

## The Babai-Luks-Zemlyachenko algorithm for general GI

Speaker: Ramprasad Satharishi

Scribe: Srikanth Srinivasan

## 1 Overview

We describe a mildly exponential time for the general Graph Isomorphism time problem. The algorithm is due to Babai and Luks, using a result of Zemlyachenko. It runs in time  $n^{O(n^{2/3})}$ .

## 2 The algorithm

We look at the more general scenario of coloured graphs. Formally, a coloured graph is a tuple  $G = (V, E, f)$  where  $(V, E)$  is a graph, and  $f$  is a colouring function mapping the vertices to a set  $C$  of colours. (We may assume, without loss of generality, that  $C = \{1, \dots, |V|\}$ .) Given a pair of coloured graphs  $G_1$  and  $G_2$ , we wish to decide if there is a colour preserving isomorphism between the two, i.e an isomorphism of graphs that, for all  $c \in C$ , maps vertices of colour  $c$  to vertices of colour  $c$ . We need the following definition:

**Definition 1.** *A coloured graph is said to have Colour valence at most  $d$  iff for every  $x \in V$ , and every  $c \in C$ , either  $|N(x) \cap f^{-1}(c)| \leq d$  or  $|\overline{N(x)} \cap f^{-1}(c)| \leq d$ . Here  $N(x)$  denotes the neighbourhood of  $x$ .*

We proved the following theorem last time:

**Theorem 1.** *For any  $d \in \mathbb{N}$ , there is an algorithm that, given any two coloured graphs  $G_1$  and  $G_2$  of colour valence at most  $d$ , decides if there is a colour-preserving isomorphism between the two graphs. The running time of the algorithm is  $n^{O(d^2)}$ .*

Our strategy will be this: to reduce the colour valence sufficiently so that the above algorithm can be applied. We will do this by using two steps repeatedly. The first of these is *Individualization*, the process of assigning to a vertex  $v$  a new colour. The second, *Refinement*, is described below.

## 2.1 Refinement

Given a coloured graph  $G$ , the process of refinement is a technique that helps us ‘distinguish’ between different vertices in the same colour class. In particular, given two coloured graphs  $G_1$  and  $G_2$ , if they refine in different ways, they are not isomorphic.

Let us describe the process. We start with a coloured graph  $G = (V, E, f)$  and iterate the following procedure:

1. To each vertex  $v$  in the graph, assign a tuple  $(c, S)$ , where  $c = f(v)$  and  $S$  is the multiset of colours assigned to its neighbours in  $G$ .
2. Rename the colours obtained in Step 1 by the numbers  $1, \dots, |V|$  (there are at most so many). This is to stop the sizes of the representations of the colours from blowing up.

We continue the above procedure until repeating it produces no refinement, i.e it does not split the colour classes any further. (It is clear that this process cannot bring together vertices coloured differently.) At this point, we say that the colouring has *stabilized*. Clearly, any colouring must stabilize after at most  $n$  iterations.

## 2.2 The details

We can now describe the algorithm in some more detail. We would like to apply the process of refinement until the colour valence of the input graphs becomes small. However, refinement may stabilize without making much headway (As an extreme case, consider the case when the graphs are regular, and all their vertices are coloured the same). To get around this, we pick and individualize a vertex in one of the graphs and continue the process of refinement. We show below that, in any coloured graph, one can efficiently find a small set of vertices such that individualizing them and refining results in a graph with small colour valence. We do this on one of the input graphs. After this, we simply run through all possibilities of the ‘corresponding vertices’ in the other graph and imitate the process there to get another graph of small colour valence (if we don’t get such a graph, we know our guess is wrong). Now, we apply Luks’s algorithm to solve the Graph Isomorphism on these coloured graphs.

The following lemma tells us that the above strategy will work.

**Lemma 2.** *Assume the graph  $G = (V, E, f)$  has colour valence at most  $d$ . Then, for some  $k \leq 2n/d$ , there exist vertices  $x_1, x_2, \dots, x_k$ , such that*

the process of individualization followed by refinement applied to all of them results in a graph of colour valence bounded by  $d/2$ . Moreover,  $x_1, \dots, x_k$  can be computed in polynomial time.

*Proof.* The proof is constructive: we simply give an algorithm to compute  $x_1, \dots, x_k$ .

Assume the set  $S = \{x_1, \dots, x_i\}$  has been picked so far. If the colour valence of the graph is at most  $d/2$ , we are done. Otherwise, there is some vertex  $v$  and some colour  $c$  such that both  $|N(x) \cap f^{-1}(c)|$  and  $|\overline{N(x)} \cap f^{-1}(c)|$  are greater than  $d$  in size. Set  $x_{i+1} = v$ , individualize it, and refine. Repeat.

Clearly, the algorithm above is polynomial time. We need to show only that after at most  $2n/d$  iterations, we have a graph with colour valence at most  $d/2$ .

Since  $G$  has colour valence at most  $d$ , for each  $x_i$  and the corresponding colour  $c$  picked above, we know that one of the two sets  $N(x_i) \cap f^{-1}(c)$  and  $\overline{N(x_i)} \cap f^{-1}(c)$  (note that  $f$  here denotes the colouring *at the time  $x_i$  was picked*) is of cardinality at most  $d$ . Let  $T_i$  be that set. We will prove that, for distinct  $i$  and  $j$ ,  $T_i \cap T_j = \emptyset$ . Assuming this, the proof is easy. For each  $i$ , by the definition of  $x_i$ ,  $T_i$  must have size at least  $d/2$ . Hence, the number of  $T_i$ s is at most  $n/(d/2) = 2n/d$ , which is what we wanted to prove.

We now prove that, for all  $i \neq j$ ,  $T_i \cap T_j = \emptyset$ . Assume, without loss of generality, that  $i < j$ . Note that when  $x_i$  is individualized, the vertices in  $T_i$  become a colour class (or perhaps split further). Hence, if  $T_j$  and  $T_i$  share a vertex  $v$ , then it must be the case that the ‘bad colour class’ with colour  $c$  picked for  $x_j$  must be a subset of  $T_i$ . But this implies that  $T_i$  contains both  $N(x) \cap f^{-1}(c)$  and  $\overline{N(x)} \cap f^{-1}(c)$ , which are both of size greater than  $d/2$ . Hence,  $T_i$  is of size greater than  $d$ . This is a contradiction since, by the choice of  $T_i$ , it is of size at most  $d$ . This concludes the proof.  $\square$

Here is the algorithm, in all its glory:

- Input: Coloured graphs  $G_1$  and  $G_2$ .

1. Apply above refinement process to  $G_1$  until its colour valence is bounded by  $d$ . ( $d$  will be fixed later) It is easy to check that the total number of individualizations is bounded by  $4n/d$ . Let the exact number be  $i$ .
2. For each choice of  $i$  vertices from  $G_2$ , individualize them and refine as was done for  $G_1$ . If the resulting graph does not have colour valence bounded by  $d$ , proceed to the next choice. Otherwise, check, using Luks’s algorithm, if the graphs are isomorphic.

The correctness of the above algorithm is trivial. The running time is  $n^{O(d^2)} \cdot n^{O(n/d)}$ . To optimize the exponent, we choose  $d = n^{1/3}$ . This gives us a running time of  $n^{O(n^{2/3})}$ .