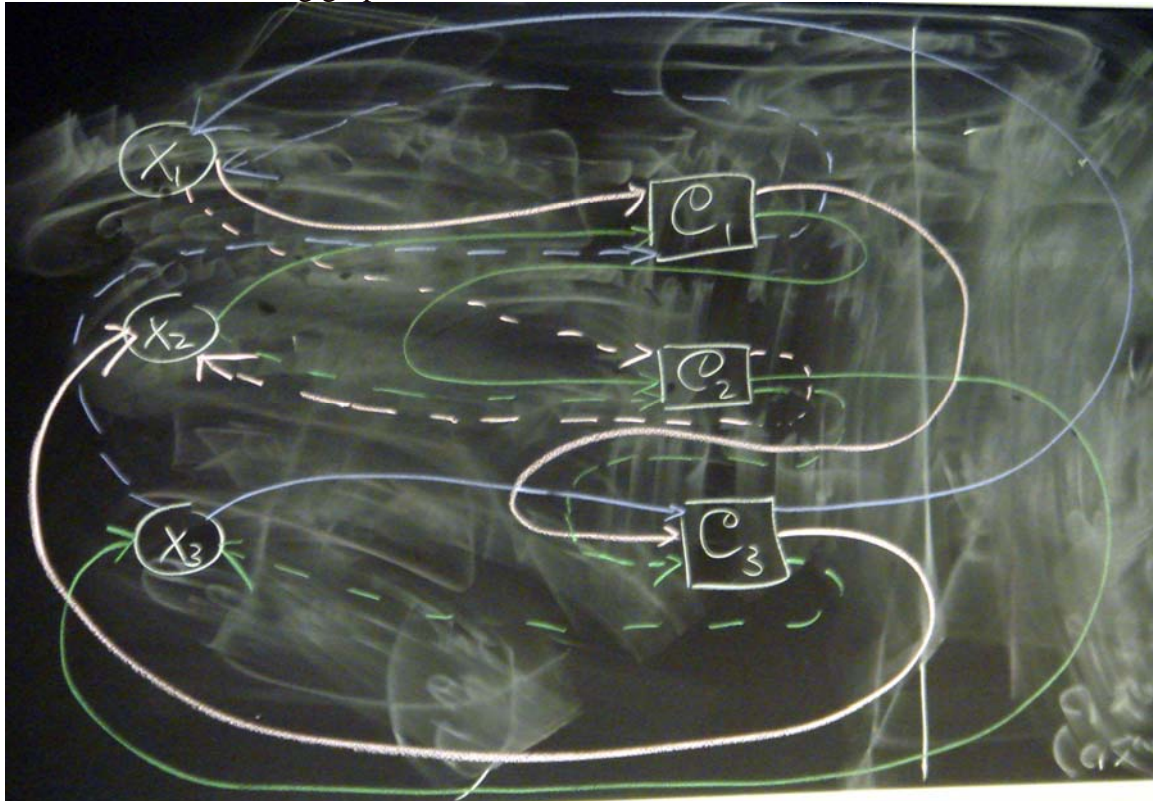
We will abstract the gadget and just draw a box with lines:



Suppose we start with our usual Boolean formulae:

$$\Phi = (x_1 \vee x_2 \vee \neg x_3) \wedge (x_2 \vee \neg x_2 \vee \neg x_1) \wedge (x_3 \vee \neg x_2 \vee x_1)$$

We create the following graph:

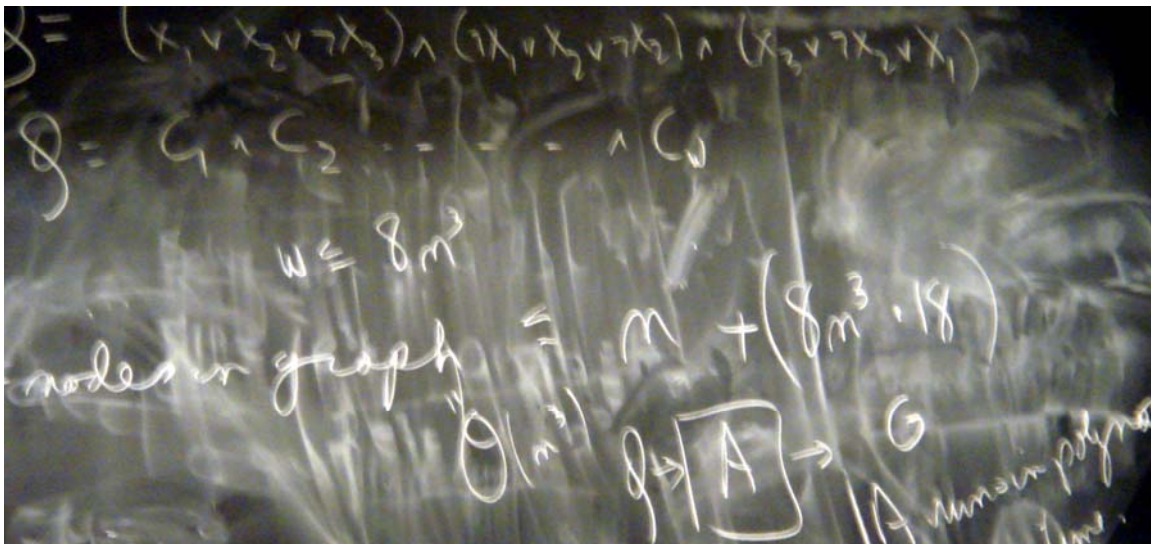


The algorithm A: on input Φ , construct a graph G as follows. For each variable create a node. For each clause create a gadget. For each literal, draw edges from that literal's

variable, to the first clause that literal appears in, to the second clause that literal appears in, etc., to the next variable (x_1 goes to x_2 , x_2 goes to x_3 , x_n goes to x_1).

