Chapter 68
Graph Isomorphism Testing without Numerics for Graphs of Bounded Eigenvalue Multiplicity

Martin Fürer *

Abstract

There are several parameterized classes of graphs for which polynomial time isomorphism tests are known. Attempts have been made to develop one conceptionally simple parameterized class of algorithms to solve the graph isomorphism problem for all of these classes. Such unified algorithms have been designed to handle almost all of these classes except for the case of bounded eigenvalue multiplicity. It is shown here that this case can also be handled in a more direct way by discrete methods. The new algorithm uses combinatorics and group theory closely related to the methods used for the other feasible classes of graphs.

The classical polynomial time graph isomorphism test of Babai, Grigoriev and Mount for graphs of bounded eigenvalue multiplicity consists of two distinct parts. First, in the linear algebra part, numerical approximations of all eigenvalues and projections of the basis vectors into the eigenspaces are computed. The precision has to be chosen carefully to ensure that it is decidable whether two such projections are equal or have equal length. Also equal angles between such projections have to be recognized. In a second combinatorial and group theoretical part, this information is used to try isomorphisms in the projections and either to combine them to a global isomorphism or to detect that none exists.

The numerical part is alien to such a discrete mathematical problem. A direct combinatorial approach is more natural and gives more insight. It is shown that such an approach is indeed possible. It is an important step towards one unified algorithm for the graph isomorphism problem for all natural polynomially solvable classes. It helps understanding under which circumstances computationally feasible isomorphism tests are possible.

1 Introduction

Graphs are flexible enough to easily encode any finite structure. Therefore the graph isomorphism problem is of fundamental importance. It asks for an efficient algorithm to decide whether two finite structures are intrinsically distinct or merely different representations of basically the same structure.

Despite great efforts over decades, no polynomial time algorithm is known for the graph isomorphism problem. On the other hand, the problem is not assumed to be NP-complete, as this would have the following very strange consequences. It would imply $\#P \subseteq NP$ [11] meaning that counting the number of solutions would not be harder than deciding whether a solution exists for any NP-complete problem. A further consequence would be $co\text{-}NP \subseteq AM$ forcing the polynomial hierarchy to collapse [3, ?].

There are several interesting parameters of graphs with a polynomial time isomorphism test for the classes of graphs with bounded values for their parameters. The most prominent examples are

- The graphs of bounded genus (see, e.g., [12])
- The graphs of bounded degree (also called valence) [10]
- The graphs of bounded eigenvalue multiplicity [2]
- The graphs of bounded color size [1]

*Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA 16802, furer@cse.psu.edu and Department of Computer Science, Princeton University, Princeton, NJ 08544. This work is supported in part by NSF grant CCR-9218309 and DIMACS.

It is an obvious question to ask about the essential ingredients of these classes that makes their isomorphism problem feasible. In particular, it would be interesting to know whether there is a nice algorithm handling all these classes. Does there exist an efficient, conceptually simple and uniform algorithm such that several or all of these feasible classes, with arbitrary fixed parameter value, could be tested for isomorphisms in polynomial time by this single algorithm. Furthermore, such an algorithm might solve the isomorphism problem efficiently for some new classes not contained in this list. There have been attempts to handle these problems. Miller [12] has been able to handle a strict generalization of bounded degree and bounded genus graphs. Finally, Ponomarenko [13] has presented such an algorithm that solves all known feasible classes except for the graphs of bounded eigenvalue multiplicity. The latter class is only solved by his uniform method if the automorphism group is primitive, which is a severely restricted case.

Indeed, the known polynomial time graph isomorphism test of Babai, Grigoriev and Mount [2] for graphs of bounded eigenvalue multiplicity has a very different flavor than any other classical graph isomorphism test. To solve this discrete combinatorial problem, it involves numerical computations of eigenvalues and eigenvectors. These quantities are computed with sufficient numerical precision to make it decidable which projections of different standard basis vectors into given eigenspaces have equal length or form equal angles. Once all these discrete results are known, the algorithm continues in a discrete manner reminiscent of other isomorphism tests.

