Electronic Colloquium on Computational Complexity, Report No. 78 (2012)

Approximate Graph Isomorphism*

V. Arvind[1], Johannes Köbler[2], Sebastian Kuhnert[2], Yadu Vasudev[1]

[1] The Institute of Mathematical Sciences, Chennai, India
{arvind,yadu}@imsc.res.in
[2] Institut für Informatik, Humboldt-Universität zu Berlin, Germany
{koebler,kuhnert}@informatik.hu-berlin.de

Abstract. We study optimization versions of Graph Isomorphism. Given two graphs G_1, G_2, we are interested in finding a bijection π from $V(G_1)$ to $V(G_2)$ that maximizes the number of matches (edges mapped to edges or non-edges mapped to non-edges). We give an $n^{O(\log n)}$ time approximation scheme that for any constant factor $\alpha < 1$, computes an α-approximation. We prove this by combining the $n^{O(\log n)}$ time additive error approximation algorithm of Arora et al. [*Math. Program.,* 92, 2002] with a simple averaging algorithm. We also consider the corresponding minimization problem (of mismatches) and prove that it is NP-hard to α-approximate for any constant factor α. Further, we show that it is also NP-hard to approximate the maximum number of edges mapped to edges beyond a factor of 0.94.
We also explore these optimization problems for bounded color class graphs which is a well studied tractable special case of Graph Isomorphism. Surprisingly, the bounded color class case turns out to be harder than the uncolored case in the approximate setting.

1 Introduction

The graph isomorphism problem (GI for short) is a well-studied computational problem: Formally, given two graphs G_1 and G_2 on n vertices, decide if there exists a bijection $\pi\colon V(G_1) \to V(G_2)$ such that $(u, v) \in E_1$ iff $(\pi(u), \pi(v)) \in E_2$. It remains one of the few problems that are unlikely to be NP-complete and for which no polynomial time algorithm is known.

Though the fastest known graph isomorphism algorithm for general graphs has running time $2^{O(\sqrt{n \log n})}$ [5], polynomial-time algorithms are known for many interesting subclasses, e.g. bounded degree graphs [18], bounded genus graphs [20], and bounded eigenvalue multiplicity graphs [4].

Motivation and Related Work. In this paper we study a natural optimization problem corresponding to the graph isomorphism problem where the objective is to compute a bijection that *maximizes* the number of edges getting mapped to edges and non-edges getting mapped to non-edges. The main motivation for this study is to explore if approximate isomorphisms can be computed

* An abbreviated version of this paper appears in the proceedings of MFCS 2012.

ISSN 1433-8092

efficiently, given that the best known algorithm for computing exact isomorphisms has running time $2^{O(\sqrt{n \log n})}$. The starting point of our investigation is a well-known article of Arora, Frieze and Kaplan [2] in which they study approximation algorithms for a quadratic assignment problem based on randomized rounding. Among the various problems they study, they also observe that approximate graph isomorphisms between n vertex graphs can be computed up to *additive error* εn^2 in time $n^{O(\log n/\varepsilon^2)}$. We show that this algorithm can be modified to obtain a multiplicative error approximation scheme for the problem. However, when we consider other variants of approximate graph isomorphism, they turn out to be much harder algorithmically.

To the best of our knowledge, the only previous theoretical study of approximate graph isomorphism is this work of Arora, Frieze and Kaplan [2]. However, the problem of approximate isomorphism and more general notions of graph similarity and graph matching has been studied for several years by the pattern matching community; see e.g. the survey article [7]. That line of research is not really theoretical. It is based on heuristics that are experimentally studied without rigorous proofs of approximation guarantees. Similarly, the general problem of graph edit distance [9] also encompasses approximate graph isomorphism. Both graph matchings and graph edit distance give rise to a variety of natural computational problems that are well studied.

Optimization versions of graph isomorphism. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two input graphs on the same number n of vertices. We consider the following optimization problems:

- Max-EGI: Given G_1, G_2, find a bijection $\pi\colon V_1 \to V_2$ that maximizes the number of matched edges, i.e., $me(\pi) = \|\{(u,v) \in E_1 \mid (\pi(u), \pi(v)) \in E_2\}\|$.
- Max-PGI: Given G_1, G_2, find a bijection $\pi\colon V_1 \to V_2$ that maximizes matched vertex pairs, i.e., $mp(\pi) = me(\pi) + \|\{(u,v) \notin E_1 \mid (\pi(u), \pi(v)) \notin E_2\}\|$.
- Min-EGI: Given G_1, G_2, find a bijection $\pi\colon V_1 \to V_2$ that minimizes mismatched edges, i.e., $\overline{me}(\pi) = \|\{(u,v) \in E_1 \mid (\pi(u), \pi(v)) \notin E_2\}\|$.
- Min-PGI: Given G_1, G_2, find a bijection $\pi\colon V_1 \to V_2$ that minimizes mismatched pairs, i.e., $\overline{mp}(\pi) = \overline{me}(\pi) + \|\{(u,v) \notin E_1 \mid (\pi(u), \pi(v)) \in E_2\}\|$.

As mentioned above, Max-PGI was studied before in [2]. Max-EGI can also be viewed as an optimization variant of subgraph isomorphism.

Clearly, $mp(\pi) + \overline{mp}(\pi) = \binom{n}{2}$ and $me(\pi) + \overline{me}(\pi) = \|E_1\|$. Thus solving one of the maximization problems with additive error is equivalent to solving the corresponding minimization problem with the same additive error. However, the minimization problems behave differently for multiplicative factor approximations, so we study them separately.

Bounded color class graph isomorphism. A natural restriction of GI is to vertex-colored graphs (G_1, G_2) where $V(G_1) = C_1 \,\dot\cup\, C_2 \,\dot\cup\, \ldots \,\dot\cup\, C_m$ and $V(G_2) = C'_1 \,\dot\cup\, C'_2 \,\dot\cup\, \ldots \,\dot\cup\, C'_m$, and C_i, C'_i contain the vertices of G_1 and G_2, respectively, that are colored i. The problem is to compute a color-preserving isomorphism π between G_1 and G_2, i.e., an isomorphism π such that for any vertex u, u and $\pi(u)$ have the same color. The bounded color-class version GI_k of GI consists of instances such that $\|C_i\| = \|C_i'\| \leq k$ for all i. For GI_k, randomized [3] and deterministic [8] polynomial time algorithms are known.

It is, therefore, natural to study the optimization problems defined above in the setting of vertex-colored graphs where the objective function is optimized over all color-preserving bijections $\pi\colon V_1 \to V_2$. We denote these problems as $\mathsf{Max\text{-}PGI}_k$, $\mathsf{Max\text{-}EGI}_k$, $\mathsf{Min\text{-}PGI}_k$ and $\mathsf{Min\text{-}EGI}_k$, where k is a bound on the number of vertices having the same color.

