On generic NP-completeness of the graph clustering problem

Alexander Rybalov
Sobolev Institute of Mathematics, Pevtsova 13, Omsk 644099, Russia
Omsk State Technical University, Mira ave., 11, Omsk 644050, Russia

E-mail: alexander.rybalov@gmail.com

Abstract. Generic-case approach to algorithmic problems was suggested by Miasnikov, Kapovich, Schupp and Shpilrain in 2003. This approach studies behavior of an algorithm on typical (almost all) inputs and ignores the rest of inputs. In this paper, we study the generic complexity of the problem of clustering graphs. In this problem the structure of relations of objects is presented as a graph: vertices correspond to objects, and edges connect similar objects. It is required to divide a set of objects into disjoint groups (clusters) to minimize the number of connections between clusters and the number of missing links within clusters. We prove that the graph clustering problem is NP-hard with respect to generic analog of polynomial Turing reduction. Supported by Russian Science Foundation, grant 18-71-10028.

1. Introduction

Kapovich, Miasnikov, Schupp and Shpilrain introdused [4] the genric approach to algorithmic problems in algebra. Within this approach, the algorithmic problem is not considered on the whole set of inputs, but on some subset of almost all inputs. Such inputs form the so-called generic set. The concept of almost all can be formalized by the introduction of a natural measure on the set of input data. In terms of practice, the algorithms that solve fast the problem on the generic set is as good as the fast algorithms for all inputs. A classic example of such an algorithm is Simplex method – it solves the linear problem in polynomial time programming for most input but has an exponential complexity at worst. Moreover, it may be that the problem intractable or generally undecidable in the classical sense, but easily resolvable on generic set. Note that a similar approach to studying problems of optimization was proposed earlier by Gimadi, Glebov and Perepelitsa [3].

One of the important problems of machine learning is the problem of graph clustering. In this problem, the structure of the relationship of objects is defined using the graph vertices of which correspond to objects, and edges connect similar objects. It is required to divide a set of objects into pairwise disjoint groups (clusters), so to minimize the number of links between clusters and the number of missing links inside clusters. In the works of Krivanek and Moravec [5], Bansala, Blum and Chaul [2], Shamir, Sharan and Tzur [8], Ageev, Iliev, Iliev, Iljeva, Kononov and Talevnin [1, 6, 7, 9] the NP-hardness of this problems for various formulations was proved. Thus, under the condition of P $\neq$ NP, there is no polynomial algorithm to solve this problem.

[1] Supported by Russian Science Foundation, grant 18-71-10028.

This article is devoted to the study of the generic complexity of the problem of graph clustering. We prove that the graph clustering problem is NP-hard with respect to generic analog of polynomial Turing reduction.

2. Generic algorithms

Let I be a set of inputs and I_n be the set of all inputs of size n. For a subset $S \subseteq I$ we define the following sequence

$$\rho_n(S) = \frac{|S_n|}{|I_n|},\ n = 1, 2, 3, \ldots,$$

where $S_n = S \cap I_n$ is the set of inputs from S of size n. *Asymptotic density* of S is the following limit

$$\rho(S) = \overline{\lim_{n\to\infty}} \rho_n(S).$$

A set S is called *negligible*, if $\rho(S) = 0$, and *strongly negligible*, if sequence $\rho_n(S)$ converges to 0 exponentialy fast i.e. there are constants σ, $0 < \sigma < 1$, and $C > 0$ such that for every n it holds $\rho_n(S) < C\sigma^n$.

Algorithm $\mathcal{A} : I \to J \cup \{?\}$ $(? \notin J)$ is called *(strongly) generic*, if

(i) $\mathcal{A}$ halts on all inputs from I;

(ii) set $\{x \in I : \mathcal{A}(x) = ?\}$ is (strongly) negligible.

Generic algorithm $\mathcal{A}$ computes a function $f : I \to J$, if for all $x \in I$

$$(\mathcal{A}(x) = y \in J) \Rightarrow (f(x) = y).$$

The equation $\mathcal{A}(x) = ?$ means that algorithm $\mathcal{A}$ can not compute function f on x. But the condition 2 guarantees that $\mathcal{A}$ correctly calculates f on almost all inputes.

There is a significant difference between generically decidable problems and strongly generically decidable problems. Suppose the problem S is decidable on some polynomially decidable generic set G such that

$$\frac{|G \cap I_n|}{|I_n|} = \frac{n-1}{n}$$

for any n. Thus G is generic but not strongly generic set. Now although the problem S is decidable for almost all inputs, nevertheless, there is an efficient method to get ¡¡bad¿¿ inputs, on which the generic algorithm does not work. A polynomial algorithm for generation of bad inputs is the following.