The number of nodes in the graph is no greater than $18(8n^3) + n = 144n^3 + n$.

A runs in polynomial time because it has $O(n^3)$ nodes and $O(n^3)$ edges. A is also answer-preserving. Every literal selection corresponds to an attempted cycle through the nodes of the graph. Thus if $x_1 = T$, then the cycle includes the edges corresponding to the literal x_1 (if $x_1 = F$, then the cycle includes the edges corresponding to the literal $\neg x_1$). If the literal selection visits each clause once, twice, or three times, then Φ is satisfiable. Thus there exists a Hamiltonian cycle in G iff Φ is satisfiable.



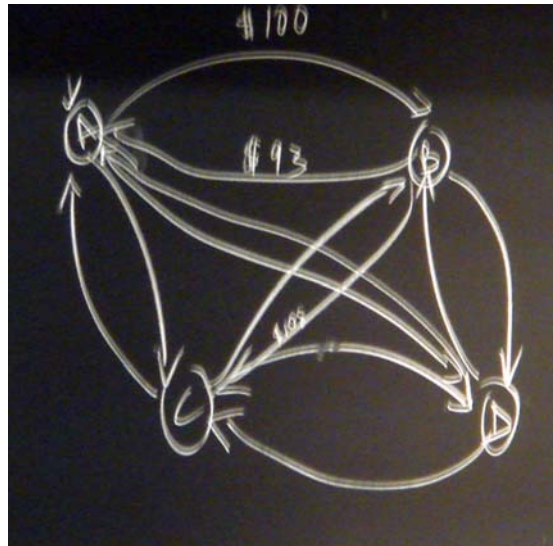
Thus the 3-SAT problem is polynomial-time reducible to the Direct Hamiltonian Cycle problem.

We have been talking about decisions problems. That is, problems that given an input can output “yes” or “no.” But there are other types:

MTS (Minimum Traveling Salesman problem):

Input: A weighted complete directed graph G

Output: k such that there exists a Hamiltonian cycle of G of weight k and there does not exist a Hamiltonian cycle of G of weight less than k .



Conjecture hypothesis

$NP \neq P$

MTS (Min Traveling Salesman problem):

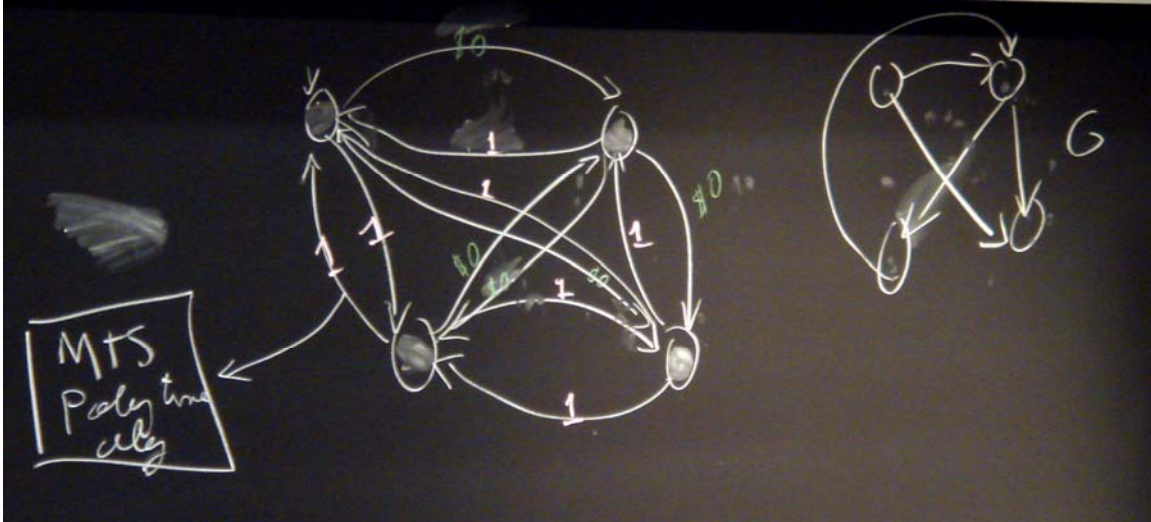
INPUT weighted complete directed graph G

OUTPUT K st \exists a hamiltonian circuit in G of weight K , & there does not exist a hamiltonian circuit in G of weight less than K .

Consider algorithm B:

On input G , a directed graph, create H , a directed graph with the same nodes as G but H is complete. Those edges that are in G have 0 weight in H , and all the rest of the edges have weight 1 each.

Clearly, B runs in polynomial time.



If a polynomial-time algorithm for MTS exists (call it C), then given a Directed Hamiltonian Cycle problem inputs G into H , inputs to C . If C says there exists a 0-weight Hamiltonian Cycle in H , then there must have existed a Hamiltonian Cycle in G . Alternatively, if C says the minimum-weight Hamiltonian Cycle in H has weight > 0 , then there must have been no Hamiltonian Cycle in G . Thus, $P \neq NP \Rightarrow$ there does not exist a polynomial-time algorithm for MTS.