Overview of the results. We first recall the notion of an α-approximation algorithm for an optimization problem. We call an algorithm $\mathcal{A}$ for a maximization problem an α-approximation algorithm, where $\alpha < 1$, if given an instance $\mathcal{I}$ of the problem with an optimum $\mathsf{OPT}(\mathcal{I})$, $\mathcal{A}$ outputs a solution with value $\mathcal{A}(\mathcal{I})$ such that $\mathcal{A}(\mathcal{I}) \geq \alpha\mathsf{OPT}(\mathcal{I})$. Similarly, for a minimization problem, we say $\mathcal{B}$ is a β-approximation algorithm for $\beta > 1$, if for any instance $\mathcal{I}$ of the problem with an optimum $\mathsf{OPT}(\mathcal{I})$, $\mathcal{B}$ outputs a solution with value $\mathcal{B}(\mathcal{I})$ such that $\mathcal{B}(\mathcal{I}) \leq \beta\mathsf{OPT}(\mathcal{I})$.

Theorem 1. *For any constant $\alpha < 1$, there is an α-approximation algorithm for* Max-PGI *running in time $n^{O(\log n/(1-\alpha)^4)}$.*

We obtain the α-approximation algorithm for Max-PGI by combining the $n^{O(\log n)}$ time additive error algorithm of [2] with a simple averaging algorithm.

Next we consider the Max-EGI problem. Langberg et al. [16] proved that there is no polynomial-time $(1/2+\varepsilon)$-approximation algorithm for the Maximum Graph Homomorphism problem for any constant $\varepsilon > 0$ assuming that a certain refutation problem has average-case hardness (for the definition and details of this assumption we refer the reader to [16]). We give a factor-preserving reduction from the Maximum Graph Homomorphism problem to Max-EGI thus obtaining the following result.

Theorem 2. *There is no $(\frac{1}{2}+\varepsilon)$-approximation algorithm for* Max-EGI *for any constant $\varepsilon > 0$ under the same average-case hardness assumption of [16].*

We observe that unlike in the case of GI_k, where polynomial time algorithms are known [3,8,19], in the optimization setting, these problems are computationally harder. We prove the following theorem by giving a factor-preserving reduction from Max-2Lin-2 (e.g. see [15]) to $\mathsf{Max\text{-}PGI}_k$ and $\mathsf{Max\text{-}EGI}_k$.

Theorem 3. *For any $k \geq 2$, $\mathsf{Max\text{-}PGI}_k$ and $\mathsf{Max\text{-}EGI}_k$ are* NP*-hard to approximate beyond a factor of* 0.94.

Since, assuming the Unique Games Conjecture (UGC for short) of Khot [14], it is NP-hard to approximate Max-2Lin-2 beyond a factor of 0.878 [15], the same bound holds under UGC for $\mathsf{Max\text{-}PGI}_k$ and $\mathsf{Max\text{-}EGI}_k$ by the same reduction. Since $\mathsf{Max\text{-}PGI}_k$ and $\mathsf{Max\text{-}EGI}_k$ are easily seen to be instances of generalized 2CSP, they have constant factor approximation algorithms, for a constant factor depending on k. In fact, it turns out that $\mathsf{Max\text{-}EGI}_2$ and $\mathsf{Max\text{-}PGI}_2$ are tightly classified by Max-2Lin-2 with almost matching upper and lower bounds (details are given in Section 2). However, we do not know of similar gap-preserving reductions from general unique games (with alphabet size more than 2) to $\mathsf{Max\text{-}PGI}_k$ or $\mathsf{Max\text{-}EGI}_k$ for larger values of k.

The following results show that the complexity of Min-PGI and Min-EGI is significantly different from Max-PGI and Max-EGI.

Theorem 4. *There is no polynomial time approximation algorithm for* Min-PGI *with any multiplicative approximation guarantee unless* $\mathsf{GI} \in \mathsf{P}$.

Theorem 5. Min-PGI *does not have a PTAS unless* $\mathsf{P} = \mathsf{NP}$.

Theorem 6. *There is no polynomial time approximation algorithm for* Min-EGI *with any multiplicative approximation guarantee unless* $\mathsf{P} = \mathsf{NP}$.

Finally, we turn our attention to the minimization problems $\mathsf{Min\text{-}PGI}_k$ and $\mathsf{Min\text{-}EGI}_k$ on bounded color-class graphs. We prove that $\mathsf{Min\text{-}PGI}_k$ is as hard as the minimization version of Max-2Lin-2, known in literature as the Min-Uncut problem, and that $\mathsf{Min\text{-}EGI}_4$ is inapproximable for any constant factor unless $\mathsf{P} = \mathsf{NP}$ by reducing the Nearest Codeword Problem (NCP) to it.

Outline of the paper. Our results on maximization problems are in Section 2, while Section 3 contains our results on the corresponding minimization problems. Section 4 concludes with some open problems.

2 Maximizing the number of matches

We first observe that computing optimal solutions to Max-PGI is NP-hard via a reduction from Clique.

Lemma 7. *Computing optimal solutions to* Max-PGI *instances is* NP*-hard.*

Proof. Let (G, k) be an instance of the Clique problem. Define the graphs $G_1 = G$ and $G_2 = K_k \cup \overline{K}_{n-k}$, i.e., a k-clique and $n - k$ isolated vertices. Let π_{opt} be a bijection that achieves the optimum value for this Max-PGI instance. Then G has a k-clique if and only if $mp(\pi_{opt}) = \binom{n}{2} - \|E_G\| + \binom{k}{2}$. ◻

Next we give a general method for combining an additive error approximation algorithm for Max-PGI with a simple averaging approximation algorithm to design an α-approximation algorithm for Max-PGI for any constant $\alpha < 1$.

Lemma 8. *Suppose $\mathcal{A}$ is an algorithm such that for any $\varepsilon > 0$, given a* Max-PGI *instance in form of two n-vertex graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, computes a bijection $\pi\colon V_1 \to V_2$ such that $mp(\pi) \geq \mathsf{OPT} - \varepsilon n^2$ in time $T(n, \varepsilon)$. Then there is an algorithm that computes for each $\alpha < 1$ an α-approximate solution for any* Max-PGI *instance (G_1, G_2) in time $O(T(n, (1-\alpha)^2/9) + n^3)$.*

Proof. Without loss of generality we can assume $V_1 = V_2 = [n]$. We denote the number of edges in G_i by t_i and the number of non-edges by $\bar{t}_i$. Notice that the optimum for $\mathsf{Max\text{-}PGI}$ satisfies $\mathsf{OPT} \le t_1 + \bar{t}_2$. Let $\pi\colon [n] \to [n]$ be a permutation chosen uniformly at random. Then, an easy calculation shows that the expected number s of matched pairs is

$$s = \frac{t_1 t_2 + \bar{t}_1 \bar{t}_2}{\binom{n}{2}} = \frac{\binom{n}{2} - \bar{t}_2}{\binom{n}{2}} t_1 + \frac{\bar{t}_2}{\binom{n}{2}} \left(\binom{n}{2} - t_1 \right) = t_1 + \bar{t}_2 - \frac{2 t_1 \bar{t}_2}{\binom{n}{2}}.$$

It is not hard to see that one can deterministically compute a permutation σ such that $mp(\sigma) \ge s$; we defer this detail to the end of the proof. We now show how this can be combined with the additive error approximation algorithm $\mathcal{A}$ for $\mathsf{Max\text{-}PGI}$ to obtain an α-approximation algorithm for $\mathsf{Max\text{-}PGI}$. This combined algorithm distinguishes two cases based on the number of edges and non-edges in G_1 and G_2, respectively.

