GRAPH ISOMORPHISM AND CIRCUIT ISOMORPHISM

Thinh D. Nguyen

Abstract

In this note, we show that graph isomorphism and some of its variant are both reducible to circuit isomorphism problem, in polynomial time.

Keywords: polynomial-time, reducible, graph

1. Notation

We denote the well-known graph isomorphism problem by **GraphIso** which is defined as follows:

$$\textbf{GraphIso} = \{ <G_1, G_2>, \text{graph } G_1 \text{ is isomorphic to graph } G_2 \}$$

A variant of the graph isomorphism is also considered. Hypergraph are a generalization notion of graph. In hypergraph, each hyper-edge is a subset of the vertex set. A weighted hypergraph is a hypergraph, in which, we assign a natural number to each hyper-edge. In the problem of weighted hypergraph isomorphism, you are given two weighted hypergraph G_1, G_2 and are ask whether there exists a vertex permutation π turning G_1 into G_2. We denote this problem by **WeightedHyperGraphIso**:

$$\textbf{WeightedHyperGraphIso} = \{ <G_1, G_2>, \text{weighted hyper-graph } G_1 \text{ is isomorphic to weighted hyper-graph } G_2 \}$$

There are numerous textbooks in graph theory. One excellent book is [2].

In this note, we consider Boolean circuits with AND, OR, NOT gates. Circuits can have one or more output bits. The fan-in of the gates can be

Email address: kosmofarmer@gmail.com (Thinh D. Nguyen)

bounded (2 for AND and OR gates, 1 for NOT gates) or un-bounded, without affecting the results. Howerver, the fan-out of gates, as well as input, should be unbounded. For otherwise, we would fall into the formulae model, which is much less in computing and expressive capability. For details, the reader is addressed to [1].

Two n-bit input, m-bit output circuits C_1, C_2 are isomorphic if there exists a permutation $\pi \in S_n$ such that for every $x \in \{0,1\}^n$, $C_1(x) = C_2(\pi(x))$

By **CircuitIso**, we denote the circuit isomorphism problem:

CircuitIso = { $<C_1, C_2>$, two n-bit input, m-bit output circuit C_1 and C_2 are isomorphic }

2. Main results

Theorem 1: **GraphIso** $\leq_p$ **CircuitIso**
Theorem 2: **WeightedHyperGraphIso** $\leq_p$ **CircuitIso**

In the following section, we will give two similar reductions to prove both of the theorems.

3. Details of the reduction

3.1. Description of the two similar reductions

In **GraphIso**, given two n-vertex graphs G_1, G_2, we construct two corresponding circuits C_1, C_2 as follows:
The number of input bits to G_1, G_2 is equal to n. We will describe the circuit C_1, the construction of C_2 is similar. To code the structure of G_1 into C_1, we consider each edge (u, v) of G_1. For each such edge, we create a sub-circuit (gadget in poly-time reduction literature) which guarantees that among the n-input bit only two bits corresponding to u and v are 1. Then, to complete our construction, we take the **OR** of all the gadgets as output of C_1.

Similarly, in **WeightedHyperGraphIso**, the above construction only needs a small change to work. Note that the idea of the above reduction is that C_1 can be seen as a edge-tester of graph G_1. The needed change is to make C_1 a hyper-edge tester when G_1 is a hypergraph. Each sub-circuit gadget now guarantees that the set of the corresponding vertices of input bits assigned 1 is a hyper-edge. The overall **OR** is now replaced by a selecting

circuit. In particular, we imagine that we have the weight of each hyper-edge beside the output of its corresponding gadget output, then a selecting circuit will output the weight value of a gadget when that one outputs 1.

A minor technicality should be of concern is that the number of output bits is then the longest binary representation of the hyper-edge weights. The overall output in the case of non-hyper-edge is some natural number not assigned to any hyper-edge of both hypergraphs. This natural number is regarded as signal of non-hyper-edge.

3.2. Correctness of the reductions

Proof of theorem 1:
As mentioned above, C_1 and C_2 should be regarded as edge-tester of G_1 and G_2, respectively. If G_1 and G_2 are isomorphic, then there exist a vertex permutation π turning G_1 into G_2. The same permutation applied to the input bits of C_1 will make this circuit equivalent to C_2. The inverse is also not difficult.

Proof of theorem 2:
Similarly, the hyper-edge weights output by C_1 and C_2 respectively corresponds to the structure of G_1 and G_2. The isomorphism mapping π turning G_1 into G_2 when applied to input bits of C_1 will make this circuit equivalent to C_2. Indded, when received a (supposed) hyper-edge from input wires, both C_1 and C_2 will output its weights. If π is an isomorphism mapping, then clearly, the two results are identical.

The inverse direction is simple.

4. Conclusion

There have been intensive ongoing research on problems in graph theory, especially from a computational point of view. This work demonstrates that a seemingly irrelevant problem from the field of Boolean circuit could be used to shed light on the (non-)solvability of computational graph-theoretic problems.

References

[1] Boolean circuit, Wikipedia, `https://en.wikipedia.org/wiki/Boolean_circuit`, 2016

[2] Norman L. Biggs, Discrete Mathematics, 2nd edition, Oxord University Press, 2003