6.890: **Algorithmic Lower Bounds: Fun With Hardness Proofs** Fall 2014

Lecture 7 Scribe Notes

Prof. Erik Demaine

1 Overview

In this (probably) final lecture about proving hardness using 3SAT, we discuss many variants of planar 3SAT and some related problems on planar graphs.

2 Planar 3SAT

2.1 Planar 3SAT = 3SAT

Planar 3SAT is a special case of 3SAT in which the bipartite graph of variables and clauses is planar (i.e., no edge crossings). We create an edge (v_i, c_j) between variable v_i and clause c_j whenever v_i or $\bar{v}_i$ is in c_j. We distinguish positive and negative literals by edge colors [Lichtenstein 1982].

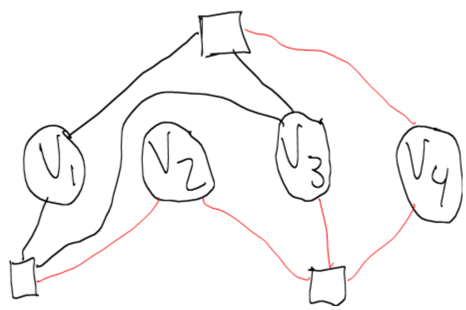
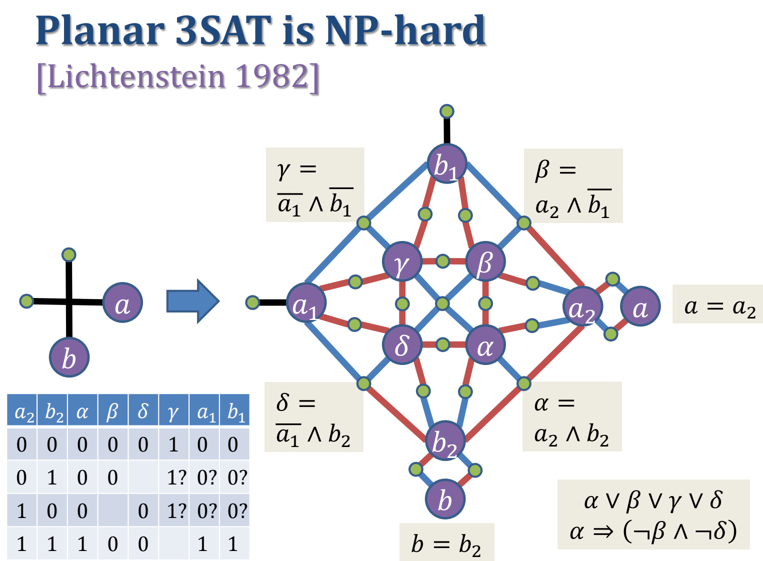


Figure 1: The variable-clause graph for $(v_1 \vee v_3 \vee \bar{v}_4) \wedge (v_1 \vee \bar{v}_2 \vee v_3) \wedge (\bar{v}_2 \vee \bar{v}_3 \vee \bar{v}_4)$. Circular nodes represent variables, square nodes represent clauses, black edges represent positive literals, and red edges represent negative literals.

Deciding whether a graph is planar and finding a planar embedding can both be done in linear time. This is important because we usually need to know the embedding for reductions because the instances of our problem that our reduction produces mirror the structure of a planar embedding of the 3SAT instance.

2.1.1 Proof of Planar 3SAT from 3SAT

We first construct the crossover gadget in our reduction from 3SAT to Planar 3SAT. The construction of the gadget is as follows: create a graph containing nodes representing variables and clauses as described above and connect it in the way shown in lecture. In this gadget, the small green



a_2	b_2	α	β	δ	γ	a_1	b_1
0	0	0	0	0	1	0	0
0	1	0	0		1?	0?	0?
1	0	0		0	1?	0?	0?
1	1	1	0	0		1	1

Figure 2: Planar 3SAT crossover gadget

circles represent clauses and big blue circles are variables. The blue connections are positive while red are negative. Because the graph is bipartite, we can't disrupt planarity by the connections between variables and clauses.