Case 1 ($\min\{t_1, \bar{t}_2\} \le (1-\alpha)\binom{n}{2}/2$): In this case we compute a permutation σ with $mp(\sigma) \ge s$. Since

$$t_1 \bar{t}_2 = \max\{t_1, \bar{t}_2\} \min\{t_1, \bar{t}_2\} \le \left(t_1 + \bar{t}_2\right)(1-\alpha)\binom{n}{2}/2,$$

it follows that

$$t_1 + \bar{t}_2 - 2 t_1 \bar{t}_2 / \binom{n}{2} \ge \alpha \left(t_1 + \bar{t}_2\right) \ge \alpha \mathsf{OPT}.$$

Case 2 ($\min\{t_1, \bar{t}_2\} > (1-\alpha)\binom{n}{2}/2$): In this case we use algorithm $\mathcal{A}$ with $\varepsilon = (1-\alpha)^2/9$ to obtain a permutation π with $mp(\pi) \ge \mathsf{OPT} - \varepsilon n^2$. Since $t_1 + \bar{t}_2 + \bar{t}_1 + t_2 = 2\binom{n}{2}$, either $t_1 + \bar{t}_2 \le \binom{n}{2}$ or $\bar{t}_1 + t_2 < \binom{n}{2}$. Without loss of generality assume $t_1 + \bar{t}_2 \le \binom{n}{2}$ (otherwise we interchange G_1 and G_2), implying that either $t_1 \le \binom{n}{2}/2$ or $\bar{t}_2 \le \binom{n}{2}/2$. Further, since the expected value of $mp(\pi)$ when π is picked at random is $t_1 + \bar{t}_2 - 2t_1\bar{t}_2/\binom{n}{2}$, it follows that for sufficiently large n,

$$\mathsf{OPT} \ge t_1 - t_1 \bar{t}_2 / \binom{n}{2} + \bar{t}_2 - t_1 \bar{t}_2 / \binom{n}{2} \ge \frac{\min\{t_1, \bar{t}_2\}}{2} > \frac{1-\alpha}{4} \binom{n}{2} \ge \frac{\varepsilon n^2}{1-\alpha}.$$

Hence, $mp(\pi) \ge \mathsf{OPT} - \varepsilon n^2 \ge \alpha \mathsf{OPT}$.

It remains to show how a permutation which achieves at least the expected number s of matched pairs can be computed deterministically. Suppose that $\sigma\colon [i] \to [n]$ is a *partial permutation.* Let $\pi\colon [n] \to [n]$ be a random permutation that extends σ, i.e., $\pi(j) = \sigma(j)$ for $j \in [i]$. Let $s(\sigma)$ denote the expected number of matched pairs over random permutations π that extend σ. It is easy to see that we can compute $s(\sigma)$ in polynomial time. We do this by counting the pairs in three parts: (a) pairs with both end points in $[i]$, (b) pairs with both end points in $[n] \setminus [i]$, and (c) pairs with one end point in $[i]$ and the other in $[n] \setminus [i]$. Matched pairs of type (a) depend only on σ and can be counted straightaway.

The expected number of matched pairs of type (b) is computed exactly as s above (since π restricted on $[n]\setminus[i]$ is random). The expected number of matched pairs of type (c) is given by $\sum_{j\in[i]} \frac{n_j n_{\sigma(j)}+(n-i-n_j)(n-i-n_{\sigma(j)})}{n-i}$, where n_j is the number of neighbors of j in the graph G_1 contained in $[n]\setminus[i]$ and $n_{\sigma(j)}$ is the number of neighbors of $\sigma(j)$ in the graph G_2 contained in $[n]\setminus\{\sigma(l) \mid l\in[i]\}$. The entire computation of $s(\sigma)$ takes $O(n^2)$ time.

Now, for $k\in[n]\setminus\{\sigma(l)\mid l\in[i]\}$, let $\sigma_k\colon[i+1]\to[n]$ denote the extension of σ by setting $\sigma(i+1)=k$. Since a random extension π of σ can map $i+1$ uniformly to any $k\in[n]\setminus\{\sigma(l)\mid l\in[i]\}$ it follows that

$$s(\sigma)=\frac{1}{n-i}\sum_k s(\sigma_k),$$

where the summation is over all $k\in[n]\setminus\{\sigma(l)\mid l\in[i]\}$.

Furthermore, each $s(\sigma_k)$ is efficiently computable, as explained above. Reusing partial computations, we can find k such that $s(\sigma_k)\geq s(\sigma)$ in time $O(n^2)$. Continuing thus, when we fix the permutation on all of $[n]$ we obtain a σ with $mp(\sigma)\geq s$ in $O(n^3)$ time. $\square$

Note that any polynomial time additive ε-error algorithm for Max-PGI, i.e., an algorithm running in time $n^{\mathrm{poly}(1/\varepsilon)}$ with an additive error $\leq\varepsilon n^2$, gives a polynomial time α-approximation algorithm for Max-PGI running in time $n^{\mathrm{poly}(1/(1-\alpha))}$.

To complete the proof of Theorem 1, we formulate Max-PGI as an instance of a quadratic optimization problem called the Quadratic Assignment Problem (QAP for short) as was done in [2] and use an additive error approximation algorithm for the Quadratic Assignment Problem due to Arora, Frieze and Kaplan [2].

Given $\{c_{ijkl}\}_{1\leq i,j,k,l\leq n}$, the Quadratic Assignment Problem is to find an $n\times n$ permutation matrix $x=(x_{ij})$ that maximizes $val(x)=\sum_{i,j,k,l}c_{ijkl}x_{ij}x_{kl}$. An instance of Max-PGI consisting of graphs $G_1=([n],E_1)$ and $G_2=([n],E_2)$ can be naturally expressed as a QAP instance by setting

$$c_{ijkl}=\begin{cases}1 & \text{if } (i,k)\in E_1 \text{ and } (j,l)\in E_2 \text{ or } (i,k)\notin E_1 \text{ and } (j,l)\notin E_2\\ 0 & \text{otherwise.}\end{cases}$$

This ensures that $val(x)=mp(\pi_x)$ for all permutation matrices x with corresponding permutation π_x; in particular, the optimum solutions of the Max-PGI and QAP instances achieve the same value.

