STATEMENT

1. Graph G is *Eulerian* if there is a cycle going through each edge of graph G exactly once. Write the decision problem associated with it. The answer should follow this formality:
   Data: *The input data*
   Question: *A question to which the answer is* yes *or* no

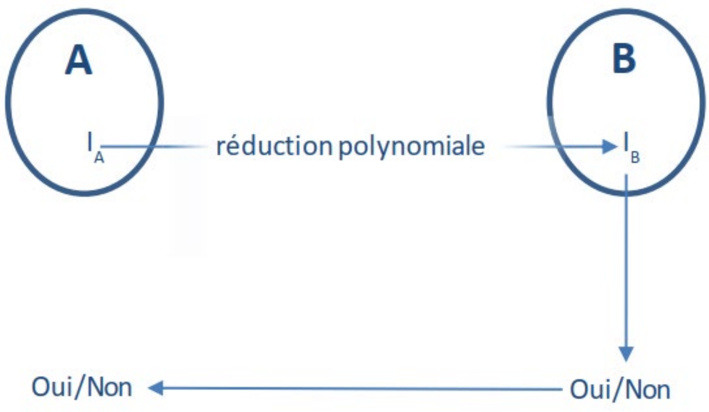
2. Graph G is *Hamiltonian* if there is a cycle going through each vertex of graph G exactly once. Write the decision problem associated with it.

3. The problem of the shortest path between two vertices in a graph is expressed as follows:
   Data: A graph $G=(U, E)$, two vertices $u, v \in U$
   Question: What is the shortest path in G from u to v?

   a. Write the corresponding decision problem, in which we are looking for a path from u to v with a lower length than a value k, given as a problem parameter.
   b. Reformulate the optimisation problem so that you use the decision problem above.

4. The problem of colouring a graph G consists in assigning a colour to each vertex of G, by prohibiting two neighbouring vertices from having the same colour, using a minimum number of colours.
   a. Write the optimisation problem.
   b. Write the associated decision problem.
   c. Reformulate the optimisation problem so that you use the decision problem above.

5. Let A, and B be decision problems. Consider that A is reducible in polynomial time to B:



Specify whether the following statements are true or false, and why:

a. B is at least as hard as A.
b. B is at most as hard as A.

6. Let A, and B be decision problems. Assuming that A is in $\mathcal{P}$. Specify whether the following statements are true or false, and why:
   a. If A is reducible in polynomial time to B, then B is in $\mathcal{P}$
   b. If B is reducible in polynomial time to A, then B is in $\mathcal{P}$

7. Consider the two Hamiltonian Cycle and Hamiltonian Path problems.

   Hamiltonian Cycle

   Data: an undirected graph G.

   Question: Does G contain a Hamiltonian cycle?

   Hamiltonian Path

   Data: an undirected graph G, two distinct vertices u and v of G.

   Question: Does G contain a Hamiltonian path (or Hamiltonian trail) between u and v?

   Assuming that the Hamiltonian Cycle problem is $\mathcal{NP}$-Complete. So it is necessary to prove that the Hamiltonian Path is also NP-Complete. To do this, it is necessary to show that the Hamiltonian Path problem is in $\mathcal{NP}$ (meaning that the correctness of a solution can be verified in polynomial time), and that it is $\mathcal{NP}$-Hard (at least as hard as any $\mathcal{NP}$ problem). Since it is among the hardest $\mathcal{NP}$ problems, it is $\mathcal{NP}$-Complete.

   Reminder: The difference between a chain and a cycle is that a cycle returns to its starting point.

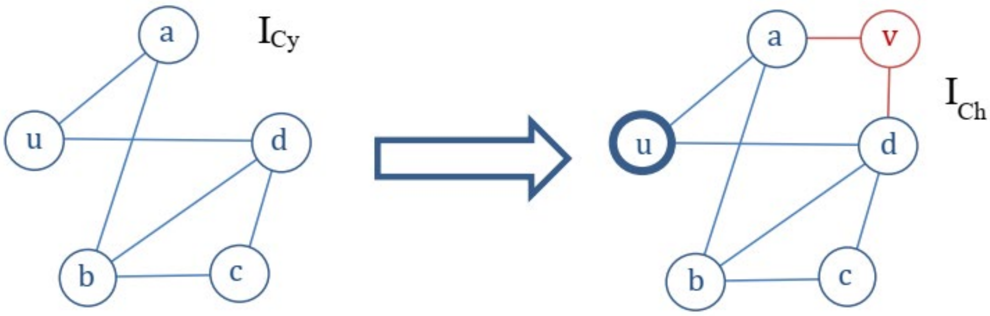
   a. Show that the Hamiltonian Path problem is in $\mathcal{NP}$: Consider:
      - an instance $\mathrm{I_{CH}}$ of the Hamiltonian Path problem, comprising graph $G=(V, E)$
      - a sequence of vertices $\mathrm{S_{Ch}} = \{u_1,\ldots, u_n\}$ of V.