If we work through the logic of the gadget starting from assigning a truth value to a and b, we see that $a = a_1$ and $b = b_1$ as expected.

We see there is a slight glitch in our gadget. Some clauses contain 4 variable and some contain only 2 variables so the premises of 3SAT are not followed exactly in the gadget shown. We can easily fix both situations. For a clause that only contains 2 variables, we can either create parallel edges from one variable to the clause or create a separate false variable to include in the constraint. For the 4 variable clause, we borrow the reduction from 4SAT to 3SAT to create a gadget that involves introducing an extra variable set to false which represents whether the left or right side of the original clause contained the satisfying literal.

The purpose of the crossover gadgets is to ensure that the graph remains planar if we connect all the variable in a cycle [Dyer & Frieze 1986]. The crossover gadgets are used for the global crossings for larger gadgets of variables. Where there is an intersection between the cycle and a variable/clause edge, we add a crossover gadget. To separate the different crossover gadgets, we have the a_2 and b_2 variables as shown in the original gadget. The proof that we present will be for Planar 3SAT with all variables connected. Solving 3SAT results in a solution for the Planar 3SAT in this case.

There are variables of the problem that still maintain planarity of Planar 3SAT and will help us with reductions later on.

Planar 3SAT graph remains planar if variable nodes are exploded into literal nodes (with edges between complimentary literals). This means positive literal connections and negative literal connections are contiguous on the variable nodes. There is a bipartite between the variables once all the literals are connected together as shown below.

The problem also remains planar if we require all positive connections to be on one side ("side"

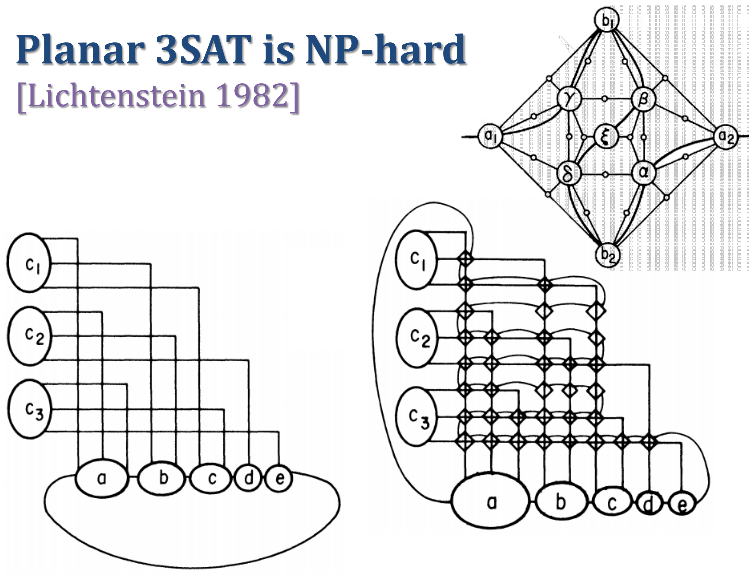


Figure 3: Planar 3SAT with a cycle through variable nodes. Top right: the path through the crossover gadget. Left: the initial arrangement of the bipartite graph, with the variables in a cycle. Bottom right: the arranged graph, with crossover gadgets inserted to preserve planarity; note that some gadgets are required to cross the path across an edge rather than to cross two edges over.

being based on the division of the plane in the planar embedding by the cycle) of the cycle and all negative connections on the other. This results in a problem that is monotone and in a stronger statement of the original problem [de Berg & Khosravi 2010].

