Algebra and Computation	Course Instructor: V. Arvind

Lecture 6,7: Special Cases of GRAPH-ISO

Lecturer: V. Arvind *Scribe: Ramprasad Saptharishi*

1 Overview

It is unlikely that we have an efficient algorithm for GRAPH-ISO, but certain special cases of it can be solved in polynomial time. In this lecture, we shall inspect two such cases, coloured graphs with bounded colour classes and trivalent graph.
[1]

2 Bounded Colour Multiplicity GRAPH-ISO: $BCGI_b$

Instead of general graph isomorphism instances, $BCGI_b$ adds an extra structure to the graphs by associating with each vertex a colour. Further, assume that each colour class[2] has its size bounded by a constant b. Given two graphs X_1 and X_2 with such a promise, can check if they are isomorphic efficiently?

Without loss of generality, we can assume that X_1 and X_2 are connected (otherwise consider their complements). Consider the graph $(V, E) = X = X_1 \cup X_2$ and from our earlier lectures, checking isomorphism can be reduced to checking the automorphism group of X. The additional colour structure imposes the constraint that $Aut(X) \leq \bigotimes \mathrm{Sym}(C_i)$ where C_i is the colour class for colour i. The idea is that we use the SET-STAB reduction in this scenario and show that we can compute the necessary stabilizer efficiently.

Let $E_i = E \cap \binom{C_i}{2}$ for each colour i (the set of intercolour edges) and $E_{ij} = E \cap (C_i \times C_j)$ for colour pairs $i \neq j$ (cross edges). Clearly, any $\pi \in \bigotimes \mathrm{Sym}(C_i)$ is a colour preserving automorphism *if and only if*, $E_i^\pi = E_i$ and $E_{ij}^\pi = E_{ij}$. Thus we now have got it to a SET-STAB form.

Let $G = \bigotimes \mathrm{Sym}\, C_i = \langle A \rangle$, by choosing transpositions as generators. Now define $\Omega = (\bigcup C_i) \cup \left(\bigcup 2^{\binom{C_i}{2}}\right) \cup \left(\bigcup 2^{C_i \times C_j}\right)$, which is just identifying

[1] We also discussed a partial solution to the solvability exercise he gave in the last class, but I'm not putting it here

[2] set of vertices of a given colour

the subsets E_i and E_{ij} as points. The bound on the colour classes tell us that $|\Omega| \leq n + r2^{\binom{b}{2}} + \binom{r}{2}2^{b^2}$, if r is the number of colour classes, and since b is a constant $|\Omega| = poly(n)$.

G can be naturally extended to act on Ω (extension in the most obvious sense, if i is sent to something and j is sent to something, E_ij should go to the right thing). And now, $Aut(X)$ is just finding the subgroup that pointwise stabilizes the E_i clusters and the E_{ij} cllusters and this can be done using our *strong generating set* tower discussed in lecture 3.

The running time is $poly(|\Omega|)$ and hence $poly(|X|)$.

3 Bounded Degree GRAPH-ISO

Another restriction that we can impose on graphs is the degree of each vertex, suppose we are given the promise that the degree of each vertex is bounded by a constant d, can we solve GRAPH-ISO?

The case when $d = 1$ is trivial, just a bunch of independant edges and so is the case when $d = 2$. The first interesting case is when $d = 3$, which is also called *trivalent graph isomorphism.*

We are given two graphs X_1 and X_2 with the promise that their degrees are bounded by 3, and we need to check if they are isomorphic. Firstly note that we can assume that both of them are connected, since if not we can just look at the connected components and work with all possible pairs[3]. Checking if they are isomorphic is equivalent to computing their automorphism group.

Further, suppose had a distinguished edge, and we are only intereseted in automorphisms that fix that edge, is that good enough? Yes it is. Fix some edge $e_{uv} \in X_1$, for each $e_{pq} \in X_2$, add a new edge that 'connects' e_{uv} and e_{pq} (add the midpoints as another vertex and join the two midpoints). Thus if an isomorphism swapped e_{uv} and e_{pq}, that isomorphism will fix e. And since we are doing this over all edges e_{pq}, the two problem are clearly equivalent. Hence we shall restrict ourselves to finding $Aut_e(X)$ for some graph X where e is a distinguished edge we want to fix.

The algorithm works by building the automorphism groups by approximations, $Aut_e(X_r)$ where each X_r is a subgraph of X. For each r, define X_r to be the consisting of all vertices and edges that appear in paths of length $\leq r$ passing through e. This layers X with respect to distance from e.

picture needed

[3]note that the complement idea won't work since graph will no longer be trivalent

And clearly, since e is a distinguished edge, any $\pi \in Aut_e(X)$ must preserve layers. And infact we have the following crucial theorem by Tutte.

Theorem 1 (Tutte)**.** *If X is a connected trivalent graph and e is any edge in X, then $Aut_e(X)$ is a 2-group (a group of order 2^m).*

Proof. The basic idea is that the automorphism groups are successive approximations and each expansion is through a 2-group.

The proof will be an induction on i, assume that $Aut_e(X_i)$ is a 2-group. Since any automorphism that preserves e has to respect the layers, we have a natural homomorphism $\phi : Aut_e(X_{i+1}) \longrightarrow Aut_e(X_i)$, which is just the projection function ($Aut_e(X_{i+1})$ preserves layers till X_{i+1}).