The purpose of this paper is to propose an efficient discrete isomorphism test for graphs of bounded eigenvalue multiplicity, completely avoiding any numerical approximations. Furthermore, the algorithm explores the power of edge coloring and its important relation to spectral properties of graphs. Vertex and edge coloring have often been claimed to solve the graph isomorphism problem completely. Many such claims have been made in the chemical literature, which is not astonishing, because molecule graphs can hardly be so complicated as to defy this simple approach. In fact, this approach alone is sufficiently strong to handle random graphs and random regular graphs.

Edge coloring alone fails to distinguish between two nonisomorphic strongly regular graphs with the same parameters. Nevertheless, edge coloring is not only the most important preprocessing tool for practical isomorphism tests. Its theoretical power in cooperation with other combinatorial, algebraic, and group theoretical methods calls for further examination. The investigations of stable edge colorings have already led to the development of important algebraic disciplines studying association schemes and coherent configurations [15, 5, 6, 8, 9].

Not only are vertex coloring and edge coloring not sufficiently strong to identify the orbits of the automorphism group of a graph, the natural generalization from vertex and edge coloring to coloring of k-tuples has been shown to fail too [4]. In fact, it fails not only for every constant k, but also for every $k = o(n)$ even for graphs of degree 3 and color class size 4.

2 Notation and Background

In this section, we relate eigenspaces to graphs, and define stable edge colorings.

All input graphs considered are finite and undirected. We denote graphs by $X = (V, E)$ with $|V| = n$ and adjacency matrix A. With every graph we associate the vector space of its vertex labelings.

DEFINITION 2.1. *The* vector space of vertex labelings *is the set of functions from V into the real numbers. The value of such a function in a vertex v is often called the* label of v *or the* v-component x_v *of the vector x. The* standard basis *of this vector space consists of the functions assigning value* 1 *to one vertex and value* 0 *to all the others. We use the symbol v not only for the vertex v, but also to denote the standard basis vector with value* 1 *in vertex v and value* 0 *everywhere else.*

Throughout this paper, the vector space of vertex labelings, its isomorphic image $\mathbf{R}^n$ (obtained through an arbitrary enumeration of the standard basis), and the subspaces of these spaces

are the only vector spaces considered. All eigenvalues and eigenvectors studied are eigenvalues and eigenvectors of the linear mapping defined by the adjacency matrix A. Since A is symmetric and real, its eigenvalues are real too, and there exists an orthonormal basis consisting of eigenvectors. The spectrum of an adjacency matrix A (i.e., the eigenvalues with multiplicities) is independent of the enumeration of the vertices. In other words, the spectrum is an invariant of the isomorphism class of a graph. Therefore, it is also called the spectrum of a graph.

The linear mapping $x \mapsto Ax$ associated with the adjacency matrix A can be viewed as follows. An old labeling is transformed into a new one. The new label of any vertex v, is the sum of the old labels over all neighbors of v. With this picture in mind, one can easily "see" the eigenvectors of some simple graphs.

Example. The path of length 2 has the following basis of eigenvectors: $(1, \pm\sqrt{2}, 1)$ with eigenvalues $\lambda_1 = \sqrt{2}$, $\lambda_3 = -\sqrt{2}$, and $(1, 0, 1)$ with eigenvalue $\lambda_2 = 0$.

Even though the original input graphs are undirected, we will study complete directed graphs with colored edges. Vertex and edge colors are an old tool to exhibit different kinds of edges. The idea is to color an edge (u, v) different from an edge (u', v'), as soon as a testing algorithm discovers that no automorphism can map (u, v) to (u', v'). Every ordinary graph can be viewed as a complete colored graph with all original edges colored black and all original non-edges colored white. The vertices v are identified with the pairs (v, v) and are colored with a third color. The adjacency matrix A is still the original adjacency matrix uneffected by the coloring. The intention of using a coloring is to identify different "kinds" of edges. Idealy we would like to color two edges differently iff they do not lie on the same orbit of the automorphism group. As no efficient algorithm is known to do that (otherwise, the graph isomorphism problem would be solved), we are a bit more modest and only color two edges differently when we have discovered that they cannot possibly lie on the same orbit.