(i) To generate randomly and uniformly an input x of size n.

(ii) If $x \in G$, repeat step 1, otherwise finish.

Indeed, the probability of getting only good inputs for all n^2 rounds is

$$\left(\frac{n-1}{n}\right)^{n^2} = \left(\left(1 - \frac{1}{n}\right)^n\right)^n \to e^{-n}.$$

Therefore, with a probability very close to 1, a bad input will be obtained. On the other hand, it is easy to see that if the problem is solvable on a strongly generic set, such simple generation algorithm will require exponential number of rounds and will be ineffective. For cryptographic applications, this means that just generic decidability of a problem does not make this problem worthless for creation of a cryptosystem based on it, since for it exists efficient procedure for generating difficult inputs. At the same time, strongly generically easily solvable problems in this sense are useless for cryptography.

A problem A is *generically polynomial Turing reducible* to a problem B if there is a polynomial probabilistic algorithm $\mathcal{A}$ with call of any strongly generic algorithm deciding B as a subprogram, which is an algorithm for the problem A. We will denote this as $A \leq_{gpt} B$. A problem A is *generically NP-hard*, if $S \leq_{gpt} A$ for every problem $S \in NP$.

3. Graph clustering problem

Hereinafter, we will consider non-oriented graphs without loops and multiple edges. A graph is called *cluster* if each of its connected components is complete graph. Denote by $\mathcal{M}(V)$ the set of all cluster graphs on vertices V. If $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ are graphs on vertices V, then *the distance* $\rho(G_1, G_2)$ between them are the number of mismatched edges in graphs G_1 and G_2, i.e

$$\rho(G_1, G_2) = |E_1 \triangle E_2| = |E_1 \setminus E_2| + |E_2 \setminus E_1|.$$

The problem of graph clustering is the following. We have a $G = (V, E)$. Find a graph $M^* \in \mathcal{M}(V)$ such that

$$\rho(G, M^*) = \min_{M \in \mathcal{M}(V)} \rho(G, M).$$

Lemma 1. *Let G_1 and G_2 be graphs with disjoint sets of vertices and M^* is a cluster graph, which is a solution of the problem of graph clustering for graph $G_1 \cup G_2$. Then*

$$M^* = M_1^* \cup M_2^*,$$

where M_i^* is a solution of the problem of graph clustering for graph G_i, $i = 1, 2$.

Proof. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Suppose, there is a cluster graph M, which is a solution of the problem of graph clustering for graph $G_1 \cup G_2$ such that

$$M = C_1 \cup C_2 \cup \ldots \cup C_m,$$

where C_i, $i = 1, \ldots, m$ – disjoint complete connected components, and among them there is a component $C_k = K(V_c)$, which has non-empty intersection with graph G_1 and with graph G_2. Replace in cluster graph M its component C_k by two complete graphs

$$C_{k,i} = K(V_c \cap V_i), \ i = 1, 2.$$

Vertices of the first graph belong to graph G_1, and the second to G_2. Denote by M' this new cluster graph. Now note that

$$\rho(G_1 \cup G_2, M') < \rho(G_1 \cup G_2, M)$$

since in the new components $C_{k,i}$, $i = 1, 2$ there are all edges of old component C_k, for which or both vertices belong to G_1, either both vertices belong to G_2, and there are no all edges of old component C_k, for which one vertex belongs to G_1 and second – to G_2. And the last edges are not in the graph $G_1 \cup G_2$.

The resulting contradiction shows, that for the cluster graph M^*, which is a solution of the problem of graph clustering for graph $G_1 \cup G_2$, we have $M^* = M_1^* \cup M_2^*$, and M_i^* is a solution of the problem of graph clustering for graph G_i, $i = 1, 2$, so how else we can replace cluster graph M_i^* by better graph, thereby reducing the distance $\rho(G_1 \cup G_2, M^*)$. □

4. Main result

To study the generic complexity of the problem of graph clustering, we will use graph representation using adjacency matrices. Moreover, since the graphs are undirected, for encoding a graph with n vertices, we will use only the upper part of such a matrix, consisting of $n(n-1)/2$ bits. Thus, we will assume that the size of a graph with n vertices is equal to $n(n-1)/2$.