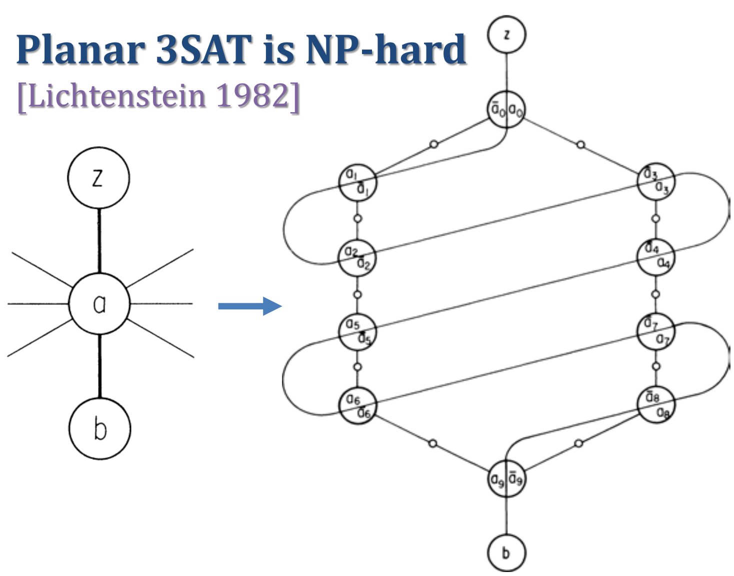


Figure 4: Planar 3SAT with variables divided.

2.2 Planar rectilinear 3SAT

Each variable is a horizontal segment on the x-axis while each clause is a horizontal segment above the x-axis with vertical connections to the 3 variables it includes on the x-axis. The connections can either be positive or negative [Knuth & Raghunathan 1992].

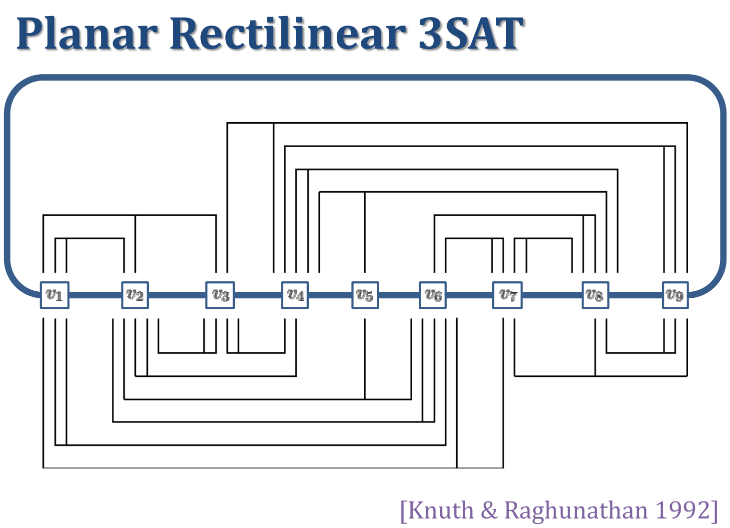


Figure 5: Planar rectilinear 3SAT.

2.3 Planar monotone rectilinear 3SAT

This is a variation of planar rectilinear 3SAT where all positive clauses are above the x-axis and all negative clauses are below the x-axis [de Berg & Khosravi 2010].

Rectilinearity doesn't change much; it is just a nice way to draw regular planar graphs. We may prove that any planar graph with the v_i cycle can be drawn as a rectilinear graph by using nesting. However, rectilinearity helps with reductions on a grid.

Surgeon General's warning: if all clauses are inside the variable cycle (both positive and negative edges) and the graph is planar, then the problem is polynomial-time. The intuition behind solving this problem is by creating a tree of the nesting structure of the clauses (clauses underneath other clauses) and use dynamic programming on the tree.

A special case to the above is when all clauses are also connected in a path, then the problem is still in P because we can show that this implies that clauses are all in one side of the variable cycle.

This has various implications for hardness proofs we've seen in this class. If we can get rid of the connections between clauses, then we don't need the crossover gadget because Super Mario Bros could just be reduced from planar 3SAT. In the case of Super Mario, we can't do this because we can't eliminate the connections between clauses.