There is no polynomial time α-approximation algorithm for QAP for any $\alpha<1$ unless P = NP [2]. Arora, Frieze and Kaplan in [2] give a general quasi-polynomial time algorithm for QAP with an additive error. Formally, they prove the following theorem.

Theorem 9 **([2]).** *There is an algorithm that, given an instance of* QAP *where each of the c_{ijkl} is bounded in absolute value by a constant c and given an ε, finds an assignment to x_{ij} such that $val(x)\geq val(x^*)-\varepsilon n^2$ where x^* is the assignment which attains the optimum. The algorithm runs in time $n^{O(c^2\log n/\varepsilon^2)}$.*

Thus for the Max-PGI problem, using Theorem 9 we can find a permutation π such that $mp(\pi) \geq \mathsf{OPT} - \varepsilon n^2$ in time $n^{O(\log n/\varepsilon^2)}$. Combining this with Lemma 8, we get an α-approximation algorithm for Max-PGI running in time $n^{O(\log n/(1-\alpha)^4)}$ and this completes the proof of Theorem 1.

In contrast to the quasi-polynomial time approximation scheme for Max-PGI, we now show that Max-EGI is likely to be $(\frac{1}{2} + \varepsilon)$-hard to approximate. To this end, define the Maximum Graph Homomorphism problem (MGH) first studied in [16]. Given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, MGH asks for a mapping $\phi\colon V_1 \to V_2$ such that $\|\{(u, v) \in E_1 \mid (\phi(u), \phi(v)) \in E_2\}\|$ is maximized. Langberg et al. [16] proved that MGH is hard to approximate beyond a factor of $1/2+\varepsilon$ under a certain average case assumption. To prove Theorem 2, we give a factor-preserving reduction from MGH to Max-EGI.

Lemma 10. *There is a polynomial time algorithm that for a given* MGH *instance $\mathcal{I}$, constructs a* Max-EGI *instance $\mathcal{I}'$ with* $\mathsf{OPT}(\mathcal{I}) = \mathsf{OPT}(\mathcal{I}')$.

Proof. Given an MGH instance $\mathcal{I} = (G_1, G_2)$, we construct the Max-EGI instance $\mathcal{I}' = (G_1', G_2')$ as follows. The graphs G_1' and G_2' both have vertex set $V_1 \times V_2$. For each edge (u_1, v_1) in the graph G_1, we put a single edge between the vertices (u_1, w_2) and (v_1, w_2) in E_1', where w_2 is an arbitrary but fixed vertex in V_2, and for each edge (u_2, v_2) in the graph G_2, we put all $\|V_1\|^2$ edges between $V_1 \times \{u_2\}$ and $V_1 \times \{v_2\}$ in E_2'. It suffices to prove the following claim.

Claim. There is a mapping $\phi\colon V_1 \to V_2$ such that $\big\|\big\{(u, v) \in E_1 \mid (\phi(u), \phi(v)) \in E_2\big\}\big\| = k$ if and only if there is a permutation $\pi\colon V_1 \times V_2 \to V_1 \times V_2$ such that $\|\{(u, v) \in E_1' \mid (\pi(u), \pi(v)) \in E_2'\}\| = k$.

Given the mapping ϕ, we construct the permutation π as follows: For each $u_1 \in V_1$, π maps the vertex (u_1, w_2) of G_1' to the vertex $(u_1, \phi(u_1))$ in G_2'. The remaining $\|V_1\| \cdot \|V_2\| - \|V_1\|$ vertices of G_1' are mapped arbitrarily.

Then each edge $(u_1, v_1) \in E_1$ is satisfied by ϕ if and only if the corresponding edge between (u_1, w_2) and (v_1, w_2) in E_1' is satisfied by π. This follows from the fact that $(\phi(u_1), \phi(v_1)) \in E_2$ if and only there is an edge between $(u_1, \phi(u_1))$ and $(v_1, \phi(v_1))$ in E_2'.

Similarly, given a permutation π between G_1' and G_2', we can obtain a mapping $\phi\colon V_1 \to V_2$ achieving the same number of matched edges by letting $\phi(u_1) = v_2$, where v_2 is the second component of the vertex $\pi(u_1, w_2)$. ☐

Unlike in the case of Max-PGI, we observe that there cannot be constant factor approximation algorithms for Max-PGI$_k$ for all constants. This is in interesting contrast to the fact that GI for graphs with bounded color-class size is in P. We now prove the hardness of approximating Max-PGI$_k$ and Max-EGI$_k$ for any $k \geq 2$.

We prove the hardness by exhibiting a factor-preserving reduction from Max-2Lin-2, which is hard to approximate above a guarantee of 0.94 unless P = NP [12]. Given a set $E \subseteq \big\{x_i + x_j = b \mid i, j \in [n], b \in \{0, 1\}\big\}$ of m equations over $\mathbb{F}_2$, the problem Max-2Lin-2 is to find an assignment to the variables $x_1, \ldots, x_n$ that maximizes the number of equations satisfied.

The following lemma proves the factor-preserving reduction from Max-2Lin-2 to Max-PGI$_k$. The proof for Max-EGI$_k$ is similar.

Lemma 11. *For any $k \geq 2$, there is a polynomial time algorithm that for a given* Max-2Lin-2 *instance $\mathcal{I}$ constructs a* $\mathsf{Max\text{-}PGI}_{2k}$ *instance $\mathcal{I}'$ such that* $\mathsf{OPT}(\mathcal{I}') = (2k)^2 \mathsf{OPT}(\mathcal{I})$.

Proof. Let $E \subseteq \{x_i + x_j = b \mid i, j \in [n], b \in \{0,1\}\}$ be the equations of $\mathcal{I}$. As a first step, if there is a pair of equations $x_i + x_j = 1$ and $x_i + x_j = 0$ in E, remove both these equations and add a new equation $y_i + y_j = 1$ on two new variables y_i and y_j. Let E' be the new set of equations obtained. Notice that $\mathsf{OPT}(E) = \mathsf{OPT}(E')$. We now describe the construction of the instance $\mathcal{I}'$ of $\mathsf{Max\text{-}PGI}_{2k}$. For each variable x_i, put two sets of vertices V_i^0 and V_i^1 with k vertices each of color i. Let $x_l + x_m = b$ be an equation in E'. In the graph G_1, add a complete bipartite graph between V_l^0 and V_m^0 and another complete bipartite graph between V_l^1 and V_m^1. Similarly, add the complete bipartite graph between V_l^0 and V_m^b and between V_l^1 and $V_m^{1\oplus b}$ in G_2. If there is no equation in E' connecting the variables x_l and x_m, add a complete bipartite graph between the color classes l and m in G_1 and the empty graph between l and m in G_2. Similarly, make all color classes cliques in G_1 and independent sets in G_2. The idea is that assigning $x_i \mapsto 0$ corresponds to mapping V_i^0 and V_i^1 to themselves, respectively, while assigning $x_i \mapsto 1$ corresponds to mapping V_i^0 to V_i^1 and vice versa.

