==
CSC 373 Tutorial Solutions for Week 10 Winter 2019
==

Showing that both problems belong to NP -- "short" version of the argument.

* HP in NP: Given (G,c), it takes polytime to verify that c is a permutation of the vertices of G [v_1,...,v_n] and that G contains every edge between successive vertices in c: (v_1,v_2),...,(v_{n-1},v_n).

* st-HP in NP: Given (G,s,t,c), it takes polytime to verify that c is a permutation of the vertices of G starting with s and ending with t [s,v_2,...,v_{n-1},t] and that G contains every edge between successive vertices in c: (s,v_2),(v_2,v_3),...,(v_{n-2},v_{n-1}),(v_{n-1},t).

Correct reductions for showing NP-hardness -- incorrect ideas discussed further below.

* HC -p> HP (also HC -p> st-HP):
  On input G = (V,E), output G' = (V',E') where
    - V' = V u {s,t,r} (add 3 new vertices r,s,t);
    - E' = E u {(s,v_0)} u {(r,t)} u {(u,r) : (u,v_0) in E}, for some v_0 in V (the choice does not matter), i.e., s is connected to v_0, every vertex that has an edge to v_0 is connected to r, and r is connected to t.

  Transformation can be computed in polytime (linear time, in fact). If G contains some Ham. cycle C, then in G', start at s, go to v_0, follow C, and just before getting back to v_0, more to r instead and then t: this is a Ham. path in G'. If G' contains some Ham. path P, it must begin at s and end at t -- because s only has one outgoing edge (s,v_0) and t only has one incoming edge (r,t) (in undirected case, both s and t have a single edge attached). By definition of Ham. path, every vertex of G appears exactly once on P. Also, by construction of G', G contains an edge (u,v_0) for every edge (u,r) in G': so G contains a Ham. cycle (can get back to v_0 after visiting every vertex in G).

  Note: new vertex r and edge (r,t) unnecessary for directed problem -- can connect u directly to t -- but required for undirected problem.

* st-HP -p> HP:
  On input (G = (V,e), s, t), output G' = (V',E') where
    - V' = V u {s',t'}; . E' = E u {(s',s),(t,t')}.

  Clearly, G' can be computed in polytime from (G,s,t). If G contains a Ham. path from s to t, then extend the path to s' and t' in G' to show G' contains a Ham. path. If G' contains a Ham. path, it must start at s' and end at t' (because of single edges connecting s' to s and t to t'), so part of the path is a Ham. path in G from s to t.

* HP -p> st-HP:
  On input G = (V,E), output (G' = (V',E'), s, t) where
    - V' = V u {s,t}; . E' = E u {(s,v) : v in V} u {(v,t) : v in V}.

  Clearly, (G',s,t) can be computed in polytime from G. If G contains a Ham. path starting at v_1 and ending at v_n, then (s,v_1),...,(v_n,t) is a Ham. path from s to t in G'. If G' contains a Ham. path from s to t, say [s,v_1,...,v_n,t], then [v_1,...,v_n] is a Ham. path in G.

Incorrect reduction ideas, with some hints about how to understand the error.

* HP -p> HC (or any other attempt to reduce in the wrong direction):
  Q: What is the goal of the reduction, i.e., what conclusion do we want to derive?
  A: HP is NP-hard.
  Q: Definition?
  A: For all D in NP, D -p> HP. So conclusion involves showing reductions _to_ HP, and can only be obtained by doing a reduction _to_ HP.

* HC -p> HP:
  "Trivial because if G contains a Ham. cycle, it's already also a Ham. path."
  Q: What are the properties that a reduction function must satisfy?
  A: Computable in polytime; if G in HC, f(G) in HP; if G not in HC, f(G) not in HP.
  Q: Are all three properties satisfied?
  A: First two are, but not the last: G could contain a Ham. path without also containing a Ham. cycle.

* HC -p> HP:
  "Take the Ham. cycle in G, remove one edge from it to turn it into a Ham. path."
  Q: What is the input for the reduction function?
  A: G (input to HC).
  Q: What does the "reduction" above use as input?
  A: G along with Ham. cycle in G.
  Reduction does _not_ have access to "certificate". It gets only problem input (G alone, in this case) and must run in polytime without any additional information. Also, reduction must work for _all_ instances, not just yes-instances, so it must be described for a general input without making any assumption about the answer.

* HC -p> HP:
  Q: What goes wrong if we omit r and t from the reduction above?
  A: Output could have a Ham. path without input having a Ham. cycle. (Come up with a specific example to show this.)
  Q: What goes wrong if we omit r from the reduction above and connect the neighbours of v_0 directly to t, when the input is undirected?
  A: Output could have a Ham. path without input having a Ham. cycle. (Show this by coming up with a specific example.)

* HP -p> st-HP and st-HP -p> HP:
  Q: Show simpler "reductions" don't work by coming up with counterexamples (similar to previous cases above).