Hence $|Aut_e(X_{i+1})| = |\phi(Aut_e(X_{i+1}))| \cdot |\ker \phi|$ and since $\phi(Aut_e(X_{i+1}))$ is a subgroup of $Aut_e(X_i)$, it is a 2-group. Hence all that's left to do is to check that $\ker \phi$ is a 2-group as well.

We are interested in counting $\pi \in Aut_e(X_{i+1})$ that fixes X_i pointwise. If $V = V(X_{i+1}) \setminus V(X_i)$, then any non-trivial π have to do something on V alone. But note that the graph X is trivalent, and hence any $u \in V(X_i)$ can be connected to atmost 2 vertices in V (since degree of u is bounded by 3) and hence any automorphism of X_{i+1} that fixes X_i can atmost swap the two neighbours of u. Thus any $\pi \in \ker \phi$ satisfies the constraint that $\pi^2 = id$ and hence $\ker \phi$ is also a 2-group. □

3.1 A Road Map

We want to compute $Aut_e(X)$, and we shall do it using the tower induced by the different layers X_r. The general philosophy is the following:

If $\phi : G \longrightarrow H$ is a group homomorphism and if we had a generating set for $\ker \phi = \{k_1, k_2, \cdots, k_n\}$ and $\phi(H) = \{\phi(g_1), \phi(g_2), \cdots, \phi(g_m)\}$, then we can find a generating set for G efficiently.

We are going to use the homomorphisms $\phi : Aut_e(X_{r+1}) \longrightarrow Aut_e(X_r)$ to ascend the tower and finally get to a generating set for $Aut_e(X)$. We need to find

1. A generating set for $\ker \phi$
2. A generating set for $\phi(Aut_e(X_{r-1}))$

3.2 A generating set for $\ker \phi$

The proof of Tutte's theorem infact gave us the algorithm. Observe that for every vertex $v \in X_r$ is attached to atmost two vertices of $X_{r+1} \setminus X_r$ and an automorphism could possible swap these two neighbours of v.

When would this not be possible? Precisely when the neighbourhoods of the two vertices are different! Thus if $w_1, w_2 \in X_{r+1} \setminus X_r$ are neighbours of $v \in X_r$, a transposition (w_1, w_2) would be a valid automorphism fixing X_r *if and only if* $\Gamma(w_1) = \Gamma(w_2)$. And this can be easily checked by inspection. Thus in linear time, we can infact get a generating set (a set of disjoint transpositions) for $\ker \phi$.

3.3 A generating set for $\phi(Aut_e(X_{r+1}))$

This is the harder part. Since we'll be referring to vertices of $X_{r+1} \setminus X_r$, we shall refer to this set as V_r. Note that every vertex $v \in V_r$ is connected to 1 or 2 or 3 neighbours of X_r. Hence, let A be the collection of all subsets of X_r of size 1 or 2 or 3. Then we have the following neighbourhood map $\Gamma_r : V_r \to A$ which takes each vertex to the set of neighbours in the graph X_{r+1}.

Note that for every automorphism $\sigma \in Aut_e(X_{r+1})$, $\Gamma_r(\sigma(v)) = \sigma(\Gamma_r(v))$, and further if it were in the kernel, then $\Gamma_r(v) = \sigma(\Gamma_r(v))$. Call two vertices $v_1, v_2 \in V_r$ as *twins* if $\Gamma_r(v_1) = \Gamma_r(v_2)$.

Hence, any $\sigma \in \phi(Aut_e(X_{r+1}))$ has to stabilize the set of fathers with just 1 son.

$$A_1 = \{a \in A \ : \ a = \Gamma_r(v) \text{ for some unique } v \in V_r\}$$

σ must also stabilize the set of fathers of twins,

$$A_2 = \{a \in A \ : \ a = \Gamma_r(v_1) = \Gamma_r(v_2), v_i \neq v_j\}$$

And apart from reaching out to vertices of V_r, the next layer also induces edges between vertices of X_r, and any automorphism must certainly preserve these as well.

$$A_3 = \{\{w_1, w_2\} \in A \ : \ (w_1, w_2) \in X_{r+1}\}$$

Infact, these are all we need to ensure so that $\sigma \in Aut_e(X_r)$ is infact in $\phi(Aut_e(X_{r+1}))$.

Claim 2. *The image is precisely those automorphisms $\sigma \in Aut_e(X_r)$ which stabilize the sets A_1, A_2 and A_3.*

Proof. We need to show that if σ stabilizes the three sets, then we can extend it to an automorphism of X_{r+1}. The extension is built as follows:

- For each single child v, $\Gamma_r(v) \in A_1$, since $\sigma(\Gamma_r(v)) \in A_1$, map v to the only child of $\sigma(\Gamma_r(v))$.

- For each pair of twins v_1, v_2, $\Gamma_r(v_1) = \Gamma_r(v_2) \in A_2$, since $\sigma(\Gamma_r(v)) \in A_2$, map $\{v_1, v_2\}$ to the sons of $\sigma(\Gamma_r(v_1))$ in any order.

The constructions enforces that it respects edges between X_r and V_r. And since it also stabilizes A_3, σ also respects the edges between vertices of X_r that were newly formed. Hence σ is indeed can be extended to an automorphism of X_{r+1}. $\square$

Now we have reduced the isomorphism problem to a set-stabilizer problem for 2-groups. We shall discuss how to deal with it in the next class.