ASSIGNMENT : HW11

NAME: PRASANTH BHAGAVATULA

CWID : A20355611

① In order to prove, that 2-CNF-SAT problem belongs to the class P, we need to prove two things. A solution to the problem can be found in polynomial time and ~~the~~ a solution can be verified in a polynomial time.

For the secondly part, we can start from the beginning of expression and evaluate each and every clause by assigning the truth values and compute the truth value of the operation and so, this verification of a solution can be done in linear time.

For the first part*, we need to construct an implication graph using the below equivalence relation.

$$a \vee b \equiv \neg a \Rightarrow b \equiv \neg b \Rightarrow a$$

In an implication graph, there will be two vertices ~~for each and~~ corresponding to each and every variable, one for ~~its~~ itself and the other for its negation. Now based on the "implication", a directed edge will be present between "u" and "v" if "u ⇒ v". Hence, using the above mentioned equivalence relation, we can construct the implication graph.

---

* Referred from wikipedia, ~~2CNF~~ 2-satisfiability, ~~section~~ algorithm by Aspvall, Plass and Tarjan

Based on the transitive property of implication, ~~if~~

If $a \Rightarrow b$ and $b \Rightarrow c$ then $a \Rightarrow c$

Hence, if $a = \bar{a}$, then it means $a \Rightarrow \bar{a}$, which is never possible and so, there cannot be a directed path between a vertex and its own negation vertex in this implication graph. If such path exists, it means there is no solution to this 2-CNF..

In order to find the solution, we need to divide the set of vertices into strongly connected components. Two vertices belong to the same strongly connected component, if and only if there is a directed path from one to the other and vice-versa. Once, the graph is sub-divided, we can condense, the strongly connected components, replacing them with single ~~into a single vertex~~ vertex. The resulting graph, will be acyclic and now starting from reverse topological order, we can assign the truth value "true" to the components, if the truth value is not yet assigned.

~~Fi~~ Construction of the implication graph is dependent on the number of variables ~~(or number)~~ as the graph has twice the number of vertices and so the edges are dependent on the number of ~~clauses~~ clauses. So, the graph can be constructed in linear time.

~~Finding on path exist~~

Finding the strongly connected components ~~ad to~~ has many algorithms and using the best known linear time algorithm, we can construct the components ~~in~~ linear time.

Checking for existence of a ~~vert~~ variable and its negation with in the same strongly connected component ~~also~~ takes polynomial ~~linear~~ time as we just need to pass through all the vertices of strongly connected component looking for ~~its~~ its ~~also~~ negation and the same has to be repeated for every vertex in the component.

Finally condensing and assigning truth values in reverse topological order takes only linear time and so, we can find a solution to 2-CNF-SAT problem in polynomial time.

Hence, the 2-CNF-SAT belongs to the ~~set~~ class P.