More precisely, a *coloring refinement step* is defined as follows. If the old colors of two edges are already distinct, then the new colors remain so. Furthermore, the new colors of two edges (u, v) and (u', v') are distinct if there is a pair of colors, say (red, green) such that the number of directed paths of length 2 colored (red, green) from u to v is different from the number of such paths from u' to v'. A coloring is called stable if no such refinement is possible. Stable edge colorings can be computed in time $O(n^3 \log n)$ [7].

Vertex coloring is even simpler. Initially all vertices have the same color. During a refinement step, two vertices of the same old color receive different new colors, if they differ in the number of adjacent vertices of any old color. Therefore, after the first refinement step, the vertices are colored by their degrees. Later the vertices are colored by the multiset of neighbor colors. (In a multiset, every element is given with its multiplicity.)

Let $\mathcal{C}^1$ be the stable vertex coloring produced by this iterative vertex coloring algorithm. We can get another vertex coloring $\mathcal{C}^2$ by using the edge coloring algorithm. We just consider the color of the loop edge (v, v) to be the color of v.

The stable coloring $\mathcal{C}^2$ of the vertices obtained by edge coloring is always a refinement of the stable coloring $\mathcal{C}^1$ obtained by vertex coloring. Often it is a strict refinement.

In addition to the vertex coloring $\mathcal{C}^1$ obtained by starting with the trivial vertex coloring, we study the stable colorings $\mathcal{C}_v$ obtained by starting with a coloring distinguishing the vertex v, i.e., the vertex v starts with one color, and all other vertices start with the same other color.

We also denote by $\mathcal{C}^1$, $\mathcal{C}_v$, $\mathcal{C}^2$ any stable refinements of the colorings defined above.

We can easily make colorings of edges and vertices canonical. This implies that isomorphic graphs are colored in such a way that vertices or edges of any color red, are mapped by any isomorphism to red vertices or edges.

To achieve a canonical stable coloring, it is sufficient to define a canonical order on the initial colors and to show how to maintain a canonical order over each refinement step. The start is straightforward. Let us say that non-edges are colored with color 0, edges with color 1, and vertices with color 2. When the old colors are already ordered at some stage, the new colors

after one additional step can initially be viewed as vectors of multiplicities (where the ith component of the new vector color of a vertex v is the number of neighbors of v with old color i). Now the vector colors are ordered lexicographically and can be replaced by ordinary integer colors before the next step.

DEFINITION 2.2. *Two stable colorings $\mathcal{C}$ and $\mathcal{C}'$ are* similar *if the number of vertices colored by any color in $\mathcal{C}$ is equal to the number of vertices colored by the same color in $\mathcal{C}'$.*

For stable edge colorings, we know that any vertex color determines the multiset of incident edge colors. Therefore, in similar edge colorings, every color occurs equally often in both colorings.

3 Projections on Eigenspaces

Let $\mathcal{C}$ be some vertex coloring, and let $C(v)$ be the color class of $\mathcal{C}$ containing vertex v. We define the averaging matrix M associated with the vertex coloring $\mathcal{C}$ by

$$M_{uv} = \begin{cases} \frac{1}{|C(v)|} & \text{if } C(u) = C(v) \\ 0 & \text{otherwise} \end{cases}$$

LEMMA 3.1. *The following statements are equivalent:*

(a) The coloring $\mathcal{C}$ is stable.

(b) The averaging matrix M of the coloring $\mathcal{C}$ commutes with the adjacency matrix A, i.e., $AM = MA$.

(c) AM is a symmetric matrix.