Theorem 1. *The problem of graph clustering is generically NP-hard.*

Proof. Let S be an arbitrary problem in the class NP. Since the problem of graph clustering is NP-hard in the classical sence, there is a classical polynomial reduction f, which for any input x of S gets an input $f(x)$ (a graph) for the graph clustering problem. Let $\mathcal{A}$ be a strongly generic algorithm deciding the problem of graph clustering. A polynomial probabilistic algorithm $\mathcal{B}$ with call of $\mathcal{A}$ as a subprogram, which will be an algorithm for the problem S works on input x in the following way.

(i) Computes $f(x) = G$ – a graph with n vertices (of size $n(n-1)/2$).
(ii) Generates a random graph H with $n^2 - n$ vertices.
(iii) Run algorithm $\mathcal{A}$ on graph $G \cup H$.
(iv) If $\mathcal{A}(G \cup H) \neq ?$, then by Lemma 1 clustering of $G \cup H$ gives a clustering of graph G and right solution for the problem S.
(v) If $\mathcal{A}(G \cup H) = ?$, then outputs K_n.

Note that polynomial probabilistic algorithm $\mathcal{B}$ outputs correct answer on the step 3, but incorrect answer on step 4. We need to prove that the probability of answer on step 4 is less than $1/2$.

Graph $G \cup H$ has n^2 vertices, so its size is $m = (n^4 - n^2)/2$. The probability that for a random graph $G \cup H$ it holds $\mathcal{A}(G \cup H) = ?$ is not greater than

$$\frac{|\{G \in \mathcal{G} : \mathcal{A}(G) \neq ?\}_m|}{|\{G \cup H : H \in \mathcal{G}\}_m|} = \frac{|\{G \in \mathcal{G} : \mathcal{A}(G) \neq ?\}_m|}{|\mathcal{G}_m|} \times \frac{|\mathcal{G}_m|}{|\{G \cup H : H \in \mathcal{G}\}_m|}.$$

Since the set $\{G \in \mathcal{G} : \mathcal{A}(G) \neq ?\}$ is strongly negligible, then there is a constant $\alpha > 0$ such that

$$\frac{|\{G \in \mathcal{G} : \mathcal{A}(G) \neq ?\}_m|}{|\mathcal{G}_m|} < \frac{1}{2^{\alpha m}} = \frac{1}{2^{\alpha(n^4-n^2)/2}}$$

for any n.

On the other hand, graph H has $n^2 - n$ vertices, thus

$$|\{G \cup H : H \in \mathcal{G}\}_m| = |\{H : H \in \mathcal{G}\}_{((n^2-n)^2-(n^2-n))/2}| = 2^{(n^4-2n^3+n)/2}.$$

Hence

$$\frac{|\mathcal{G}_m|}{|\{G \cup H : H \in \mathcal{G}\}_m|} = \frac{2^{(n^4-n^2)/2}}{2^{(n^4-2n^3+n)/2}} = 2^{(2n^3-n^2+n)/2}.$$

Therefore, the need probability is not greater

$$\frac{2^{(2n^3-n^2+n)/2}}{2^{\alpha(n^4-n^2)/2}} < \frac{1}{2}$$

for large enough n. □

[1] Ageev A A and Il'ev V P and Kononov A V and Talevnin A S 2007 Computational complexity of the graph approximation problem *Journal of Applied and Industrial Mathematics* **1 (1)** pp 1–8

[2] Bansal N and Blum A and Chawla S 2004 Correlation clustering *Machine Learning* **56** pp 89–113
[3] Gimadi E H and Glebov N I and Perepelitsa V A 1975 Algoritmy s ocenkami dlya zadach diskretnoi optimizacii *Problemy kibernetiki* **31** pp 35–42
[4] Kapovich I and Myasnikov A and Schupp P and Shpilrain V 2003 Generic-case complexity, decision problems in group theory and random walks *Journal of Algebra* **264(2)** pp 665–694
[5] Kŕivanek M and Morávek J 1986 NP-hard problems in hierarchical-tree clustering *Acta informatica* **23** pp 311–323
[6] Il'ev V P and Il'eva S D 2016 O zadachah klasterizacii grafov *Vestnik Omskogo Universiteta* **2** pp 16–18
[7] Il'ev A V and Il'ev V P 2018 Ob odnoi zadache klasterizacii grafa s chastichnym obucheniem *Prikladnaya Diskretnaya Matematika* **42** pp 66–75
[8] Shamir R and Sharan R and Tsur D 2004 Cluster graph modification problems *Discrete Applied Mathematics* **144 (1-2)** pp 173–182
[9] Talevnin A S 2004 O slozhnosti zadachi approksimacii grafov *Vestnik Omskogo Universiteta* **4** pp 22–24