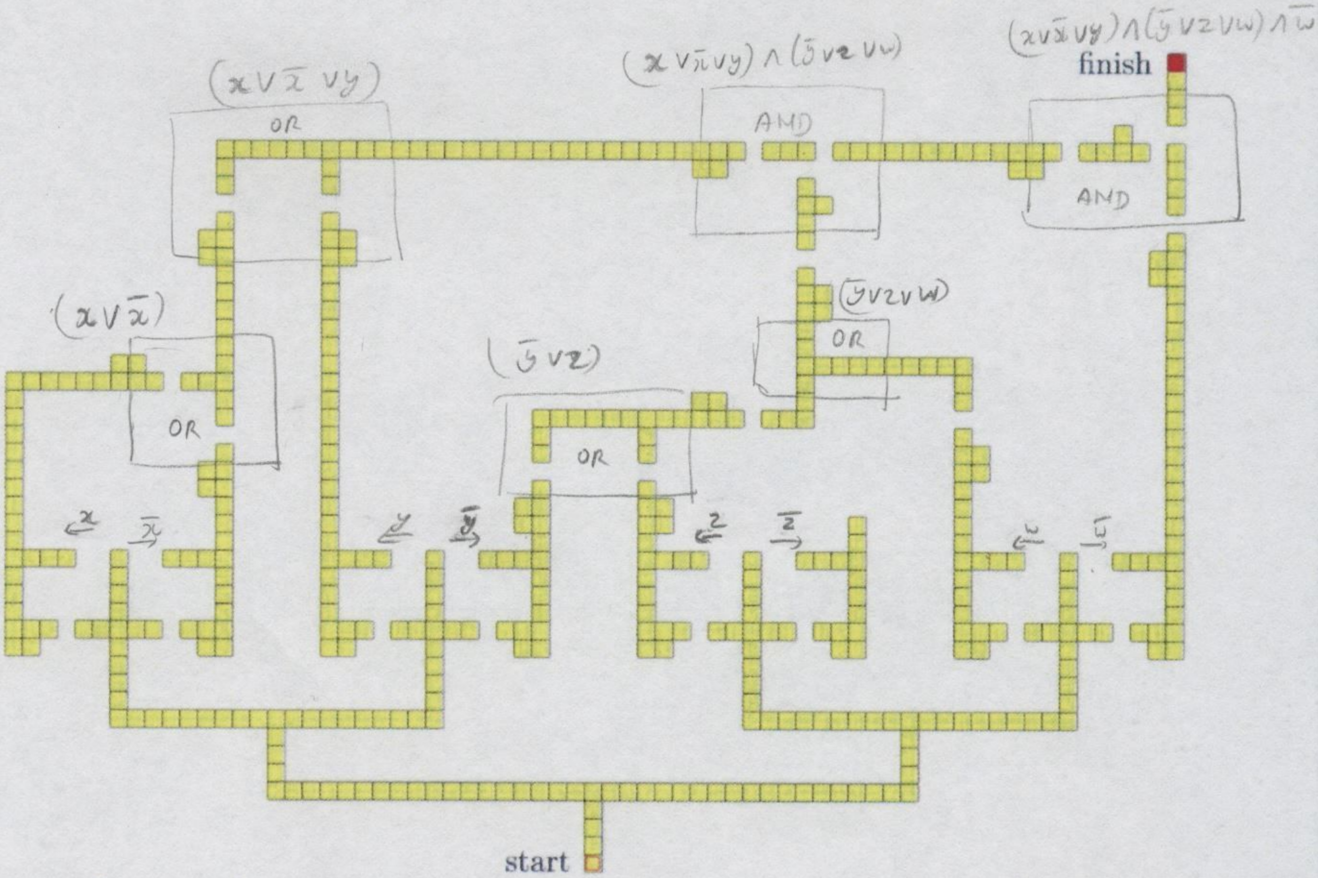


Figure 10. TipOver puzzle for $((x \vee \bar{x} \vee y) \wedge (\bar{y} \vee z \vee w) \wedge \bar{w})$.

ing that it is also no harder than SAT, i.e., that TipOver is itself an NP problem. A problem is in NP if it can be solved nondeterministically (allowing lucky guesses) in polynomial time, so a simple litmus test for an NP problem is this: if you are given a proposed solution, can you verify it in polynomial time? In TipOver, you can certainly verify it quickly enough: each crate is tipped at most once, and it's easy to check that each successive move of a purported solution is legal, and that the target crate is finally reached. Therefore, TipOver is in the class NP, and thus is NP-complete.

Beyond NP-Complete

Earlier I mentioned Rush Hour, another puzzle made by ThinkFun. In Rush Hour, cars and trucks slide backwards and forwards on a grid; a particular target car has to escape the grid. Perhaps Rush Hour is also NP-complete? Actually, it is (we think) even harder! The problem is that our method for verifying a proposed solution in polynomial time doesn't work for Rush Hour. Cars can move backwards and forwards many times, unlike crates, which can only tip over once. This has the effect that there can be solution sequences which require exponentially many moves, so we can't check such sequences in polynomial time. It turns out that Rush Hour is PSPACE-complete [1]. (We can solve it using polynomial *space*.) Just as we're not positive that NP-complete problems are harder than ones solvable in polynomial time, we're not positive that PSPACE-complete problems are harder than NP-complete problems. But in both cases, the smart money says yes.

In any case, it is a curious fact that many, if not most, interesting games and puzzles seem to be NP-complete or harder. It seems as though the very features that make puzzles challenging also tend to give them a kind of computational power, which is reflected in their formal complexity. What, if anything, this says about the nature of intelligence, and the appeal of puzzles, is an interesting question.

