Methodology to prove NP-completeness of my problem

In order to prove that my problem is NP-complete, I did the following steps:

1. Prove your problem L belongs to NP (that is that given a solution you can verify it in polynomial time)
2. Select a known NP-complete problem L'
3. Describe an algorithm f that transforms L' into L
4. Prove that your algorithm is correct (formally: x ∈ L' if and only if f(x) ∈ L )
5. Prove that algo f runs in polynomial time

Step1: proof my problem is NP

First note that the evolution provenance graph matching problem is in the class NP. To verify a given solution G12 result of merging graph G1 and G2, we check first whether there are no cycles in G12 which can be done in O(n2) time.

Note that we need also to verify that G12 is the best tradeoff that maximizes the multiobjective problem (high similarity values as well high number of matched pairs).

This is can be verified using a skyline computation approach in O(n2) time.

Melanie said that the application of a pareto frontier solution solves my problem in linear time.

Step2: Select a known NP-complete problem L' maximum common subgraph problem

In the MCS problem, we are given two undirected graphs G1 = (V1,E1), G2 = (V2,E2), and a size k . We want to find two sets of vertices V1′ ⊆ V1 and V2′ ⊆ V2 such that deleting V1′ in G1 and V2′ in G2 leaves at least k vertices in each graph, and makes the two graphs identical.

Finding the MCS in random graphs is an NP-complete problem [1,2,3].

[1] H. Bunke, P. Foggia, C. Guidobaldi, C. Sansone, and M. Vento, “A comparison of algorithms for maximum common subgraph on randomly connected graphs,”

[2] D. Conte, P. Foggia, and M. Vento, “Challenging complexity of maximum common subgraph detection algorithms: A performance analysis of three algorithms on a wide database of graphs,”

[3] W. Henry Suters, F. Abu-Khzam, Y. Zhang, C. Symons, N. Samatova, and M. Langston, “A new approach and faster exact methods for the maximum common subgraph problem,”

Step3: Describe an algorithm f that transforms L' into L

To prove NP-hardness, we reduce from the MCS problem.

In the MCS problem, we are given two undirected graph G=(V,E) and G’=(V’,E’), we research the maximum common sub graph.

Given an instance of the MCS problem, we reduce to an instance of our evolution provenance graphs matching as follows: 🡺 reduction is done as follows

We build a [DAG](https://en.wikipedia.org/wiki/Directed_acyclic_graph) from each graph input to the MCS problem. Note that a directed graph is a DAG if and only if it can be [topologically sorted](https://en.wikipedia.org/wiki/Topological_sorting).

* Do a topological sort on the graph using DFS for instance
* Based on this new order, from first to last set the direction of each edge such that the source node is the one with the lower topological sort.
* Time complexity of the algorithm is linear in the number of nodes and edges.

Thereby, we will have 2 DAG input graphs Gdag and G’dag to our evolution provenance graphs matching problem.

**Step 4: Proof of correctness:** x ∈ L' if and only if f(x) ∈ L

Suppose that Gm= (Vmu,Emu) is a solution to our matching problem applied on Gdag and G’dag when setting =1. Then we have a solution for MCS problem. The MCS solution is obtained as follows: We take the set of merged nodes and edges done when solving the evolution provenance graphs matching problem🡪 the maximum number of connected elements constitute the MCS solution.

Conversely, suppose that there is (V1′ , V2′) (with V1′ ⊆G and V2′ ⊆G′) is the solution of MCS of order k when comparing G, G’. Then there is a solution of our evolution provenance graphs matching applied on Gdag and G’dag when setting =1. This solution relies mainly on the DAGS version of transformation applied on (V1′ , V2′) . This ensures that the solution of our problem provides a high number of matching pairs

Recall that V1′ and V2′ are isomorphic. Then Sim(m1,m2)=1. This ensures that the solution of our evolution provenance graph problem maximizes the similarity of matched pairs.

Now we need to prove that our solution does not introduce a cycles. Let’s assume that our solution introduces cycles. I don’t have solution for that

V1′ and V2′ are isomorphic. This ensures that each node in V1′ in level li has its matching with a node in V2′ in level li. Thereby there is no DAGs introduced by the solution of our problem.