Some remarks on the Clique problem

Thinh Nguyen

Moscow State University
kosmofarmer@sgmail.com

June 22, 2018

Abstract

We survey some information about the clique problem - one of the cornerstone of the field of research in theory of algorithms. A short note at the end of the paper should be read with care on the reader's own.

I. Introduction

An Independent Set in a graph is a set of nodes no two of which have an edge. E.g., in a 7-cycle, the largest independent set has size 3, and in the graph coloring problem, the set of nodes colored red is an independent set. The Independent Set problem is: given a graph G and an integer k, does G have an independent set of size âL'ě k?

We reduce from Max-Clique. Given an instance (G, k) of the Max-Clique problem, we output the instance (H, k) of the Independent Set problem where H is the complement of G. That is, H has edge (u,v) iiňĂ G does not have edge (u,v). Then H has an independent set of size k iiňĂ G has a k-clique.

II. More on cliques

As mentioned earlier, it is an open problem whether P 6= NP (though everyone believes they are diiňĂerent). It is also an open problem whether NP 6= co-NP (though again, everyone believes they are diiňĂerent). One can also deïňĄne even more expressive classes. For instance, PSPACE is the class of all problems solvable by an algorithm that uses a polynomial amount of memory. Any problem in NP is also in PSPACE, because one way to solve the problem is to take the given instance I and then simply run the veriiňĄer V (I,X) on all possible proof strings X, halting with YES if any of the runs outputs YES, and halting with NO otherwise. (Remember, we have a polynomial upper bound on |X|, so this uses only a polynomial amount of space.) Similarly, any problem in co-NP is also in PSPACE. Unfortunately, it is not even known for certain that P 6= PSPACE (though all the classes mentioned above are believed to be diiňĂerent). One can also deïňĄne classes that are provably larger than P and NP. For instance, EXPtime is the class of all problems solvable in time O(2nc) for some constant c. This class is known to strictly contain P. The class NEXPtime is the class of all problems that have a veriiňĄer which runs in time O(2nc) for some constant c. This class is known to strictly contain NP. The class of all Turing-computable problems is known to strictly contain all of the above, and some problems such as the Halting problem (given a program A and an input x, determine whether or not A(x) halts) are not even contained in that!

NP-complete problems have the dual property that they belong to NP and they capture the essence of the entire class in that a polynomial-time algorithm to solve one of them would let you solve anything in NP. We proved that 3-SAT is NP-complete by reduction from Circuit-SAT. Given a circuit C, we showed how to compile it into a 3-CNF formula by using extra variables for each gate, such that the formula produced is satisïňĄable

if and only if there exists x such that C(x) = 1. This means that a polynomial-time algorithm for 3-SAT could solve any problem in NP in polynomial-time, even factoring. Moreover, 3-SAT is a simple-looking enough problem that we can use it to show that many other problems are NP-complete as well, including Max-Clique, Independent Set, and Vertex Cover.

III. Results

What do these two conditions mean? Well (1) implies that the literals zab,zcd corresponding to the vertices vab,vcd respectively belong to diïňĂerent clauses Ca 6= Cc in ÏĘ. The second condition implies that both literals can be satisïňĄed simultaneously. This step of the construction takes O(m2) time. The ïňĄnal step is to determine the value of k; we will set k to be m, the number of clauses in ÏĘ. Now I claim that ÏĘ is satisïňĄable if and only G as constructed above has a clique of size at least k = m.

IV. Discussion

The set S = v1,v2,...vm must form an m-clique in G. Why? Well notice that zâĹů 1,zâĹů 2,...,zâĹů m all have the same truth assignment, since otherwise zâĹů i = ÂňzâĹů j for some i,j âĹĹ [1,m] thus implying that one of zâĹů i and zâĹů j is not the satisfying literal of Ci,Cj, a contradiction to our choice of the zâĹů literals. Notice also that the zâĹůâĂŹs belong to diïňĂerent clauses, thatâĂŹs how we chose them. Therefore, by the construction of G, every pair of v1,...,vn must have a connecting edge and thus S = v1,v2,...vm forms an m-clique in G. Conversely, suppose G has a clique of size at least m = k. Let v1,v2,...,vq be a clique in G of size q âL'ě m, then the ïňĄrst m vertices v1,...,vm must also form a clique in G. Since there are no edges connecting vertices from the same clause, every vi corresponds to a literal zi from exactly one clause Ci. Moreover, since v1,...,vm is a clique, the corresponding literals zi,zj of any pair vi,vj âĹĹ v1,...,vm can be satisïňĄed simultaneously (by construction). Now, to construct a satisfying assignment x1,...,xn for ÏĘ, we just need to satisfy all of z1,...,zm and assign the remaining variables arbitrarily. Every Ci contains one zi, and every zi is satisïňĄed thus every Ci is satisïňĄed and so ÏĘ is satisïňĄed. To conclude, weâĂŹve shown that CLIQUE is in NP and that it is NP-hard by giving a reduction from 3-SAT. Therefore CLIQUE is NP-complete.

V. Nothing

The Maximal Clique Problem

Although many theoretical papers have been published on DNA computing since AdlemanâĂŹs first crude demonstration in 1994, it would be over two years, in 1997, until another NP complete problem would be solved. This problem was the Maximal Clique Problem: given a group of vertices some of which have edges in between them, the maximal clique is the largest subset of vertices in which each point is directly connected to every other vertex in the subset. Every time a new point is added, the number of total cliques that must be searched at least doubles; hence we have an exponentially growing problem. Once again, researches sought to take advantage of DNAâĂŹs high level parallelism which would essentially allow all the possible paths to be calculated simultaneously. After all the possible cliques were constructed, the scientists would simply need to fish out the largest clique. However, like many DNA experiments, each possible âĂIJchoiceâĂİ needed a unique DNA strand. This is a problem because, as noted early, the number of cliques grows exponentially. In AdlemanâĂŹs experiment, each city and every connecting path had to be âĂIJhard-coded,âĂİ in other words, specially made to order in a biology lab. In a problem with more cities, it would be virtually impossible to manually create every possible path. Ouyang and company designed an algorithm that would only require the manual creation a linearly growing number of DNA strands, 2N to be exact, where âĂŸNâĂŹ is the number of ver-

tices in the graph.

VI. Conclusion

In a (possibly) finite universe, anyone can thrive if (s)he knows about the secret of nature. Whichever is of philosophical concerns is in another dimensional spacey.