REFERENCES

[1] Gary William Flake and Eric B. Baum. *Rush Hour* is PSPACE-complete, or "Why you should generously tip parking lot attendants." *Theoretical Computer Science*, 270(1–2):895–911, January 2002.

[2] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, 1979.

[3] Robert A. Hearn. Amazons, Konane, and Cross Purposes are PSPACE-complete. In R. J. Nowakowski, editor, *Games of No Chance 3*, 2006. To appear.

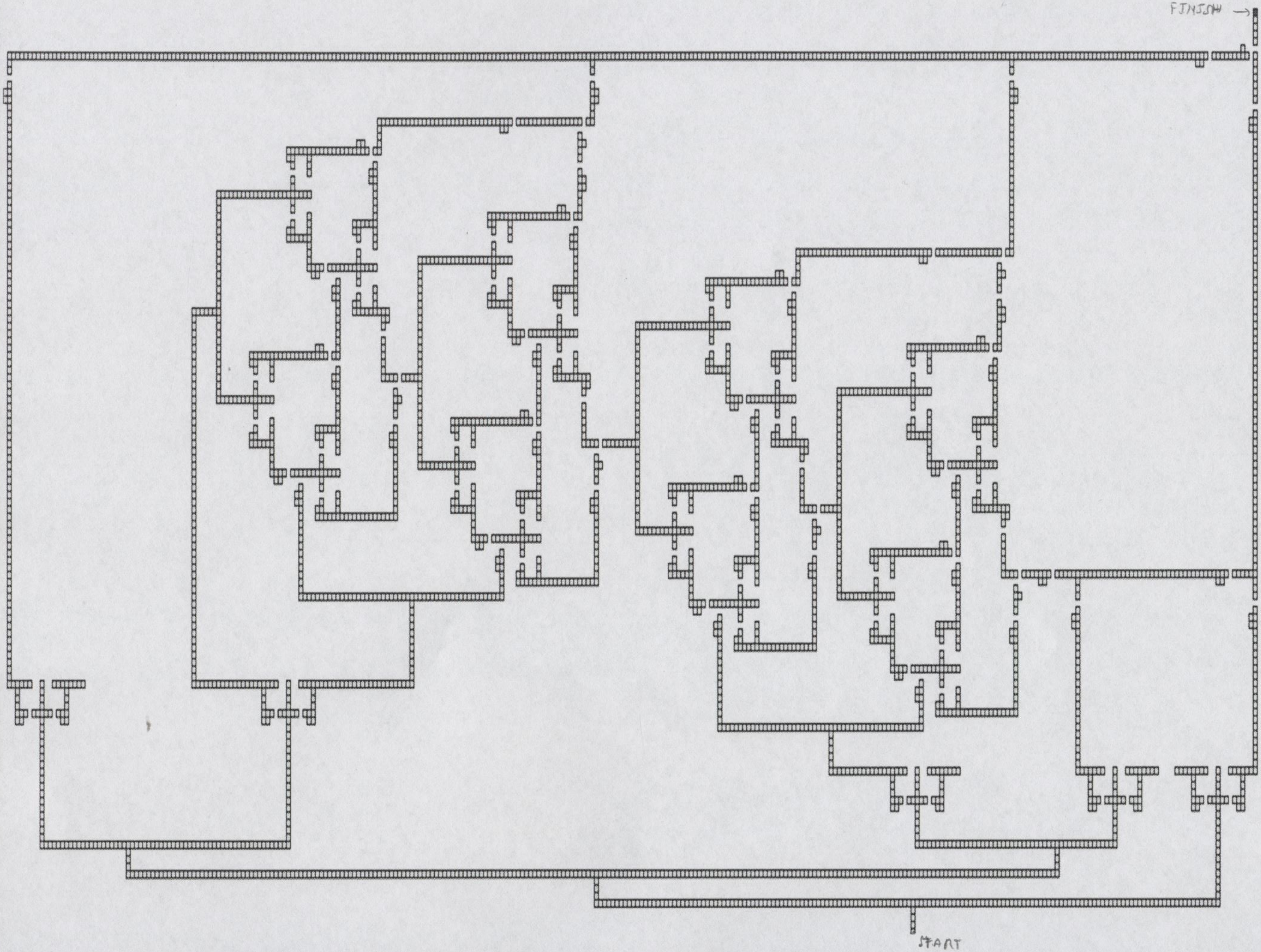
[4] Robert A. Hearn. *Games, Puzzles, and Computation*. PhD dissertation, Massachusetts Institute of Technology, 2006. To appear.

