Prolog assignment - Graph Isomorphism

May 5, 2015

Assignment

In graph theory, an isomorphism of graphs G and H is a bijection between the vertex sets $V(X)$ of G and H

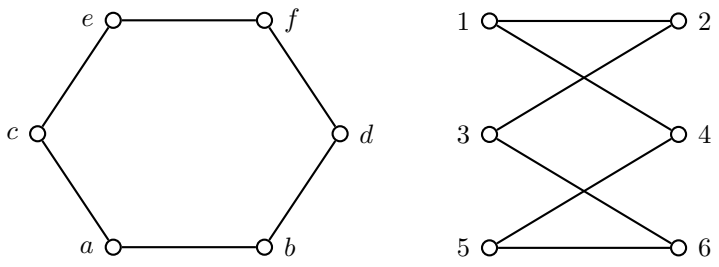
$$f: V(G) \to V(H)$$

such that any two vertices u and v of G are adjacent in G if and only if (u) and (v) are adjacent in H. An example of two isomorphic graphs G and H is given in Figure 1(a) and 1(b), respectively.

Besides its practical importance, the graph isomorphism problem is a curiosity in computational complexity theory as it is one of a very small number of problems belonging to NP category, neither known to be solvable in polynomial time nor NP-complete. That is, if you manage to solve this assignment in polynomial time you will be famous (with instant A from the course, of course).

Task

Given any two finite, undirected, non-weighted graphs G and H, decide whether they are isomorphic, i.e. whether there exists an isomorphic mapping of $V(G) \to V(H)$.



(a) G (b) H

Figure 1: Two isomorphic graphs G and H with $f_1 : \{1 \to a, 2 \to b, 3 \to c, 4 \to d, 5 \to e, 6 \to f\}$ as one of corresponding mappings f (there may be a multitude of such).

Representation

Both input graphs will be presented as sets of ground Prolog facts determining the graph's edges, i.e.

```
e(1, 2).          f(1, 2).
e(2, 3).          f(1, 4).
...               ...
e(5, 6).          f(4, 9).
```

- These are to be understood as undirected edges (`e(1, 2)` and `e(2, 1)` are equivalent).
- You can expect the sizes of input graphs not to exceed $n = 30$ number of vertices.

Where to start

Some tips you might want to follow:

1. Implement a brute force search solution
2. Think about some rejection checks using *invariants*. Two graphs *can not* be isomorphic if they differ in the number of vertices and edges, vertex degree distribution, ...
3. Use Google (allowed here)
4. Use your imagination, we expect you to be creative

Solution

We expect your solution to be correct, complete, and able to process graphs of given size in reasonable computational time (still, exponential complexity will suffice).

You will be given 8 pairs of graphs for evaluation. Some of them are isomorphic, some are not. You are expected to solve all of them to gain 100% points. Teachers reserve the right to reduce the grade if a serious problem in the code is found.

Implement the algorithm in Prolog and save it in `iso.pl`. The main predicate should be called `iso` and succeed iff the two graphs are isomorphic.

Submit your work before 31/05/2015 23:59. The deadline is strict.

References

[1] Testing graph isomorphism Fischer, Eldar, and Arie Matsliah. *"Testing graph isomorphism."* SIAM Journal on Computing 38.1 (2008): 207-225.

[2] Corneil, Derek Gordon, and Calvin C. Gotlieb. "An efficient algorithm for graph isomorphism." Journal of the ACM (JACM) 17.1 (1970): 51-64.

[3] Foggia, Pasquale, Carlo Sansone, and Mario Vento. *"A performance comparison of five algorithms for graph isomorphism."* Proceedings of the 3rd IAPR TC-15 Workshop on Graph-based Representations in Pattern Recognition. 2001.

[4] A Graph Isomorphism Algorithm http://www.dharwadker.org/tevet/isomorphism/