Proof. (b) and (c) are clearly equivalent, because the matrices A and M are symmetric. If $AM = MA$, then

$$(AM)^T = M^T A^T = MA = AM$$

If $(AM)^T = AM$, then

$$AM = (AM)^T = M^T A^T = MA$$

For the equivalence of (a) and (b), Let $E_{UU'}$ be the number of edges from vertices of U to vertices of U'. Then

$$\begin{aligned} (MA)_{uw} &= \sum_{v \in V} M_{uv} A_{vw} \\ &= \sum_{v \in C(u)} M_{uv} A_{vw} \\ &= \frac{1}{|C(u)|} \sum_{v \in C(u)} A_{vw} \\ &= \frac{1}{|C(u)|} E_{C(u)\{w\}} \end{aligned}$$

$$\begin{aligned} (AM)_{uw} &= \sum_{v \in V} A_{uv} M_{vw} \\ &= \sum_{v \in C(w)} A_{uv} M_{vw} \\ &= \frac{1}{|C(w)|} \sum_{v \in C(w)} A_{uv} \\ &= \frac{1}{|C(w)|} E_{\{u\}C(w)} \end{aligned}$$

If $\mathcal{C}$ is stable, then the two expressions

$$\frac{1}{|C(u)|} E_{C(u)\{w\}}$$

and

$$\frac{1}{|C(w)|} E_{\{u\}C(w)}$$

for $(MA)_{uw}$ and $(AM)_{uw}$ are equal, because

$$E_{C(u)\{w\}} = \frac{1}{|C(w)|} E_{C(w)C(u)}$$

and

$$E_{\{u\}C(w)} = \frac{1}{|C(u)|} E_{C(w)C(u)}$$

If $\mathcal{C}$ is not stable, then there exists a vertex u and a color class $C(w)$ such that u's degree into the color class $C(w)$ is more than the average for vertices in $C(u)$, i.e.,

$$E_{\{u\}C(w)} > \frac{1}{|C(u)|} E_{C(w)C(u)}$$

On the other hand, every set $C(w)$ contains a vertex w whose degree into the color class $C(u)$ is not more than the average for vertices of $C(w)$, i.e.,

$$E_{C(u)\{w\}} \leq \frac{1}{|C(w)|} E_{C(w)C(u)}$$

In this case the two previous expressions for $(MA)_{uw}$ and $(AM)_{uw}$ are not equal. ∎

THEOREM 3.1. *Let x be an eigenvector of the adjacency matrix A for an eigenvalue λ. Let $\bar{x}$ be obtained by averaging the components of x over each color class of any stable vertex coloring. Then $\bar{x}$ belongs to the eigenspace of the eigenvalue λ too.*

Proof. Using Lemma 3.1 and observing that $\bar{x} = Mx$, we get

$$A\bar{x} = AMx = MAx = M\lambda x = \lambda Mx = \lambda\bar{x}$$

Hence, $\bar{x}$ belongs to the same eigenspace as x. ∎

One should note that $\bar{x}$ is not necessarily an eigenvector, as it might be $\vec{0}$.

THEOREM 3.2. *The projection p of a standard basis vector v into any eigenspace S has constant components on each color class of $\mathcal{C}_v$.*

Proof. Let $\bar{p}$ be obtained from p by averaging the components of P over each color class of $\mathcal{C}_v$. If v is orthogonal to S, then $p = \vec{0}$ and the claim of the Theorem is trivial. Otherwise, the v-component p_v of p is not zero. As v forms a color class of its own, $\bar{p}_v = p_v$, and therefore $\bar{p}$ is not zero either. The Cauchy-Schwarz Inequality implies $|\bar{p}| \leq |p|$. Therefore, we get

$$v \cdot \frac{p}{|p|} = \frac{p_v}{|p|} \leq \frac{\bar{p}_v}{|\bar{p}|} = v \cdot \frac{\bar{p}}{|\bar{p}|}$$

Now we use the fact that a unit vector in the direction of the projection of v forms an inner product with v that is the unique minimum among all inner products of unit vectors of S with v. This implies that p and $\bar{p}$ have the same direction. As they have also the same non-zero v-component, they are equal. ∎

COROLLARY 3.1. *The number of eigenspaces containing non-zero projections of v is less than or equal to the number of colors in $\mathcal{C}_v$.*