      Propose an algorithm that uses $\mathrm{I_{Ch}}$ and $\mathrm{S_{Ch}}$ as input parameters, and checks whether $\mathrm{S_{Ch}}$ is a Hamiltonian path. Prove that the asymptotic complexity of this algorithm is a polynomial with the same length as $\mathrm{I_{Ch}}$.

   b. Show that the Hamiltonian Cycle problem is reducible in polynomial time to the Hamiltonian Path problem:

      Consider an algorithm that uses, as input parameter, an instance $\mathrm{I_{Cy}}$ of the Hamiltonian Cycle problem, comprising graph $G=(V, E)$, and that returns the instance $\mathrm{I_{Ch}}$ of the Hamiltonian Path problem comprising:
      - graph G', obtained by adding a vertex v to G, and by connecting said vertex to all the neighbours of a vertex u arbitrarily chosen in G
      - both vertices u and v

      In the following example, the vertex chosen to transform $\mathrm{I_{Cy}}$ into $\mathrm{I_{Ch}}$ is u:



i. Show that the asymptotic complexity of this algorithm is polynomial.
ii. Show that if there is a Hamiltonian path from u to v in G', then there is a Hamiltonian cycle in G.

   In the previous example, a possible solution to the instance I_{Ch} of the directed Hamiltonian path (or trail) between u and v is (u, a, b, c, d, v), and a possible solution to the instance I_{Cy} of the Hamiltonian cycle is (u, a, b, c, d, u), which is obtained by replacing v with u in the solution of I_{Ch}.

   Please note: This example is meant to illustrate the reasoning, given that the goal is not to demonstrate the existence of a cycle in this particular graph, but in every graph obtained by making the transformation shown above from any graph.

iii. Show that if there is no Hamiltonian path from u to v in G', then there is no Hamiltonian cycle in G.

Determine that the problem is $\mathcal{NP}$-Hard.

Version	Date	Designers	Proofreaders	Comment
1.0	13/03/2022	Benjamin COHEN BOULAKIA		Content retrieved from the first part of the Workshop Addition of simple questions More detailed correction of the Hamiltonian Path exercise

SOLUTION

1. Data: a graph G
   Question: Is there an Eulerian cycle in G?

2. Data: a graph G
   Question: Is there a Hamiltonian cycle in G?

3. Decision problem:
   Data: a graph G, two distinct vertices *u* and *v*, an integer k
   Question: Is there a path between *u* and *v* in *G* with a lower length than *k*?

   Optimisation problem:
   Data: a graph G, two distinct vertices *u* and *v*, an integer k
   Question: What is the smallest value of k for which the answer to the decision problem is *yes*?

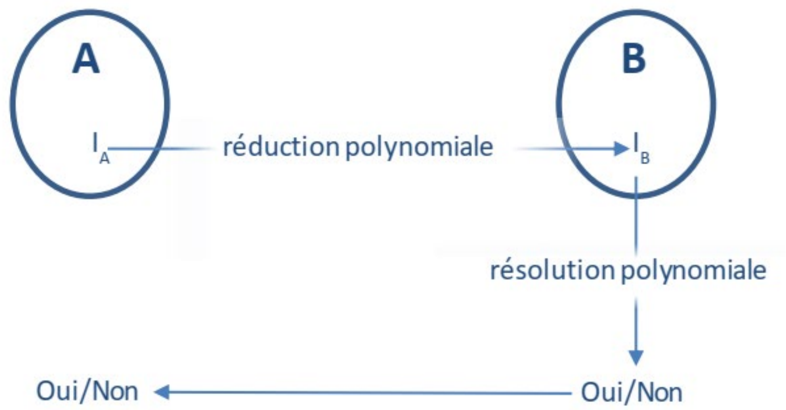
4.
   a. Optimisation problem:
   Data: A graph $G=(U, E)$
   Question: What is the smallest number of colours k with which we can colour G, so that we have $(u, v) \in E \Rightarrow k(u) \neq k(v)$?

   b. Decision problem:
   Data: A graph $G=(U, E)$, an integer k
   Question: Can we colour G with k colours, so that we have $(u, v) \in E \Rightarrow k(u) \neq k(v)$?

   c. Reformulation of the optimisation problem:
   Data: A graph $G=(U, E)$, an integer k
   Question: Can we colour G with k colours, so that we have $(u, v) \in E \Rightarrow k(u) \neq k(v)$?

5. 'A is reducible in polynomial time to B' means that there is a polynomial algorithm which transforms an instance I_A of A into an instance I_B of B, such that the answer to I_A is the same as the answer to I_B. More formally, we say that $I_A \in$ YES(A)[1] if and only if $I_B \in$ YES(B).

We show that if B can be solved in polynomial time, then A can be solved in polynomial time:



- We transform an instance I_A of A into an instance I_B of B in polynomial time (with the polynomial-time reduction algorithm)
- We solve instance I_B in polynomial time (since we assumed that B can be solved in polynomial time)

Symmetrically, we also demonstrate that if A cannot be solved in polynomial time, then B cannot be solved in polynomial time either (if an algorithm that would solve B existed, we could use it to solve A)

In conclusion, if we can solve B, we can solve A, and if we cannot solve A, we cannot solve B either. B is at least as *hard* as A (answer a. is right, hence answer b. is wrong).