Given an assignment $\sigma \colon [n] \to \{0,1\}$ that satisfies t of the equations in E, let π_σ be the permutation that maps the j^{th} vertex in V_i^b to the j^{th} vertex in $V_i^{b\oplus\sigma(i)}$. For each satisfied equation $x_i + x_j = b$, this guarantees that all $(2k)^2$ pairs in $(V_i^0 \cup V_i^1) \times (V_j^0 \cup V_j^1)$ are matched. Thus $\mathsf{OPT}(\mathcal{I}') \geq mp(\pi_\sigma) = (2k)^2 t$.

To prove the converse, let $\pi \colon [n] \to [n]$ be a permutation with $mp(\pi) = t$. Define f_i as the number of vertices in V_i^0 that are mapped to V_i^1 by π (it is also the number of vertices in V_i^1 mapped to V_i^0). If $f_i \in \{0, k\}$ for all i, it is straightforward to reverse the above construction, obtaining an assignment that satisfies $mp(\pi)/(2k)^2$ equations.

If there is an i with $f_i \notin \{0, k\}$, let $mp_{i,j}(\pi)$ denote the number of matched pairs between color classes i and j. Thus $mp_{i,j}(\pi) = 4\left[(k - f_j)f_i + (k - f_i)f_j\right]$. Define $mp_i(\pi) = \sum_j mp_{i,j}(\pi)$, obtaining

$$\begin{aligned} mp_i(\pi) &= \sum_j 4\left[(k - f_j)f_i + (k - f_i)f_j\right] \\ &= k^2 \sum_j 4\left[\left(1 - \frac{f_j}{k}\right)\frac{f_i}{k} + \left(1 - \frac{f_i}{k}\right)\frac{f_j}{k}\right]. \end{aligned}$$

Let $mp_i'(\pi) = (1/k^2) mp_i(\pi) = \sum_j 4\left[(1 - f_j')f_i' + (1 - f_i')f_j'\right]$ where $f_i' = f_i/k$ and $f_j' = f_j/k$.

Define $\pi_{i,b}$ (for $b \in \{0,1\}$) as the permutation that maps the j^{th} vertex of $V_i^{b'}$ to the j^{th} vertex of $V_i^{b\oplus b'}$, and that acts like π on all other color classes. Thus, $mp_i(\pi_{i,0}) = 4\sum_j f_j'$ and $mp_i(\pi_{i,1}) = 4\sum_j (1 - f_j')$.

Since $mp_i'(\pi)$ is a convex combination of $mp_i(\pi_{i,0})$ and $mp_i(\pi_{i,1})$, one of the two must be at least as large as $mp_i'(\pi)$. Replace π by that permutation, and repeat this process until $f_i \in \{0, k\}$ for all i. $\square$

This construction still works if we replace $mp(\pi)$ with $me(\pi)$, as for all equations $x_i + x_j = b$ in E, exactly half of the possible edges between color classes i and j are present. It follows that there is a factor-preserving reduction from Max-2Lin-2 to Max-EGI$_{2k}$.

Lemma 12. *For any $k \geq 2$, there is a polynomial time algorithm that for a given* Max-2Lin-2 *instance $\mathcal{I}$ constructs a* Max-EGI$_{2k}$ *instance $\mathcal{I}'$ such that* $\mathsf{OPT}(\mathcal{I}') = 2k^2\mathsf{OPT}(\mathcal{I})$.

Since there is no α-approximation algorithm for Max-2Lin-2 for $\alpha > 0.94$ unless P = NP [12], Lemmas 11 and 12 complete the proof of Theorem 3 that there is no α-approximation algorithm for Max-PGI$_k$ and Max-EGI$_k$ for $\alpha > 0.94$ unless P = NP.

It is easy to see that for each constant $k > 0$, both Max-PGI$_k$ and Max-EGI$_k$ are subproblems of the generalized Max-2CSP(q), where q depends on k. Thus, both Max-PGI$_k$ and Max-EGI$_k$ have constant factor approximation algorithms by virtue of the semidefinite programming based approximation algorithm for Max-2CSP(q) [11]. The following lemma shows the reduction of Max-EGI$_2$ to Max-2CSP(2). The reduction from Max-PGI$_k$ and Max-EGI$_k$ to Max-2CSP(q) is similar.

Lemma 13. *There is a polynomial time algorithm that for two given vertex-colored graphs G_1 and G_2 where each color class has size at most 2, outputs a* Max-2CSP(2) *instance $\mathcal{F} = \{f_1, \ldots, f_m\}$ where $m = \|E(G_1)\|$ and $f_i \colon \{0,1\}^2 \to \{0,1\}$ such that there is a color-preserving bijection $\pi \colon V(G_1) \to V(G_2)$ with $me(\pi) = k$, if and only if there is an assignment which satisfies k constraints in $\mathcal{F}$.*

Proof. For each color class C_i, we assign a variable x_i. For an edge e from C_i to C_j in G_1, construct the function $f_e \colon \{0,1\}^2 \to \{0,1\}$ over the variables x_i and x_j as follows. Any Boolean assignment to the variables can be looked upon as a permutation: If $x_i \mapsto 0$, then we have the identity permutation on C_i, otherwise the permutation swaps the vertices of C_i. The value f_e on that particular assignment is 1 if the permutation that it corresponds to sends the edge e to an edge in G_2. Hence there is an assignment that satisfies k constraints if and only if there is a permutation π with $me(\pi) = k$. ◻

As the problem of Max-2CSP(2) has an approximation algorithm with a guarantee of 0.874 [17], this implies an approximation algorithm for Max-EGI$_2$ with the same guarantee and since Max-2Lin-2 is hard to approximate beyond 0.878 under UGC [15], we have almost matching upper and lower bounds for Max-EGI$_2$ under UGC.

3 Minimizing the number of mismatches

We first consider the problems Min-PGI and Min-EGI, where the objective is to minimize the number of mismatched pairs and edges, respectively.

Theorem 4. *There is no polynomial time approximation algorithm for* Min-PGI *with any multiplicative approximation guarantee unless* $\mathsf{GI} \in \mathsf{P}$.