[5] Richard Kaye. Minesweeper is NP-complete. *Mathematical Intelligencer*, 22(2):9–15, 2000.

[6] David Lichtenstein. Planar formulae and their uses. *SIAM J. Comput.*, 11(2):329–343, 1982.

[7] James W. Stephens. The kung fu packing crate maze. http://www.puzzlebeast.com/crate/ index.html.

MIT Computer Science and Artificial Intelligence Laboratory
Cambridge, MA 02139
USA
e-mail: rah@csail.mit.edu



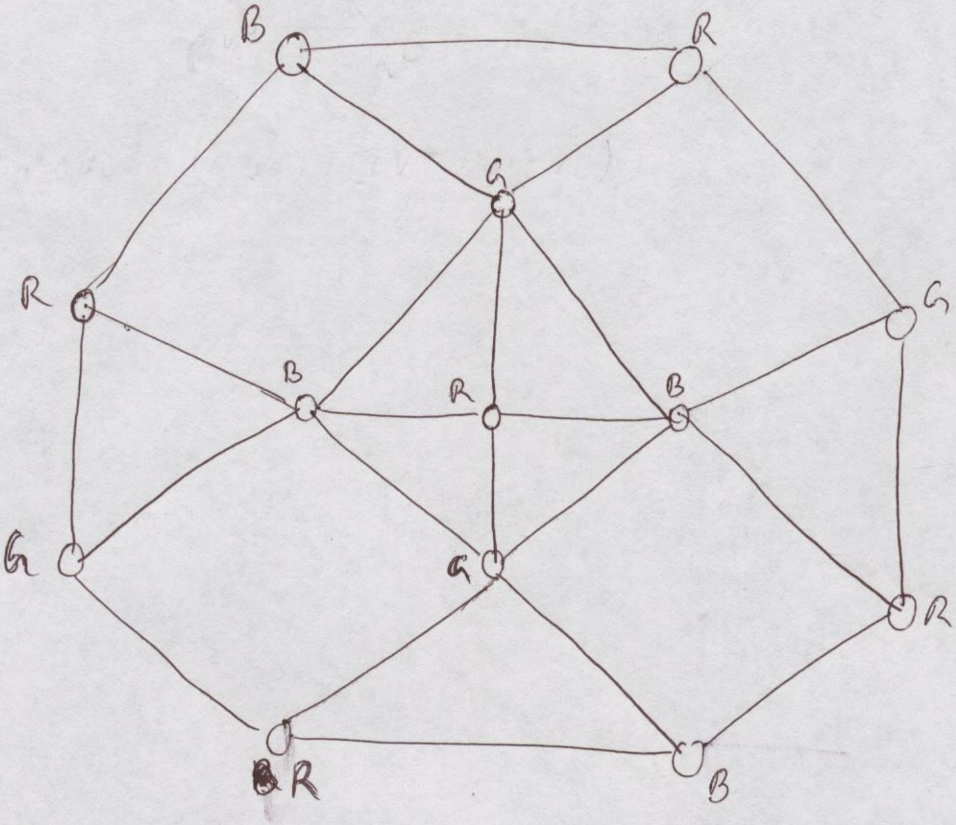
FINISH →
START

ASSIGNMENT: HW1

NAME: PRASANTH BHAGAVATULA

CWID: A20355611

(3) We need to show that the graph - ~~Gcro~~ i.e. the cross over object can be 3-colourable. ~~Ief we cle~~



As, you can see, the above colouring shows that the opposite edges ~~can be~~ will be of same colour and the entire graph is 3-coloured. So, using this crossover object, we can connect the edges of vertices from different colours ~~to~~ of a non-planar graph to the respective vertex and pull out an edge from the opposite end.