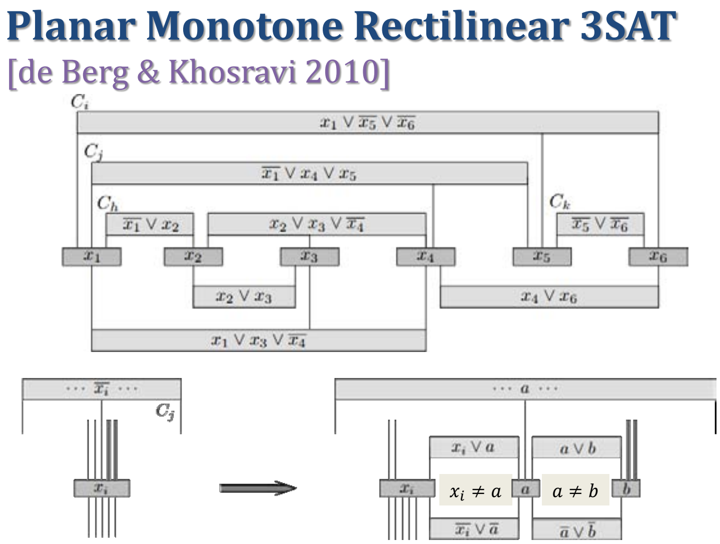


Figure 6: Planar monotone rectilinear 3SAT.

2.4 Planar 1-in-3SAT

We can provide a reduction from Planar 3SAT when graphs are planar. To preserve planarity, we take every 3SAT clause and transform into a corresponding 1-in-3SAT clause and layer different clauses one on top of the other. The dashed line on the border of the gadget represents that the gadget preserves planarity.

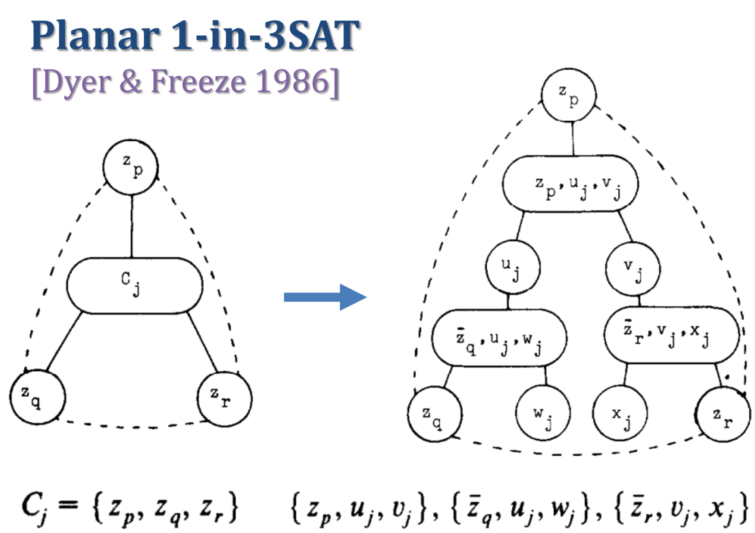


Figure 7: Planar 1-in-3SAT reduction transformation.

2.5 Planar positive 1-in-3SAT

This is a stronger version of Planar 1-in-3SAT and is also NP-hard. For this problem, we require that every connection is positive.

2.5.1 Planar positive rectilinear 1-in-3SAT

The construction is the same as planar rectilinear 3SAT except we no longer allow negative connections. We can reduce from planar rectilinear 3SAT, removing negations with equal and not all equal gadgets and expanding clauses to exactly three variables [Mulzer & Rote 2006].