Proof. By Theorem 3.2 all these projections have constant components on the color classes. But the dimension of the subspace of vectors having constant components on the color classes is equal to the number of color classes $|C_v^1|$. As all these projections are pairwise orthogonal, at most $|C_v^1|$ of them can be non-zero. ∎

DEFINITION 3.1. *Let $\mathcal{C}_u$ and $\mathcal{C}_v$ be the stable colorings obtained by the same canonical coloring algorithm applied to the same graph with distinguished vertex u and v respectively. Let the color classes be $0_u, 1_u, 2_u, \ldots$ and $0_v, 1_v, 2_v, \ldots$ with $0_u = \{u\}$ and $0_v = \{v\}$. Then the coloring $\mathcal{C}_{uv}$ is defined by forming the vertex sets $0_u \cup 0_v$, $1_u \cup 1_v$, $2_u \cup 2_v, \ldots$ and then replacing any overlapping vertex sets by their unions.*

COROLLARY 3.2. *Vectors p which are common projections of u and v into some eigenspace are constant on the color classes of $\mathcal{C}_{uv}$.*

Proof. This is an immediate consequence of Theorem 3.2 and Definition 3.1. ∎

DEFINITION 3.2. *For every stable vertex coloring $\mathcal{C}$, the* color respecting space $\tilde{\mathcal{C}}$ *is the subspace of the vector space of all vertex labelings defined by having equal labels on equally colored vertices.*

DEFINITION 3.3. *Two vectors $x \in \tilde{\mathcal{C}}_u$ and $y \in \tilde{\mathcal{C}}_v$ are similar, if $\mathcal{C}_u$ is similar to $\mathcal{C}_v$, and equally colored components of x and y have equal values.*

With any permutation π of the vertices, we associate the corresponding permutation matrix P. This is a matrix with 0 and 1 entries defined by

$$P_{uv} = 1 \text{ iff } \pi(v) = u$$

LEMMA 3.2. *If $\mathcal{C}_u$ is similar to $\mathcal{C}_v$ and x is an eigenvector in $\tilde{\mathcal{C}}_u$ to some eigenvalue λ, then there exists a permutation π and its associated permutation matrix P such that Px is a vector similar to x, and Px is an eigenvector in $\tilde{\mathcal{C}}_v$ to the eigenvalue λ.*

Proof. If $\mathcal{C}_u$ is similar to $\mathcal{C}_v$, then we can choose π to be any color respecting permutation of the vertices (i.e., the color of any vertex w in $\mathcal{C}_u$ is equal to the color of $\pi(u)$ in $\mathcal{C}_v$. The corresponding permutation matrix P maps the eigenvector x into a vector $y = Px$. This vector y is an eigenvector, and it has the same eigenvalue λ as x, because corresponding vertices (under π) not only have equal labels, but (as the similar colorings are stable) also have an equal number of neighbors with any given label. ∎

LEMMA 3.3. *If π is a permutation of the components of x and y with permutation matrix P, then the inner product $x \cdot y$ is equal to the inner product $Px \cdot Py$.*

Proof. $Px \cdot Py = (Px)^T \cdot Py = x^T P^T P y = x^T y = x \cdot y$ ∎

THEOREM 3.3. *If C_u and C_v are similar vertex colorings, then for every eigenspace S, u and v have similar projections into S. In particular, these projections $pr_S(u)$ and $pr_S(v)$ have equal lengths.*

Proof. Let $\mathrm{pr}^1(u), \mathrm{pr}^2(u), \ldots, \mathrm{pr}^m(u)$ be the collection of all projections of u into the eigenspaces. By Theorem 3.2, these projections belong to $\tilde{C}_u$. They are constant on every vertex color class of C_u. For every vector in $\tilde{C}_u$, there exists a corresponding vector in $\tilde{C}_v$ with the same component values on equally colored vertices.

