

1 Problemes Tractables vs Problemes Intractables

1. **2-color vs 3-color.** Recordeu que un graf no dirigit $G = (V, E)$ és *bipartit* si existeix una partició de V en V_1, V_2 de manera que tota aresta $uv \in E$ satisfà o bé $u \in V_1 \wedge u \in V_2$ o bé $u \in V_2 \wedge u \in V_1$. Fixeu-vos que dir que un graf és bipartit és equivalent a dir que és 2-colorable (2-COLOR). Definim $\text{BIPARTITE GRAPH} = \{\langle G \rangle \mid G \text{ és bipartit}\}$.

Demostreu que $\text{BIPARTITE GRAPH} \in \text{P}$.

2. **DNF-Sat vs CNF-Sat.** Demostreu que el problemes següents pertanyen a la classe P:

- (a) Donada una fórmula booleana en *Forma Normal Conjuntiva (CNF)* on cada clàusula té com a màxim dos literals, decidir si és satisfactible.
- (b) Donada una fórmula booleana en *Forma Normal Disjuntiva (DNF)*, decidir si és satisfactible.
- (c) Una fórmula booleana en CNF es pot transformar en una fórmula booleana en DNF equivalent? Si es pot, en quant temps?

3. **Eulerian graph vs Hamiltonian graph** Recordeu que un *recorregut eulerià* en un graf no dirigit és un cicle que pot visitar repetides vegades un mateix vèrtex i ha d'utilitzar cada aresta exactament una vegada. Diem que un graf és *eulerià* si conté un recorregut eulerià.

Demostreu que

$$\text{EULERIAN GRAPH} = \{\langle G \rangle \mid G \text{ és eulerià}\} \in \text{P}.$$

Dissenyeu un algorisme eficient que en el cas que el graf sigui eulerià ens retorni un recorregut eulerià.

4. **Shortest path vs longest path** Considerem les dues versions següents de *Shortest Path* i *Longest Path*, respectivament, per a grafs no dirigits:

$$\text{SPATH} = \{\langle G, a, b, k \rangle \mid G \text{ conté un camí simple de } a \text{ a } b \text{ de longitud menor o igual que } k\}$$

$$\text{LPATH} = \{\langle G, a, b, k \rangle \mid G \text{ conté un camí simple de } a \text{ a } b \text{ de longitud més gran o igual que } k\}.$$

- (a) Demostreu que $\text{SPATH} \in \text{P}$.
- (b) Demostreu que $\text{LPATH} \in \text{NP}$. Demostreu que si $\text{LPATH} \in \text{P}$ aleshores el problema del camí Hamiltonià en graf no dirigits també seria a P.

5. Consider the problem MODULAR EXPONENTIATION: given $a, b, n \in \mathbb{N}$, compute $d = a^b \bmod n$. Can MODULAR EXPONENTIATION be solved in polynomial time?

6. **Half clique.** Considereu el problema SUBGRAF COMPLET D'ORDRE MEITAT (HALF-CLIQUE): Donat un graf no dirigit G , decidir si conté una $\lfloor \frac{n}{2} \rfloor$ -clique.

- (a) Demostreu que $\text{HALF-CLIQUE} \in \text{NP}$
- (b) Demostreu que si $\text{HALF-CLIQUE} \in \text{P}$, aleshores $\text{CLIQUE} \in \text{P}$.

7. **Self reducibility SAT.** Supposem que tenim un algorisme que decideix SAT en temps polinòmic. Dissenyeu un algorisme tal que donada una fòrmula booleana F retorni una assignació que la satisfaci, si existeix. Demostreu que si $\text{SAT} \in \text{P}$ aleshores *trobar* una assignació que la satisfà també és computable en temps polinòmic. Aquesta propietat s'anomena *self-reducibility*.
8. **Self reducibility CLIQUE.** Supposem que tenim un algorisme que decideix CLIQUE en temps polinòmic. Dissenyeu un algorisme tal que donat un graf G i donat un natural k , retorni una clíca de k vèrtexs, si existeix. Demostreu que si $\text{CLIQUE} \in \text{P}$ aleshores *trobar* una clíca també és computable en temps polinòmic. Aquesta propietat s'anomena *self-reducibility*.
9. **Stingy SAT.** Let us consider the following problem:

STINGY SAT Given a set of clauses (each a disjunction of literals) and an integer k , find a satisfying assignment in which at most k variables are true, if such an assignment exists.

