A new algebraic approach to the graph isomorphism and clique problems

Roman Galay

The secondary school № 8, Nizhny Novgorod

(10th grade student)

As it follows from Gödel's incompleteness theorems, any consistent formal system of axioms and rules of inference should imply a true unprovable statement. Actually this fundamental principle can be efficiently applicable in Computational Mathematics and Complexity Theory concerning the computational complexity of problems from the class NP, particularly and especially the NP-complete ones. While there is a wide set of algorithms for these problems that we call heuristic, the correctness or/and complexity of each concrete algorithm (or the probability of its correct and polynomial-time work) on a class of instances is often too difficult to determine, although we may also assume the existence of a variety of algorithms for NP-complete problems that are both correct and polynomial-time on all the instances from a given class (where the given problem remains NP-complete), but whose correctness or/and polynomial-time complexity on the class is impossible to prove as an example for Gödel's theorems. However, supposedly such algorithms should possess a certain complicatedness of processing the input data and treat it in a certain algebraically "entangled" manner. The same algorithmic analysis in fact concerns all the other significant problems and subclasses of NP, such as the graph isomorphism problem and its associated complexity class GI.

The following short article offers a couple of algebraically entangled polynomial-time algorithms for the graph isomorphism and clique problems whose correctness is yet to be determined either empirically or through attempting to find proofs.

An heuristic polynomial-time algorithm for the graph isomorphism problem.

For a real-valued matrix B, by Sp(B) let's denote the set of values its entries are equal to, and by MSp(B) the multi-set of those values (including the multiplicity of each value). We'll call those set and multi-set the *entry spectrum* and the *entry multi-spectrum* of B correspondingly.

Given two simple undirected graphs G_1 and G_2 with n vertices whose adjacency matrices are A_1 and A_2 correspondingly, we're going to build two sequences of real-valued matrices $A_1^{(i)}$ and $A_2^{(i)}$ by the following recursive scheme:

$$A_1^{(0)} = A_1, \qquad A_2^{(0)} = A_2$$

Beginning with i = 1, at the step i of our recursive process, first of all we determine whether $\mathrm{MSp}(A_1^{(i)}) = \mathrm{MSp}\left(A_2^{(i)}\right)$.

If it's not so then G_1 and G_2 are definitely non-isomorphic. If, otherwise, the equality holds then we create a random $|\mathrm{Sp}(A_1^{(i)})|$-vector y and in both matrices $A_1^{(i)}$ and $A_2^{(i)}$ we replace each entry whose value is the j-th element of $\mathrm{Sp}(A_1^{(i)})$ by y_j . After that, we choose a random polynomial p(t) of degree n-1 and calculate $A_1^{(i+1)} = p(A_1^{(i)})$ and $A_2^{(i+1)} = p(A_2^{(i)})$.

We stop the whole process when either the entry multi-spectrums of the two current matrices are different or the cardinality of their common entry spectrum doesn't increase any more from one step to another, i.e. $|\mathrm{Sp}(A_1^{(k)})| = |\mathrm{Sp}(A_1^{(k-1)})|$ (accordingly, the overall number k of performed steps can't exceed n^2).

We hence declare the initial graphs' non-isomorphism in the former case, and their isomorphism in the latter one.

When the process is stopped, with the overall number of steps equal to k and the final entry spectrum $\{\alpha_1, \dots, \alpha_m\}$, there should exist symmetric 0,1-matrices $H_1^{(1)}, \dots, H_1^{(m)}, H_2^{(1)}, \dots, H_2^{(m)}$ such that

$$A_1^{(k)} = \sum_{u=1}^{m} \alpha_u H_1^{(u)}, \qquad A_2^{(k)} = \sum_{u=1}^{m} \alpha_u H_2^{(u)},$$

We can consider, for u = 1,..,m, $H_1^{(u)}$ and $H_2^{(u)}$ as the adjacency matrices of some graphs $G_1^{(u)}$ and $G_2^{(u)}$. In case if the initial graphs G_1 and G_2 are really isomorphic, those two graphs should be isomorphic as well. Moreover, due to the entry spectrum's non-growing at the end of the above-described process, for any pair $v, w \in \{1, \dots, m\}, v \neq w$, $G_1^{(v)}$ and $G_1^{(w)}$ should possess no common edges and $G_2^{(v)}$ and $G_2^{(w)}$ should too; also there should exist a sequence of coefficients $d_1^{(v,w)}, \dots, d_m^{(v,w)}$ such that $H_1^{(v)}H_1^{(w)} + H_1^{(w)}H_1^{(v)} = \sum_{u=1}^{m} d_u^{(v,w)} H_1^{(u)}$ and $H_2^{(v)}H_2^{(w)} + H_2^{(w)}H_2^{(v)} = \sum_{u=1}^{m} d_u^{(v,w)} H_2^{(u)}$ (the latter condition also concerns the case v = w). Verifying these additional relations is the proposed algorithm's final action when declaring the isomorphism of G_1 and G_2. Thus we get a pair of identical commutative algebras that are two linear spaces with the bases $H_1^{(1)}, \dots, H_1^{(m)}$ and $H_2^{(1)}, \dots, H_2^{(m)}$ whose elements are $X = \sum_{u=1}^{m} x_u H_1^{(u)}$ and $X = \sum_{u=1}^{m} x_u H_2^{(u)}$ correspondingly and whose algebra product is defined as $X * Y = XY + YX$.