Let $q^1, \ldots, q^m$ be a sequence of vectors similar to $\mathrm{pr}^1(u), \mathrm{pr}^2(u), \ldots, \mathrm{pr}^m(u)$ respectively, in the graph colored by C_v. As the eigenspaces are orthogonal and span the whole space, we have

$$\sum_{i=1}^{m} \mathrm{pr}^i(u) = u$$

and by similarity

$$\sum_{i=1}^{m} q^i(v) = v$$

As $\mathrm{pr}^1(u), \mathrm{pr}^2(u), \ldots, \mathrm{pr}^m(u)$ are pairwise orthogonal eigenvectors to different eigenvalues, also $q^1, \ldots, q^m$ are pairwise orthogonal by Lemma 3.3 and eigenvectors to different eigenvalues by Lemma 3.2. Hence, $q^1, \ldots, q^m$ are the projections of v into the eigenspaces. Because corresponding projections of u and v are similar, they have equal length. ∎

THEOREM 3.4. *If the edges (u, v) and (u', v') have the same color in some stable edge coloring, then the inner products $pr_S(u) \cdot pr_S(v)$ and $pr_S(u') \cdot pr_S(v')$ of projections of standard basis vectors u and v into the same eigenspace S are equal.*

Proof. For every $w, w' \in V$, let $\mathrm{pr}_S(w)_{w'}$ be the w'-component of the projection $\mathrm{pr}_S(w)$ of the standard basis vector w into the eigenspace S. The edge color of (w, w') determines the vertex color of w' in C_w, which in turn (by Theorem 3.3) determines the value of the w'-component $\mathrm{pr}_S(w)_{w'}$ of the projection $\mathrm{pr}_S(w)$.

If the colors of (u, v) and (u', v') are equal, then for every pair of edge colors (red, green), the number of vertices w with (u, w) colored red and (v, w) colored green, is equal to the number of vertices w' with (u, w') colored red and (v, w') colored green. Because $\mathrm{pr}_S(u)_w = \mathrm{pr}_S(u')_{w'}$ and $\mathrm{pr}_S(v)_w = \mathrm{pr}_S(v')_{w'}$ are the same, the contribution $\mathrm{pr}_S(u)_w \mathrm{pr}_S(v)_w$ of each such vertex w to the inner product $\mathrm{pr}_S(u) \cdot \mathrm{pr}_S(v)$ is equal to the contribution $\mathrm{pr}_S(u')_{w'} \mathrm{pr}_S(v')_{w'}$ of each such vertex w' to the inner product $\mathrm{pr}_S(u') \cdot \mathrm{pr}_S(v')$. Hence, the two inner products are equal. ∎

COROLLARY 3.3. *All angles between projections of standard basis vectors into eigenspaces are determined by the edge colors of C^2.*

Proof. In any stable coloring, the edge color of (u, v) determines the colors of (u, u) and (v, v), which determine the lengths of the projections of u and v into eigenspaces by Theorem 3.3. Edge colors determine inner products by Theorem 3.4. Together they determine the cosines of the angles. ∎

COROLLARY 3.4. *For every eigenspace S, there is a set of edge colors in C^2, such that the color of any edge (u, v) is in the set, if and only if $pr_S(u) = pr_S(v)$.*

Proof. Equal projections into and eigenspace S are characterized by equal lengths and angle 0 in the projection. By Corollary 3.3, all lengths and angles in turn are determined by the edge colors. ∎

LEMMA 3.4. *The projection of a basis of any vector space into any subspace spans the subspace.*

Proof. Every vector x is spanned by the basis $z_1, \ldots, z_n$ as $x = \sum_{i=1}^{n} a_i z_i$. If x is in the subspace S, then

$$x = \mathrm{pr}_S(x) = \mathrm{pr}_S(\sum_{i=1}^{n} a_i z_i) = \sum_{i=1}^{n} a_i \mathrm{pr}_S(z_i)$$

