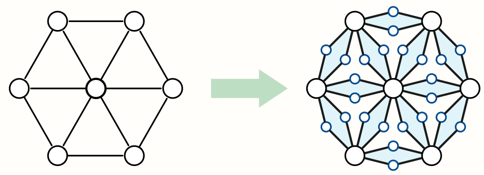
1. A subset S of vertices in an undirected graph G is called ***square-free*** if, for every four distinct vertices $u, v, w, x \in S$, at least one of the four edges uv, vw, wx, xu is *absent* from G. That is, the subgraph of G induced by S has no cycles of length 4. Prove that finding the size of the largest square-free subset of vertices in a given undirected graph is NP-hard.

Solution: We prove this problem NP-hard by reduction from the maximum independent set problem.

Given an arbitrary undirected graph $G = (V, E)$, we construct a new graph $G' = (V', E')$ as follows:

- $V' = V \cup \{x_{uv}, y_{uv} \mid uv \in E\}$
- $E' = \{ux_{uv}, vx_{uv}, uy_{uv}, vy_{uv} \mid uv \in E\}$

Intuitively, we construct G' by replacing each edge uv of G with a square using two new vertices x_{uv} and y_{uv}. Let $W = V' \setminus V$ denote the set of all these new vertices.



Now I claim that G contains an independent set of size at least k if and only if G' contains a square-free subset of size at least $k + 2E$. This claim immediately implies that the largest independent set in G has size $\ell - 2E$, where ℓ is the size of the largest square-free set in G'.

$\Longrightarrow$ Let I be an independent set of k vertices in G. Let $S = I \cup W$. Because I does not contain both endpoints of any edge in G, the set S does not contain all four vertices of any edge-square in G'. But the edge-squares are the only squares in G; every other cycle in G' is a subdivision of a cycle in G, and therefore has length at least 6. We conclude that S is a square-free subset of $k + 2E$ vertices in G'.

$\Longleftarrow$ On the other hand, let S be a square-free subset of ℓ vertices in G'.

First, suppose S does not contain some new vertex $x_{uv} \in W$. If S contains both u and v, let $S' = S + x_{uv} - u$; otherwise, let $S' = S' + x_{uv}$. The new set S' is still square-free, and at least as large as S. Thus, by induction, there is a square-free subset S'' that contains every vertex in W and has size at least ℓ.

Finally, let $I = S'' \setminus W$. Because S'' is square-free, it omits at least one vertex in every edge square; but S'' contains every vertex in W, so it must omit at least one endpoint of every edge in G. We conclude that I is an independent set of size at least $\ell - 2E$.

We can easily construct G' from G in polynomial time. ■

Rubric: 10 points: standard NP-hardness rubric

2. Fix an alphabet $\Sigma = \{0, 1\}$. Prove that the following problems are NP-hard.[1]

(a) Given a regular expression R over the alphabet Σ, is $L(R) \neq \Sigma^*$?

Solution: We describe a polynomial-time reduction from 3SAT. Let Φ be an arbitrary 3CNF boolean formula. Let n be the number of variables in Φ and let k be the number of clauses. We construct a regular expression R of length $O(nk)$ as follows.

Let $\Phi = C_1 \wedge C_2 \wedge \cdots \wedge C_k$, where each C_k is a clause. For each clause C_i and each variable x_j, we define a regular expression r_{ij} as follows:

- If C_i contains the variable x_j, then $r_{ij} = 0$.
- If C_i contains the negated variable $\bar{x}_j$, then $r_{ij} = 1$.
- Otherwise, $r_{ij} = (0 + 1)$.