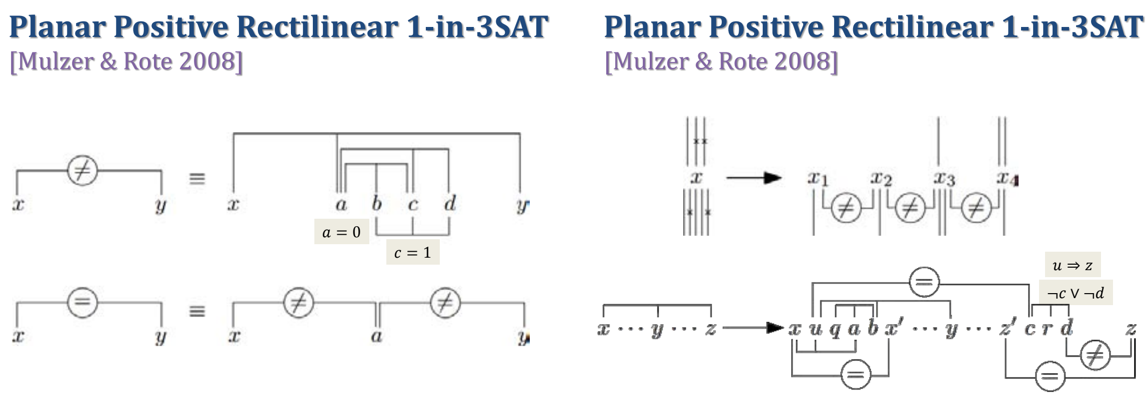


Figure 8: Planar positive rectilinear 1-in-3SAT equals and not-equals gadgets and transformation.

2.6 Planar NAE 3SAT

This problem can surprisingly be solved in polynomial time by reduction *to* planar max cut. The intuition for the method to solve the problem is to make colors alternate between black/white for the maximum cut. Exactly when the variables are not all equal, then we may specify a target with our cut [Moret 1988].

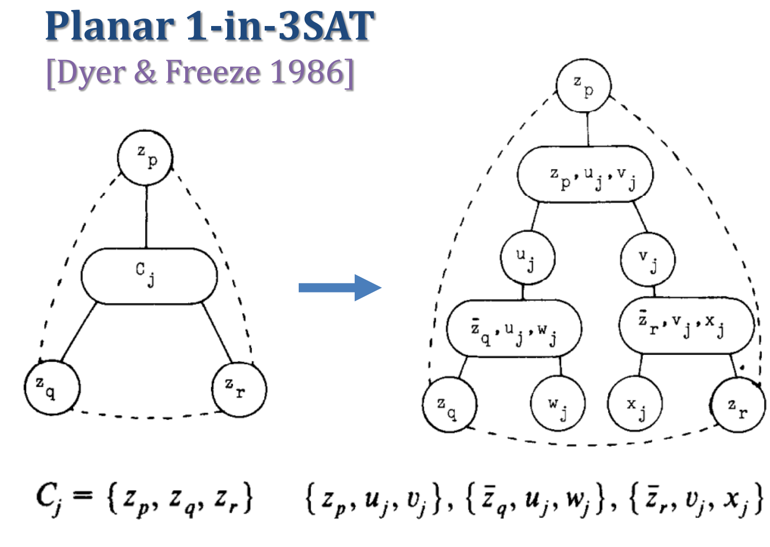


Figure 9: Planar NAE 3SAT gadgets.

3 Other Problems

3.1 Planar X3C (exact cover with 3-sets)

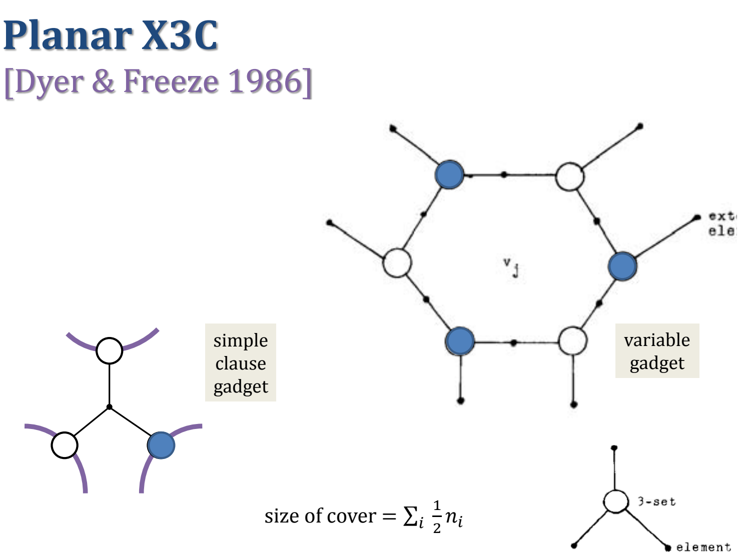


Figure 10: Planar X3C reduction gadgets (simple version). See the Planar 3DM gadgets for the complex version (just ignore the colors).