Proof. Assume that there is a polynomial time α-approximation algorithm $\mathcal{A}$ for Min-PGI. If the two input graphs G_1 and G_2 are isomorphic, then there is a bijection $\pi\colon V_1 \to V_2$ such that $\overline{mp}(\pi) = 0$, and if G_1 and G_2 are not isomorphic, then $\overline{mp}(\pi) > 0$ for all π. Thus, it immediately follows that G_1 and G_2 are isomorphic, if and only if $\mathcal{A}$ outputs a bijection $\sigma\colon V(G_1) \to V(G_2)$ with $\overline{mp}(\sigma) = 0$ (i.e., an isomorphism). ◻

In order to show that it is unlikely that Min-PGI has a polynomial time approximation scheme, we give a gap-preserving reduction from the Vertex-disjoint Triangle Packing problem (VTP) defined as follows: Given a graph G find the maximum number of vertex-disjoint triangles that can be packed into G. We look at the corresponding gap version of the VTP problem.

Gap-VTP$_{\alpha,\beta}$: Given a graph G and $\alpha > \beta$,

1. Answer YES, if at least $\alpha n/3$ triangles can be packed into G.
2. Answer NO, if at most $\beta n/3$ triangles can be packed into G.

It is known that VTP does not have an algorithm which when given a graph and parameter α as input, computes a vertex-disjoint triangle packing of size at least $\alpha\mathsf{OPT}$ in time $O(n^{\mathrm{poly}(1/(1-\alpha))})$ unless $\mathsf{P} = \mathsf{NP}$ [6]. It is also known that for a fixed value of $\beta < 1$, Gap-VTP$_{1,\beta}$ is NP-hard on graphs of bounded degree [10,21]. Indeed, Petrank [21] gives a gap-preserving reduction from 3Sat to 3DimensionalMatching. It is not hard to see that replacing the hyperedges in the generated instances with triangles results in a gap-preserving reduction to VTP, as all triangles in the resulting graph correspond to a hyperedge. All vertices in the generated graph G have degree 4 or 6. Thus there is a β such that Gap-VTP$_{1,\beta}$ is NP-hard on such graphs. By attaching the gadget depicted in Fig. 1 to each vertex of degree 4 in G, we obtain a 6-regular graph G', which we again consider as VTP instance. Let n and n' denote the number of vertices in G and G', respectively. If G can be packed with $n/3$ vertex-disjoint triangles, then G' can also be packed fully by vertex-disjoint triangles. If $\mathsf{OPT}(G) \le \beta n/3$, then $\mathsf{OPT}(G') \le \big(1 - \frac{1-\beta}{13}\big)n'/3$. Thus there is a β' such that Gap-VTP$_{1,\beta'}$ is NP-hard on 6-uniform graphs.

Lemma 14. *Given a* Gap-VTP$_{\alpha,\beta}$ *instance* $\mathcal{I}$ *(a 6-uniform graph on n vertices), in polynomial time we can find an* Min-PGI *instance* $\mathcal{I}'$ *such that*

$$\mathsf{OPT}(\mathcal{I}) \ge \frac{\alpha n}{3} \Rightarrow \mathsf{OPT}(\mathcal{I}') \le 2n(2-\alpha)$$
$$\mathsf{OPT}(\mathcal{I}) \le \frac{\beta n}{3} \Rightarrow \mathsf{OPT}(\mathcal{I}') \ge \frac{2n}{3}(4-\beta)$$

This reduction together with the hardness of VTP proves Theorem 5.

Proof. Let the instance $\mathcal{I}$ of VTP be a 6-regular graph G on n vertices. We construct a Min-PGI instance $\mathcal{I}' = (G_1, G_2)$ as follows: $G_1 := G$ and G_2 is a

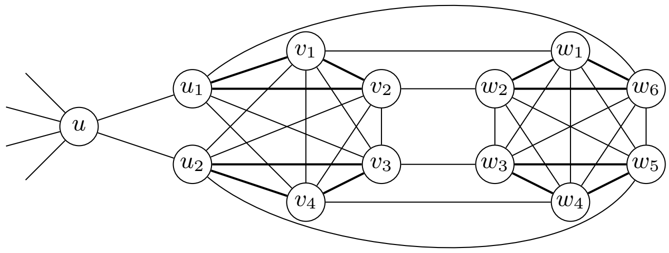

Fig. 1. Converting a VTP instance G of degrees 4 and 6 to a 6-uniform VTP instance G'

collection of $n/3$ vertex-disjoint triangles on the same vertex set as G_1, without any further edges. Suppose $\mathsf{OPT}(\mathcal{I}) \geq \alpha n/3$, then there is a permutation π that maps at least $\alpha n/3$ triangles to vertex-disjoint triangles of G_1. Hence the number of edges of G_1 that are mapped to non-edges of G_2 is at most $3n-\alpha n$. Similarly, the number of edges of G_2 that are images of non-edges of G_1 is at most $(1-\alpha)n$. Therefore, $\mathsf{OPT}(\mathcal{I}') \leq \overline{mp}(\pi) \leq 2n(2-\alpha)$.

Now suppose $\mathsf{OPT}(\mathcal{I}) \leq \beta n/3$. Since G_1 has at most $\beta n/3$ disjoint triangles, any permutation π maps at least $(1-\beta)n/3$ non-edges of G_1 to edges of G_2. Further, since G_1 has at least $2n$ edges more than G_2 and since already at least $(1-\beta)n/3$ of the edges of G_2 are images of non-edges of G_1, π maps at least $2n+(1-\beta)n/3$ edges of G_1 to non-edges of G_2. Thus we have

$$\overline{mp}(\pi) \geq \frac{n}{3}(1-\beta) + 2n + \frac{n}{3}(1-\beta) = \frac{2n}{3}(4-\beta). \qquad \square$$

Next we prove Theorem 6.

Theorem 6. *There is no polynomial time approximation algorithm for* $\mathsf{Min\text{-}EGI}$ *with any multiplicative approximation guarantee unless* $\mathsf{P} = \mathsf{NP}$.

Proof. The theorem follows from the following reduction from the Clique problem. Given an instance (G, k) of Clique, we construct the instance of $\mathsf{Min\text{-}EGI}$ as follows. G_1 consists of a k-clique and $n-k$ independent vertices, and $G_2 := G$. $(G, k) \in \mathsf{Clique}$ if and only if there exists a π such that in the $\mathsf{Min\text{-}EGI}$ problem $\overline{me}(\pi) = 0$. Hence any polynomial time approximation algorithm with a multiplicative guarantee for $\mathsf{Min\text{-}EGI}$ gives a polynomial time algorithm for Clique. $\square$

The input for the $\mathsf{Min\text{-}Uncut}$ problem is a set $E \subseteq \{x_i + x_j = 1 \mid i, j \in [n]\}$ of m equations. The objective is to minimize the number of equations that must be removed from the set E so that there is an assignment to the variables that satisfy all the equations. This problem is known to be MaxSNP-hard [13], and assuming the Unique Games Conjecture, hard to approximate within any constant factor [14]. The following lemma shows that $\mathsf{Min\text{-}PGI}_k$ is as hard as the $\mathsf{Min\text{-}Uncut}$ problem.