∎

LEMMA 3.5. *$\tilde{\mathcal{C}}_u$ is spanned by eigenvectors.*

Proof. As our graphs are undirected, there exists a basis of eigenvectors for the whole vector space of vertex labelings. The averaging matrix M associated with $\mathcal{C}_u$ has rank equal to the number of color classes $|\mathcal{C}_u|$. It maps eigenspaces into eigenspaces by Theorem 3.1. Therefore, it maps any basis of eigenvectors into a set of vectors spanning a $|\mathcal{C}_u|$-dimensional subspace of $\tilde{\mathcal{C}}_u$. As $\tilde{\mathcal{C}}_u$ itself has only dimension $|\mathcal{C}_u|$, this subspace has to be the whole space $\tilde{\mathcal{C}}_u$. ∎

LEMMA 3.6. *For any pair of vertices (u, v), the subspace $\tilde{\mathcal{C}}_{uv}$ of vectors being constant on any color class of $\mathcal{C}_{uv}$ is spanned by eigenvectors.*

Proof. Just replace $\mathcal{C}_u$ by $\mathcal{C}_{uv}$ in the proof of Lemma 3.5. ∎

It remains to be shown that for every pair of vertices u, v, belonging to the same $\mathcal{C}^2$ vertex color class, there is an eigenspace S such that u and v have a common projection into S. Furthermore, this is a common projection of all vertices from the color class of u and v in $\mathcal{C}_{uv}$. In addition, if $\mathcal{C}_{uv}$ has any other color class, then such an eigenspace S can be chosen such that not every vertex has the same projection.

This allows our algorithm to partition the equally colored vertices very much the same way as done by Babai, Grigoriev, and Mount [2], even though, we have no handle on the eigenspaces themselves and cannot characterize which eigenspace has produced a partition.

4 The Algorithm

The new algorithm starts by producing a canonical stable vertex coloring $\mathcal{C}$ from scratch and also for each vertex v a canonical stable vertex coloring $\mathcal{C}_v$ with distinguished vertex v. Furthermore, it has to identify corresponding final colors when starting with different distinguished vertices. The simplest way to get all this done together is to run the edge coloring algorithm (without any distinct vertices). We just view the color of edge (u, v) as the color of v in the edge coloring with distinct vertex u. This method might actually produce a finer coloring than strictly required, but that does never do any harm. Anyway, we will need the edge coloring too.

Noticing that every automorphism will respect this coloring of vertices and edges, we build now a tree on top of every vertex color class. The leaves of each such tree are the vertices of one color class. All automorphisms will respect the trees too. Hence, these trees have properties similar to imprimitivity trees, but we neither insist that the automorphism group should act transitively on the vertex color classes, nor that it should act primitive on all the children of any node.

The formation process for these trees is quite simple. Initially, the trees all have height one. The children of each root are all the vertices of one color class. Pairs of vertices in the forest are always colored by the set of colors of graph edges between their descendent leaves. As long as there is an edge color say green such that some children of a tree node u are connected by a green edge, but not all of them are connected by a green path, we insert a new layer. The node u is now the grandparent of its previous children. All components (of these previous children) connected by green paths, obtain their own parent. This process stops, when no such refinement (by inserting new levels) is possible anymore.

In each such refinement step, we consider a finer partitioning of the vertex set of G corresponding to some level of a tree. The partitioning is defined by putting two vertices into the same class if we have not yet handled a tree node that is an ancestor of only one of them. Corresponding to the finer partitioning, we have a new eigenvector with constant values on the classes of the finer, but not the coarser partitioning. The finer partitioning is indeed produced by reclassifying vertices according to the projection on its eigenspace. Therefore, as in the previous algorithm of Babai, Grigoriev, and Mount [2], there are only polynomially many possibilities to try to extend any single automorphism of the upper parts of the tree to include the new layer.

A crucial observation is how to extend these automorphisms. Babai, Grigoriev, and Mount [2] have used vector space considerations. The automorphism has to be a linear mapping and is therefore defined by its images on a basis. We have not identified the projections onto these eigenspaces numerically. Instead we extend the

automorphism based on Theorem 3.4.

Therefore, whenever the image of some projections has been selected, the images of linearly dependent projections are uniquely determined by the edge colors.

Handling every tree separately, we replace the numerical part in the algorithm of Babai, Grigoriev, and Mount [2]. The algorithm is followed by the group theoretical part of [2].