Planar X3C involves a bipartite graph between sets and variables where sets have degree 3 but variables have arbitrary degree. Variables may appear in at most 3 sets. The induced planar graph solves Planar X3C if the sets cover all variables.

Planar X3C is the dual of Planar 1-in-3SAT in the sense that in X3C the variables should be exactly covered by the sets (which have fixed degree) where as in 1-in-3 the clauses are exactly covered by the variables (which have arbitrary degree).

We can show that Planar X3C is NP-hard by reducing from Planar 1-in-3SAT [Dyer & Frieze 1986]. Each variable gadget consists of 3-sets connected in an even cycle by elements (one element per set left free to connect to clauses); this gadget forces any covering to choose one of the two alternations. The clause gadget is just an element connected to the three sets in variable gadgets representing its literals. Exactly one of the sets in the clause can cover the variable, representing the one true literal in that clause. (The reduction in the reference uses a more complex clause gadget and connector gadgets to set up for the Planar 3DM reduction described below.) This reduction maintains an element cycle.

3.2 Planar 3DM

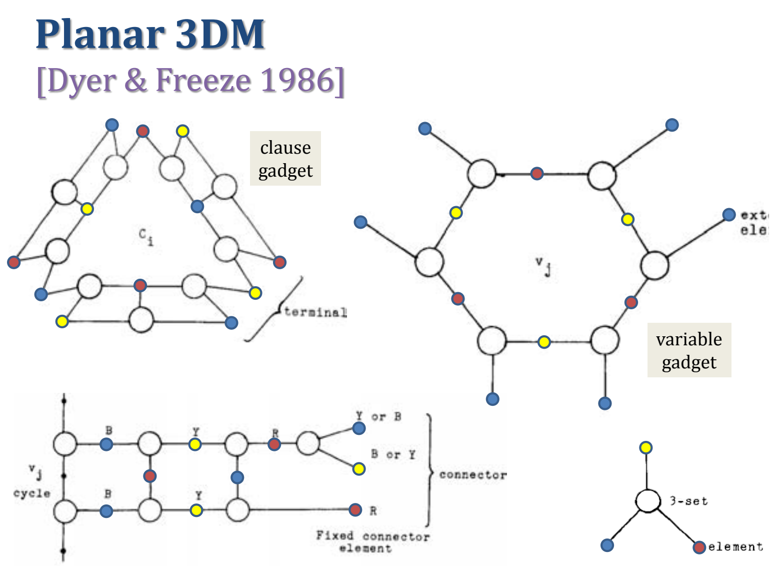


Figure 11: Planar 3DM 1-in-3SAT gadgets. The connector gadget has other possible colorings.

Planar 3DM is a special case of Planar X3C in which each element is one of three colors and each set must cover exactly one element of each color. The reduction is correspondingly similar to the Planar X3C reduction, except with a more complex clause gadget and various connector gadgets to ensure any variable can connect to any clause without violating the coloring requirement. Again, the gadgets maintain an element cycle to keep the graph planar [Dyer & Frieze 1986].