Lemma 15. *Let $\mathcal{I}$ be an instance of* Min-Uncut *and let k be a positive integer. There is a polynomial time algorithm that constructs an instance $\mathcal{I}'$ of* $\mathsf{Min\text{-}PGI}_{2k}$ *such that* $\mathsf{OPT}(\mathcal{I}') = (2k)^2\mathsf{OPT}(\mathcal{I})$.

The proof of this lemma is similar to the proof of Lemma 11. Given a set $E \subseteq \{x_i + x_j = 1 \mid i, j \in [n]\}$ of equations over $\mathbb{F}_2$, we construct an instance $\mathcal{I}'$ of $\mathsf{Min\text{-}PGI}_{2k}$ exactly as described in the proof of Lemma 11. If the minimum number of equations that have to be deleted from E to make the rest satisfiable is at most t, then there is an assignment such that at most t equations in E are not satisfied. This implies that there is a permutation π such that the only edges that are mapped to non-edges and vice-versa are from at most t pairs of color classes. The same argument as in the proof of Lemma 11 shows that for any permutation π there is a permutation σ such that $\overline{mp}(\sigma) \le \overline{mp}(\pi)$ and σ has the following property: For any color class j, σ maps all the vertices in V_j^0 to V_j^1 and vice-versa or is the identity mapping on that color class.

Finally we show that $\mathsf{Min\text{-}EGI}_4$ is hard to approximate.

Theorem 16. *For any constant $\alpha > 1$, there is no α-approximation algorithm for* $\mathsf{Min\text{-}EGI}_4$ *unless* $\mathsf{P} = \mathsf{NP}$.

An instance of NCP consists of a subspace $\mathcal{S}$ of $\mathbb{F}_2^n$ given as a set of basis vectors $\mathcal{B} = \{s_1, \ldots, s_k\}$ and a vector $v \in \mathbb{F}_2^n$. The objective is to find a vector $u \in \mathcal{S}$ which minimizes the hamming weight $\mathrm{wt}(u + v)$, i.e., the number of bits where u and v differ. It is NP-hard to approximate NCP within any constant factor [1]. The following lemma gives a reduction that transfers this hardness to $\mathsf{Min\text{-}EGI}_4$.

Lemma 17. *There is a polynomial time algorithm that for a given* NCP *instance $\mathcal{I}$, constructs a* $\mathsf{Min\text{-}EGI}_4$ *instance $\mathcal{I}'$ with* $\mathsf{OPT}(\mathcal{I}') = \mathsf{OPT}(\mathcal{I})$.

The idea of the proof is to construct two graphs G_1 and G_2 such that any vector from the given subspace $\mathcal{S}$ that is equal to v in all but k positions, can be converted into a color-preserving bijection from $V(G_1)$ to $V(G_2)$ that maps all but k edges to edges, and vice versa.

Let the instance $\mathcal{I}$ be given by the vector $v \in \mathbb{F}_2^n$ and the basis $\mathcal{B} = \{s_1, \ldots, s_m\}$ of the subspace $\mathcal{S}$, i.e., $\mathcal{S} = \{\sum_{i=1}^m \alpha_i s_i \mid \alpha_i \in \{0,1\}\}$. The computation of a vector $u \in \mathcal{S}$ can be thought of as n circuits $\mathcal{C}_1, \ldots, \mathcal{C}_n$. Thus $\mathcal{C}_i$ computes the i^{th} bit of u, i.e., $\mathcal{C}_i(\alpha) = \bigoplus_{j \in [m], s_{j,i}=1} \alpha_j$, where $\alpha = \alpha_1 \cdots \alpha_m$ is the input and $s_{j,i}$ is the i^{th} bit of s_j. We assume that these circuits contain only parity gates with fanin 2.

We now proceed to construct a graph G from these circuits such that there is a one-one correspondence between all assignments of values to α and all automorphisms of G. For each input bit α_j, add two vertices $\alpha_{j,0}, \alpha_{j,1}$ of the same color. Assigning $\alpha_j = 0$ corresponds to the identity permutation on this color class, assigning $\alpha_j = 1$ corresponds to exchanging these vertices. We also add two vertices of the same color for the output of each parity gate. To get the desired correspondence between assignments and automorphisms, we use the graph gadget of Torán [22]: For a parity gate with inputs x and y which computes $z = x \oplus y$, the gadget $G_\oplus$ connects the vertices x_0, x_1 corresponding to x, y_0, y_1 corresponding to y, and z_0, z_1 corresponding to z using four additional intermediate vertices $w_{0,0}, w_{0,1}, w_{1,0}, w_{1,1}$ that receive the same (new) color. For $b \in \{0,1\}$, the vertex x_b is connected to $w_{b,0}$ and $w_{b,1}$, while y_b is connected to $w_{0,b}$ and $w_{1,b}$. The vertex w_{b_1,b_2} is connected to $z_{b_1 \oplus b_2}$ for $b_1, b_2 \in \{0,1\}$. The construction is depicted in Figure 2.

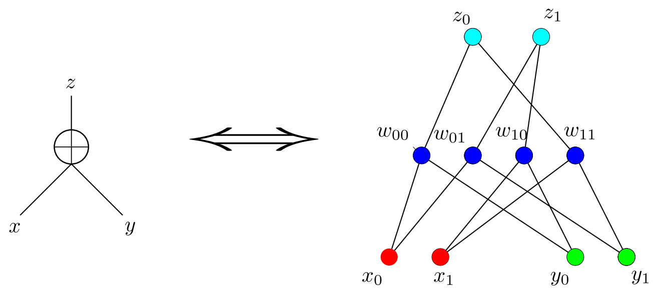

Fig. 2. Gadget $G_\oplus$ corresponding to a parity gate $z = x \oplus y$ [22]

The gadget is useful due to the following lemma.

Lemma 18 ([22]). *There is a unique automorphism ϕ for $G_\oplus$ which maps x_i to $x_{a\oplus i}$ and y_i to $y_{b\oplus i}$ for $a, b, i \in \{0,1\}$. This automorphism ϕ maps z_i to $z_{a\oplus b\oplus i}$.*

Lemma 18 implies that the automorphisms of G exactly correspond to the valid computations of the circuits $\mathcal{C}_1, \ldots, \mathcal{C}_n$ on all possible 2^m assignments. We obtain the two graphs G_1 and G_2 for the $\mathsf{Min\text{-}EGI}_4$ instance $\mathcal{I}'$ from the graph G by adding marker gadgets to the vertices corresponding to the output bits. Let $u_{i,0}$ and $u_{i,1}$ be the vertices corresponding to the output bit of $\mathcal{C}_i$. For each circuit, we add a new vertex u'_i (with a new color) in G_1 as well as in G_2. In G_1, we connect u'_i to $u_{i,0}$ if $v_i = 0$, and to $u_{i,1}$ otherwise, whereas in G_2, we connect u'_i to $u_{i,0}$ unconditionally. Now we are ready to prove Lemma 17.