The above-formulated algorithm can be naturally adjusted for digraphs with absolutely the same computational circuit (while, though strange, generating commutative algebras at the very end as well). Moreover, it can be generalized for an arbitrary field (instead of $\mathbb{R}$) with all the computations performed over that field. In case if the chosen field is of characteristic 2, we'll receive Lie algebras.

Another type of generalization this algorithm might be subjected to is as follows. Given an n×n-matrix B, let's define a product $B^{r_1}1_{n\times n}B^{r_2}1_{n\times n}\ldots B^{r_{s-1}}1_{n\times n}B^{r_s}$ (where $1_{n\times n}$ is an n×n-matrix all whose entries equal unity and $r_1,\ldots,r_s$ are non-negative integers unexceeding n-1) as its *meta-power* of *meta-degree* $(r_1,r_2,\ldots,r_s)$ and a linear combination of a set of its meta-powers with coefficients taken from a chosen field as a *meta-polynomial* in B over the field. If an n×n-matrix can be turned into another one via permuting its rows and columns by a permutation π (i.e. if we have a pair of isomorphic matrices) then an arbitrary meta-polynomial computed in both of them should give us a pair of isomorphic matrices as well (with the same transitional permutation π). Hence the idea of replacing, in the proposed algorithmic approach, random polynomials p(t) of degree n-1 by random meta-polynomials may look yet perspective, even though it's still difficult to figure out how the meta-polynomials' sets of utilized meta-degrees could be restricted in such a case. Nevertheless, the principle of stopping the recursive process upon getting either different entry multi-spectrums of the two current matrices or their common entry spectrum's cardinality ceasing to grow from one step to another remains intact (while the matrices $A_1^{(i)}$ and $A_2^{(i)}$ generically cease to be symmetric for i > 0 even in the case of undirected initial graphs). The final splittings of the two matrices $A_1^{(k)}$ and $A_2^{(k)}$ (where k is the overall number of performed steps) apparently will have nearly the same structure as in the case of random polynomials p(t), but with the only additional condition of $H_1^{(u)}1_{n\times n}$ and $1_{n\times n}H_1^{(u)}$ belonging to the first final algebra and $H_2^{(u)}1_{n\times n}$ and $1_{n\times n}H_2^{(u)}$ to the second one for u = 1,…,m. As, obviously, such a final algebra doesn't depend on the chosen random polynomials or, generally, meta-polynomials within the algorithmic circuit (as well as the random substitution vectors we use for entry spectrums' replacements), we'll call it the *splitting algebra* of a graph that is a system of its invariants. And accordingly we can formulate the **conjecture** that **the splitting algebra is a complete system of graph invariants**.

A polynomial-time heuristic approach

to the clique problem

The general types of techniques and notions that we applied when dealing with the graph isomorphism problem could also be applicable in resolving the much more important problems of determining a graph's clique number and finding its maximum clique, i.e. for the clique problem which is well-known to be NP-complete, unlike the graph isomorphism one.

Given a simple undirected graph G with n vertices whose adjacency matrix is A, we're going to build a sequence of real-valued n×n-matrices $X^{(q)} = X^{(q)}(G)$ (whose columns we'll consider as the coordinate-vectors of n points in $\mathbb{R}^n$, while the j-th point we'll associate with the vertex j of G) by the following recursive scheme:

$$X^{(0)} = I_n$$

Beginning with q = 1, at the step q of our recursive process, we define for j =1,…,n

(*) $\quad x_j^{(q+1)} = x_j^{(q)} + g\sum_{k\in\{1,\ldots,n\}\setminus\{j\}} \frac{\varepsilon a_{jk}}{\mathbf{d}^s(x_j^{(q)},x_k^{(q)})}(x_k^{(q)} - x_j^{(q)})$

where $x_j^{(q)}$ is the j-th column of $X^{(q)}$, h, s, ε are the proposed algorithm's parameters (h, s, ε > 0), and for two n-vectors y, z **d**(y,z) denotes the Euclidean distance between them in $\mathbb{R}^n$.