3.3 Planar Vertex Cover

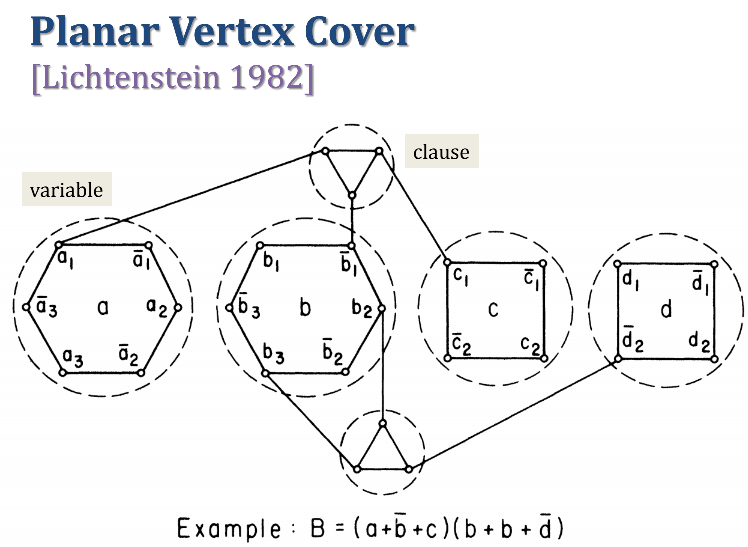


Figure 12: Planar vertex cover gadgets.

In Planar Vertex Cover, given a planar graph, we choose a set of k vertices that cover all the edges. (Each vertex covers all edges incident on it.) The cover need not be exact.

We can reduce from Planar 3SAT [Lichtenstein 1982]. Variable gadgets are an even cycle, forcing a covering to choose one of the two alternations (representing true and false) to cover the cycle. Clause gadgets are triangles connected to the variables on odd or even nodes as appropriate for positive or negative literals.

3.4 Planar (Directed) Hamiltonian Cycle

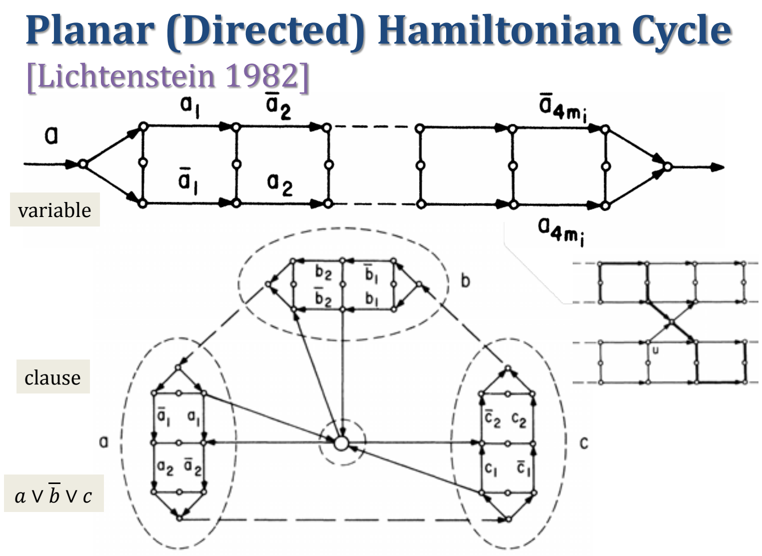


Figure 13: Planar Hamiltonian cycle gadgets.

In Planar Hamiltonian Cycle, given a planar directed graph, we must find a path that visits every vertex exactly once. In order to reduce from Planar 3SAT, we use alternating-wire gadgets representing variables. Each wire has two sides, where the side representing true alternates at each vertex; the overall shape resembles a ladder with triangles at each end. These wire gadgets are connected to clause vertices from the true (respectively false) side for positive (negative) literals. The path will be on the correct side of the alternating wire to visit a clause vertex exactly when the corresponding literal satisfies the clause. Attempts to cross between wires using a clause vertex will always result in one vertex on the wire becoming unreachable unless other vertices are used more than once. This reduction also works for undirected graphs [Lichtenstein 1982].

3.5 Shakashaka

Figure 14: Shakashaka puzzle encoding a planar 3SAT instance using variable, clause and wire gadgets.

This game invented by Nikoli can be proven to be NP-Hard by a reduction from Planar 3SAT.