Prove that STINGY SAT is NP-complete.

10. **At most 3.** Consider the CLIQUE problem restricted to graphs in which every vertex has degree at most 3. Call this problem CLIQUE-3.
 - (a) Prove that CLIQUE-3 is in NP.
 - (b) What is wrong with the following proof of NP-completeness for CLIQUE-3? We know that the CLIQUE problem in general graphs is NP-complete, so it is enough to present a reduction from CLIQUE-3 to CLIQUE. Given a graph G with vertices of degree ≤ 3 , and a parameter k , the reduction leaves the graph and parameter unchanged: clearly the output for the reduction is a possible input for the CLIQUE problem. Furthermore, the answer to both problems is identical. This proves the correctness of the reductions, and, therefore, the NP-completeness of CLIQUE-3.
 - (c) It is true the the VERTEX COVER problem remains NP-complete even when restricted to graphs in which every vertex has degree at most 3. Call this problem VC-3. What is wrong in the following proof of NP-completeness for CLIQUE-3?
 We present a reduction from VC-3 to CLIQUE-3. Given a graph $G = (V, E)$ with node degrees bounded by 3, and a parameter b , we create an instance of CLIQUE-3 by leaving the graph unchanged and switching the parameter to $|V| - b$. Now, a subset $C \subseteq V$ is a vertex cover in G if and only if the complementary set $V - C$ is a clique in G . Therefore G has a vertex cover of size $\leq b$ if and only if it has a clique of size $\geq |V| - b$. This proves the correctness of the reduction and, consequently, the NP-completeness of CLIQUE-3.
 - (d) Describe an $O(|V|)$ algorithm for CLIQUE-3.
11. **At most twice.** Show that 3SAT remains NP-complete even when restricted to formulas in which each literal appears at most twice.
12. **Degree at most 4** In the previous exercise we saw that 3SAT remains NP-complete even when restricted to formulas in which each literal appears at most twice.
 - (a) Show that if each literal appears at most once, then the problem is solvable in polynomial time.

- (b) Show that INDEPENDENT SET remains NP-complete even in the special case when all nodes in the graph have degree at most 4.
13. Consider a set of linear inequalities where all the coefficients are integers. Consider the problem INTLINEQ of deciding if a given set of linear inequalities has an integer solution. Related to this problem we can consider a variant of linear programming INTEGER PROGRAMMING in which the variables are restricted to be integers.
- Show that INTLINEQ is NPcomplete and that INTEGER PROGRAMMING is NP-hard.
14. Lucas' theorem says the following: If we have an integer a such that: $a^{n-1} \equiv 1 \pmod{n}$, and, for every prime factor q of $n-1$, it is not the case that $a^{(n-1)/q} \equiv 1 \pmod{n}$, then n is prime.
- Can this result be used to show that Primality belongs to NP?
15. Consider the following problems:
- MODULAR FACTORIAL: Given N bits natural numbers x, y compute $x! \pmod{y}$.
 - SMALLEST PRIME DIVISOR: Given a N bit natural number x , compute the smallest prime divisor of x .
 - FACTORING: Given a N bit natural number x , compute the factorization of x as product of primes.
- (a) Prove that y is prime if and only if, for each integer $x < y$, we have that $\text{mcd}(x!, y) = 1$.
- (b) Show that if MODULAR FACTORIAL can be solved in polynomial time, then SMALLEST PRIME DIVISOR and FACTORING could be solved in polynomial time.