Proof of Lemma 17. Given an instance of NCP specified by a subspace $\mathcal{S}$ generated by the basis vectors $\mathcal{B} = \{s_1, \ldots, s_m\}$ and a vector $v \in \mathbb{F}_2^n$, we construct graphs G_1 and G_2 as described above.

Suppose there exists a vector $u = \sum_{i=1}^m \alpha_i s_i$ such that $\mathrm{wt}(u + v) \leq t$. Given this α, we construct an automorphism π_α of G as follows: For each input node of $\mathcal{C}_i$, apply the automorphism on the vertices corresponding to the value of α_i to it. For each parity gate, Lemma 18 specifies how to extend an automorphism to the output vertices of the gadget, given a permutation of the input vertices. Continuing this process for the whole graph we get an automorphism of G that maps the vertex $u_{i,0}$ to u_{i,u_i}. We extend this automorphism to a mapping from G_1 to G_2, fixing the output marker vertices u'_i. The only unmatched edges are those incident to the vertices u'_i with $u_i \neq v_i$, so all but at most t edges of G_1 are mapped to edges of G_2.

Now suppose that there is a permutation π such that $\overline{me}(\pi) \leq t$ between the graphs G_1 and G_2. By construction, each parity gate is used for only one output bit, so at most t output bits are affected by the mismatched edges. Thus we can convert this permutation π to a new permutation σ such that $\overline{me}(\sigma) \leq \overline{me}(\pi)$ where the only edge that is mapped to a non-edge is $(u_{i,b}, u'_i)$. This is because for each circuit $\mathcal{C}_j$, starting from a permutation of its inputs, we can consistently extend the permutation till the output gate of $\mathcal{C}_j$. Thus depending on whether the input vertices were flipped by the permutation or not, we can assign a value to each α_j and hence get a vector $u \in \mathcal{S}$ such that $\mathrm{wt}(u+v) \leq t$. This completes the proof of the lemma and finishes the proof of Theorem 16. ☐

4 Conclusion

Although GI expressed as an optimization problem was mentioned in [2], as far as we know this is the first time that the complexity of the other three variants of this optimization problem has been studied. Considering the upper and lower complexity bounds that we have proved in this paper, the following questions seem particularly interesting.

In Theorem 1 we describe an α-approximation algorithm for Max-PGI that runs in quasi-polynomial time. Does Max-PGI also have a polynomial time approximation scheme? Theorem 2 shows that it is unlikely that Max-EGI has an $(\frac{1}{2}+\varepsilon)$-approximation algorithm. Does Max-EGI have a constant factor approximation algorithm? We can use the Quadratic Assignment Problem to get an additive error algorithm for it which runs in quasi-polynomial time but we do not know whether this algorithm can be used to get a constant factor approximation algorithm for Max-EGI (as was possible for Max-PGI). In the case of vertex-colored graphs, even though we can rule out the existence of a PTAS for Max-PGI$_k$ and Max-EGI$_k$, it remains open whether these problems have efficient approximation algorithms providing a good constant factor approximation guarantee.

Acknowledgement. We thank the anonymous referees for their suggestions to improve the article.

References

1. Sanjeev Arora, László Babai, Jacques Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. Syst. Sci*, 54(2):317–331, 1997.
2. Sanjeev Arora, Alan M. Frieze, and Haim Kaplan. A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. *Math. Program.*, 92(1):1–36, 2002.
3. László Babai. Monte-Carlo algorithms in graph isomorphism testing. Technical Report 79-10, Univ. de Montréal, Dép. de mathématiques et de statistique, 1979.

4. László Babai, D. Yu. Grigoryev, and David M. Mount. Isomorphism of graphs with bounded eigenvalue multiplicity. In *STOC*, pages 310–324, 1982.
5. László Babai and Eugene M. Luks. Canonical labeling of graphs. In *STOC*, pages 171–183, 1983.
6. Alberto Caprara and Romeo Rizzi. Packing triangles in bounded degree graphs. *Inf. Process. Lett.*, 84(4):175–180, November 2002.
7. Donatello Conte, Pasquale Foggia, Carlo Sansone, and Mario Vento. Thirty years of graph matching in pattern recognition. *IJPRAI*, 18(3):265–298, 2004.
8. Merrick L. Furst, John E. Hopcroft, and Eugene M. Luks. Polynomial-time algorithms for permutation groups. In *FOCS*, pages 36–41, 1980.
9. Xinbo Gao, Bing Xiao, Dacheng Tao, and Xuelong Li. A survey of graph edit distance. *Pattern Anal. Appl.*, 13(1):113–129, 2010.
10. Venkatesan Guruswami and Piotr Indyk. Embeddings and non-approximability of geometric problems. In *SODA*, pages 537–538, 2003.
11. Venkatesan Guruswami and Prasad Raghavendra. Constraint satisfaction over a non-boolean domain: Approximation algorithms and unique-games hardness. In *APPROX-RANDOM*, pages 77–90, 2008.
12. Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.
13. Sanjeev Khanna, Madhu Sudan, Luca Trevisan, and David P. Williamson. The approximability of constraint satisfaction problems. *SIAM J. Comput.*, 30(6):1863–1920, 2000.
14. Subhash Khot. On the power of unique 2-prover 1-round games. In *STOC*, pages 767–775, 2002.
15. Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? In *FOCS*, pages 146–154, 2004.
16. Michael Langberg, Yuval Rabani, and Chaitanya Swamy. Approximation algorithms for graph homomorphism problems. In *APPROX-RANDOM*, pages 176–187, 2006.
17. Michael Lewin, Dror Livnat, and Uri Zwick. Improved rounding techniques for the max 2-sat and max di-cut problems. In *IPCO*, pages 67–82, 2002.
18. Eugene M. Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. *J. Comput. Syst. Sci.*, 25(1):42–65, 1982.
19. Eugene M. Luks. Parallel algorithms for permutation groups and graph isomorphism. In *FOCS*, pages 292–302, 1986.
20. Gary L. Miller. Isomorphism of k-contractible graphs. a generalization of bounded valence and bounded genus. *Information and Control*, 56(1/2):1–20, 1983.
21. Erez Petrank. The hardness of approximation: Gap location. *Computational Complexity*, 4:133–157, 1994.
22. Jacobo Torán. On the hardness of graph isomorphism. *SIAM J. Comput*, 33(5):1093–1108, 2004.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de