The game consists of board of white squares and black squares with numbers. The numbers in black squares specify how many half-filled white squares it has as its neighbors. The goal is to half-fill squares such that all black squares with numbers are satisfied and the white regions you create are rectangular (including at a 45 degree angle from the grid). The player is not allowed to completely fill squares.

There are several gadgets needed in the reduction from Planar 3SAT. These gadgets include a wire, clause, and parity shift gadget. The role of the parity shift gadget is to provide 2 possible different configurations which shift the parity in different ways [Demaine et. al. 2014].

3.6 Flattening Fixed-Angle Chains

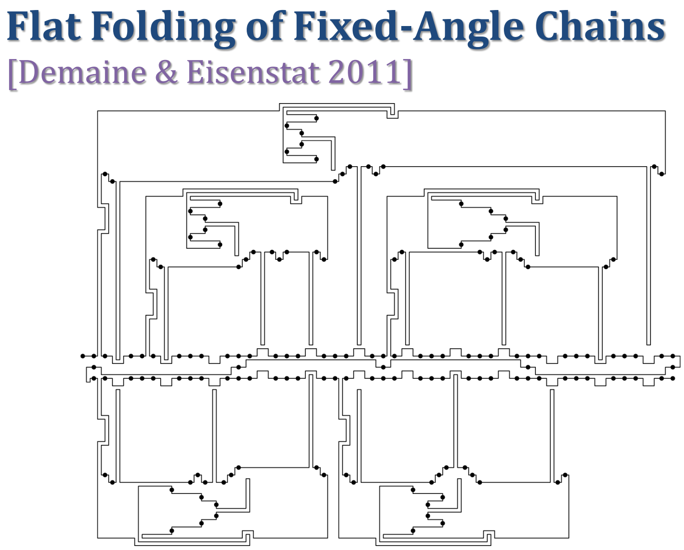


Figure 15: Fixed-angle chain-flattening instance encoding a planar monotone rectilinear 3SAT instance.

This problem is strongly NP-Hard from a reduction from planar monotone rectilinear 3SAT. The problem is to decide whether a polygonal chain with fixed edge lengths and angles has a planar configuration without crossings.

We can create a gadget such that all variables are in a chain. A clause has three possible positions, one each with a protrusion at a point corresponding to a variable, thus forcing the variable chain to take one position or the other. The positions determine the value of the variables within the clause [Demaine & Eisenstat 2011].

References

[Lichtenstein 1982] David Lichtenstein. Planar Formulae and Their Uses. *SIAM J. Comput.*, 11(2), 329-343, 1982.

[Knuth & Raghunathan 1992] Donald E. Knuth and Arvind Raghunathan. The problem of compatible representatives. *SIAM J. Discrete Math*, 5(3), 422-427, 1992.

[de Berg & Khosravi 2010] Mark de Berg and Amirali Khosravi. Optimal Binary Space Partitions in the Plane. *Lecture Notes in Computer Science*, 6196, 216-225, 2010.

[Dyer & Frieze 1986] M.E Dyer and A.M Frieze. Planar 3DM is NP-complete. *Journal of Algorithms*, 7(2), 174-184, 1986.

[Mulzer & Rote 2006] Wolfgang Mulzer and Guenter Rote. Minimum-weight triangulation is NP-hard. *Journal of the ACM*, 55(2), No.11, 2006.

[Moret 1988] B. M. E. Moret. Planar NAE3SAT is in P. *ACM SIGACT News*, 19(2), 51-54, 1988.

[Demaine et. al. 2014] Erik D. Demaine, Yoshio Okamoto, Ryuhei Uehara, and Yushi Uno. Computational complexity and an integer programming model of Shakashaka. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E97-A(6), 1213-1219, 2014.

[Demaine & Eisenstat 2011] Erik D. Demaine and Sarah Eisenstat. Flattening Fixed-Angle Chains Is Strongly NP-Hard. *Proceedings of the 12th Algorithms and Data Structures Symposium (WADS 2011)*, 314-325, 2011.

6.890 Algorithmic Lower Bounds: Fun with Hardness Proofs
Fall 2014