6.

a. If A is reducible in polynomial time to B, then B is in $\mathcal{P}$: False.
If A $\in \mathcal{P}$ is reducible in polynomial time to B, this means that B is at least as hard as A, therefore potentially harder. Nothing special can be derived from this.

b. If B is reducible in polynomial time to A, then B is in $\mathcal{P}$: True.
If B is reducible in polynomial time to A, then A is at least as hard as B. Since A is in $\mathcal{P}$, B is also in $\mathcal{P}$.

7.

a. We have to build an algorithm that checks whether S_{CH} is a Hamiltonian path in I_{CH}. This algorithm will have three steps:

[1] The set of instances of *A* for which the answer is *yes*

1. Checking whether S_{CH} is actually a path of the I_{CH} graph, i.e. if there is an edge in I_{CH} between each pair of successive vertices of S_{CH}.
2. Browsing S_{CH} to check that each vertex of I_{CH} appears only once.
3. Checking whether S_{CH} has u and v as its ends.

Now let's determine the complexity of this algorithm. We consider that checking the existence of an edge in I_{CH} can be done in $O(1)$ for the considered data structure[1]. We also consider that reading an element in S_{CH} can be done in $O(1)$ for the considered data structure [2].

Note: most $\mathcal{NP}$-Completeness proofs skip this step of assumptions about access performance, given that constant-time access for the required operations is implied and allowed. Nevertheless, this must remain a focal point.

Let n be the number of vertices of the I_{CH} graph. The asymptotic complexity of each step is as follows:

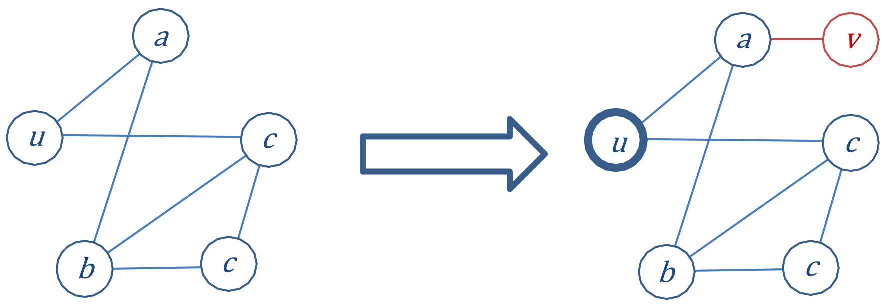
1. The first step is done in $O(n)$. In fact, there are, at most, n-1 edges in a valid solution. If there are more than n-1 edges, we know that the solution is invalid.
2. The second step is done in $O(n)$. Students only have to implement a dictionary associating to each vertex the fact that it has been visited during the S_{CH} trail.
3. The third step is done in $O(1)$.

Given that each step is of polynomial complexity, the verification algorithm is also polynomial.

b.

i. Instance I_{Cy} of the Hamiltonian Cycle problem comprises graph $G=(V, E)$. Instance I_{Ch} of the Hamiltonian Path problem comprises:

- graph $G' = (V', E')$ with
  - $V' = V + \{v\}$ with $v \notin V$
  - $E' = E \cup \{(v, l)$: l is a neighbour of u in $G\}$ with u, a random vertex of G
- both vertices u and v



We assume that the addition of vertex v is done in $O(1)$ for the considered data structure[3]. The addition of edges to the neighbours of u is done in, at most, n-1 operations

[1] This is true for the adjacency matrix, and for the adjacency list when implemented with a dictionary.
[2] This is true for a simple contiguous structure (array in C, tuple in Python, ...) or an optimised structure (Set in Python).
[3] This is true for the adjacency matrix.

(if u is connected to all other vertices of G). Therefore, the complexity of this transformation is $O(n)$.

ii. Assuming there is a Hamiltonian path in G' going from u to v. This path is of form $(u, l_1, \ldots, l_{n-1}, v)$. In fact, the vertices $\{l_i\}$ are the neighbours of u in G, and they are the only vertices connected to v in G', by design of G'.
We then derive that there is necessarily a Hamiltonian cycle in G. This cycle consists of the previous path, in which we have replaced the final vertex v by vertex u in order to form a cycle (which is feasible since, in the path, the predecessor of v is also a neighbour of u). This cycle passes through each vertex of G exactly once (except for v because this is a cycle), since the path from which it originates passes through each vertex of G' exactly once.

iii. Assuming there is no Hamiltonian path from u to v in G'. There cannot exist a Hamiltonian cycle in G. If such a cycle existed, we could obtain a path in G' by doing the inverse transformation of the path shown above.
If an algorithm capable of solving the Hamiltonian Path problem in polynomial time existed, we could use it to solve the Hamiltonian Cycle problem in polynomial time. So, the Hamiltonian Cycle $\leqslant$ the Hamiltonian Path, and therefore the Hamiltonian Path is $\mathcal{NP}$-Hard.

As the Hamiltonian Path is in $\mathcal{NP}$, the Hamiltonian Path is $\mathcal{NP}$-Complete.