For each index i, let $R_i = r_{i1} r_{i2} \cdots r_{in}$. The regular expression R_i encodes all assignments to the n variables that do *not* satisfy C_i. For example, if $n = 8$, we would transform the clause $(x_3 + \bar{x}_7 + \bar{x}_4)$ into the regular expression

$$(0+1)(0+1)\,0\,1\,(0+1)(0+1)\,1\,(0+1)$$

Let $R = R_1 + R_2 + \cdots + R_k$. The regular expression R encodes all assignments to the n variables that do *not* satisfy the formula Φ. In particular, Φ is satisfiable if and only if $L(R) \neq (0 + 1)^n$.

Finally, let $R_<$ be a regular expression for the set of all strings of length smaller than n, and let $R_>$ be a regular expression for the set of all strings of length larger than n. For example:

$$R_< = \underbrace{(0+1+\varepsilon)(0+1+\varepsilon)\cdots(0+1+\varepsilon)}_{n-1}$$

$$R_> = (0+1)^*\underbrace{(0+1)(0+1)\cdots(0+1)}_{n+1}$$

Then Φ is satisfiable if and only if $L(R_< + R + R_>) \neq (0+1)^*$. The final regular expression $R_< + R + R_>$ has length $O(n^2 + nk)$, and it can be constructed from Φ in $O(n^2 + nk)$ time by brute force. ■

Solution: We describe a polynomial-time reduction from 4COLOR. Let $G = (V, E)$ be an arbitrary graph; arbitrarily index the vertices as $V = \{1, 2, \ldots, n\}$. We construct a regular expression of length $O(n^3)$ whose language is not Σ^* if and only if G is 4-colorable, as follows.

Intuitively, we represent each possible 4-coloring of G as a string of length $2n$, where each pair of bits represents the color of one vertex. For each edge ij,

[1] In fact, both of these problems are NP-hard even when $|\Sigma| = 1$, but proving that is much more difficult.

where without loss of generality $i < j$, let R_{ij} be the regular expression

$$\begin{aligned}
&(0+1)^{2(i-1)}\,00\,(0+1)^{2(j-i-1)}\,00\,(0+1)^{2(n-j)}\\
&+(0+1)^{2(i-1)}\,01\,(0+1)^{2(j-i-1)}\,01\,(0+1)^{2(n-j)}\\
&+(0+1)^{2(i-1)}\,10\,(0+1)^{2(j-i-1)}\,10\,(0+1)^{2(n-j)}\\
&+(0+1)^{2(i-1)}\,11\,(0+1)^{2(j-i-1)}\,11\,(0+1)^{2(n-j)},
\end{aligned}$$

where A^k is shorthand for the concatenation of k copies of A. R_{ij} encodes the set of all 4-colorings of G in which vertices i and j have the same color. Each expression R_{ij} has length $O(n)$.

Now let R be the sum of the expressions R_{ij}, over all edges ij. Then $L(R)$ is the set of all strings encoding *bad* 4-colorings of G. In particular, G is 4-colorable if and only if $L(R) \neq (0+1)^{2n}$.

Finally, let $R_<$ be a regular expression for the set of all strings of length smaller than $2n$, and let $R_>$ be a regular expression for the set of all strings of length larger than $2n$. For example,

$$\begin{aligned}
R_< &= (0+1+\varepsilon)^{2n-1}\\
R_> &= (0+1)^*(0+1)^{2n+1},
\end{aligned}$$

where again A^k is shorthand for the concatenation of k copies of S.) Then G is 4-colorable if and only if $L(R_< + R + R_>) \neq (0+1)^*$. The final regular expression $R_< + R + R_>$ has length $O(n^3)$, and it can be constructed from Φ in $O(n^3)$ time by brute force. ■

Rubric: 5 points: standard poly-time reduction rubric (scaled). These are not the only correct solutions.

(b) Given an NFA M over the alphabet Σ, is $L(M) \neq \Sigma^*$?

Solution: We can reduce from the problem in part (a) using Thompson's algorithm, which converts any regular expressions into in equivalent NFA in polynomial time. ■

Rubric: 5 points: standard poly-time reduction rubric (scaled). Yes, this is enough for full credit.