5 Conclusion

We have designed a very simple *discrete* algorithm $\mathcal{A}_{\text{Discrete}}$ that can be used to replace the linear algebra part $\mathcal{A}_{\text{Lin-Alg}}$ in the Graph isomorphism test of Babai Grigoriev and Mount [2]. The latter involves the computation of numerical approximations to eigenvalues and projections into eigenspaces. Our new algorithm retains the subsequent group theoretic part $\mathcal{A}_{\text{Group}}$ [2]. Even though the new algorithm is very simple, its analysis is more involved, because this algorithm has no immediate access to the eigenspaces. We summarize the result of this paper as follows.

THEOREM 5.1. *The algorithm $\mathcal{A}_{Discrete}$ together with $\mathcal{A}_{Group}$ solves the isomorphism problem for graphs of bounded eigenvalue multiplicity in polynomial time.*

The degree of the polynomial running time is the same as in the original algorithm. The advantages of the new algorithm are first its potential for a better theoretical understanding of the interplay between stable edge colorings and spectral properties of graphs, and second the fact that it is a discrete algorithm for a discrete problem. From a practical point of view, the results of such an algorithm are probably more reliable, because a typical implementation of the numerical approximation algorithm would hardly contain a rigorous analysis of the required precision. Although theoretically possible, it would be very tedious.

References

[1] László Babai, Monte Carlo Algorithms in Graph Isomorphism Testing, Tech. Rep. DMS 79-10, Université de Montréal, 1979.

[2] Babai, L., Grigoryev, D.Y., Mount, D.M., Isomorphism of Graphs with Bounded Eigenvalue Multiplicity, *Proceedings 14th ACM Symposium on Theory of Computing*, STOC (1982), 310–324.

[3] R.B. Boppana, J. Håstad, and S. Zachos. Does co-NP have Short Interactive Proofs? *IPL*, 25:127–132, 1987.

[4] Cai, Jin-Yi, Martin Fürer, Neil Immerman, An Optimal Lower Bound on the Number of Variables for Graph Identification, *Combinatorica* **12** (1992), 389–410.

[5] Ya. Yu. Gol'fand and M.H. Klin, On k-Regular Graphs, in *Algorithmic Research in Combinatorics*, Nauka Publ., Moscow, 1978, 76–85.

[6] D.G. Higman, Coherent Configurations I.: Ordinary Representation Theory, *Geometriae Dedicata* **4** (1975), 1–32.

[7] Neil Immerman and Eric S. Lander, Describing Graphs: A First-Order Approach to Graph Canonization, in *Complexity Theory Retrospective*, Alan Selman, ed., Springer-Verlag, 1990, 59–81.

[8] M.H. Klin, M.E. Muzichuk, and I.A. Faradžev, Cellular Rings and Groups of Automorphisms of Graphs, Introductory Article to a Book to be Published by D. Reidel Publ. Co.

[9] M.Ch. Klin, R. Pöschel, and K. Rosenbaum, Angewandte Algebra, Vieweg & Sohn Publ., Braunschweig 1988.

[10] Eugene M. Luks, Isomorphism of Graphs of Bounded Valence Can be Tested in Polynomial Time, *J. Comput. System Sci.* **25** (1982), 42–65.

[11] Rudolf Mathon, A Note On the Graph Isomorphism Counting Problem, *Inform. Proc. Let.* **8** (1979), 131–132.

[12] Gary Miller, Isomorphism of k-Contractible Graphs, a Generalization of Bounded Valence and Bounded Genus, *Information and Control* **56** (1983), 1–20.

[13] Ilja N. Ponomarenko, The Isomorphism Problem for Classes of Graphs, *Soviet Math. Dokl.* **39** (1989), 119–122.

[14] U. Schöning. Graph Isomorphism is in the Low Hierarchy. *J. Comput. Syst. Sci.*, 37:312–323, 1987.

[15] Boris Weisfeiler, ed., *On Construction and Identification of Graphs*, Lecture Notes in Mathematics 558, Springer, 1976.