The above recursive relation is, in fact, a computational circuit for numerically solving the autonomous system of differential equations

$$\dot{x}_j(t) = g \sum_{k\in\{1,\dots,n\}\backslash\{j\}} \frac{a_{jk}}{\mathbf{d}^s\left(x_j(t), x_k(t)\right)} (x_k(t) - x_j(t))$$

and the chief idea standing behind it is the **conjecture** that **after a certain period of time since the beginning of the initial point system's contraction under the influence of the introduced "gravitational forces" between pairs of points connected, as vertices, by G's edges the smallest distance will always appear between a pair of vertices belonging to a maximum clique of G**.

A special interest this conjecture presents for the case of regular and semi-regular graphs, particularly for the graph generated by a CNF so that its vertex set is the CNF's set of literals and a pair of literals isn't to be connected by an edge if and only if either both of them belong to one disjunctive clause or they're the opposite powers of one variable (this graph's clique number equals the CNF's number of disjunctive clauses if and only if the CNF is satisfiable and is smaller otherwise, and the graph becomes close to regular upon bounding by constants the CNF's quantity of literals in a disjunctive clause and the number of times a variable can occur in literals, while it's regular when the two numbers are just constants, i.e. same for all the clauses and all the variables correspondingly). It's actually known that the clique problem is NP-complete for regular graphs, even when restricted to the case of graphs complimentary to cubic planar ones.

Hence such an algorithm is supposed to perform a certain polynomial (in n) number of steps (*) (considered as the algorithm's functional parameter), determine the pair of vertices (u,v) of the smallest distance, and construct the graph $G_{N_G(u,v)}$ received from G as the graph induced by the set $N_G(u,v)$ of common neighbors of u and vin G. After that the whole process is to be repeated for $G_{N_G(u,v)}$ etc. until we get either an empty graph or a graph with one vertex.

An additional idea for enhancing the above approach to the clique problem may be introducing "repelling forces" (aka "anti-gravitational") between pairs of points that aren't connected, as vertices, in G. In such a case we'll receive the equation

$$\dot{x}_j(t) = g \sum_{k\in\{1,\dots,n\}\backslash\{j\}} \frac{a_{jk}}{\mathbf{d}^s\left(x_j(t), x_k(t)\right)} (x_k(t) - x_j(t)) -$$

$$- g_1 \sum_{k\in\{1,\dots,n\}\backslash\{j\}} \frac{1 - a_{jk}}{\mathbf{d}^{s_1}\left(x_j(t), x_k(t)\right)} (x_k(t) - x_j(t))$$

with the corresponding computational circuit for numerical solving

(**) $x_j^{(q+1)} = x_j^{(q)} + g\sum_{k\in\{1,\dots,n\}\backslash\{j\}} \frac{\varepsilon a_{jk}}{\mathbf{d}^s(x_j^{(q)},x_k^{(q)})}(x_k^{(q)} - x_j^{(q)}) -$

$$- g_1 \sum_{k\in\{1,\dots,n\}\backslash\{j\}} \frac{\varepsilon(1-a_{jk})}{\mathbf{d}^{s_1}(x_j^{(q)},x_k^{(q)})}(x_k^{(q)} - x_j^{(q)})$$

Let's call g and g_1 the *gravitation* and *anti-gravitation coefficients* correspondingly.

Besides, we can notice that the matrices $X^{(q)}(G)$ are, of course, invariant under any of G's automorphisms and, given two graphs G_1 and G_2, we accordingly may also use $X^{(q)}(G_1)$ and $X^{(q)}(G_2)$ for determining if they're isomorphic (via comparing $\mathrm{MSp}(X^{(q)}(G_1))$ and $\mathrm{MSp}(X^{(q)}(G_2))$ and eventually obtaining a pair of final algebras upon their entry spectrums' ceasing to grow), but, nevertheless, in such a case we're actually supposed to figure out whether it's not a partial case of the initial graph's adjacency matrix's modification via a series of meta-polynomial and entry spectrum replacement transformations.

The proposed algorithm for the clique problem had been tested (via computer modeling) on graphs received (as described above) from random CNF samples with several hundred Boolean variables whose maximal number of literals in a clause and maximal number of a variable's occurrence was 3 and showed correctness and polynomial-time performance in finding, in case of the CNF's satisfiability, a satisfying Boolean vector.

And, at last, it would be worth noting that any kind of approach to the clique problem can be further enhanced with a randomization parameter via embedding the given graph G with n vertices whose clique number we need to determine into the graph received from G through *bipartitely gluing* it with a random graph G_1 with n_1 vertices whose clique number we know, -- while we define the *bipartite gluing* of two graphs $G = (V, E)$, $G_1 = (V_1, E_1)$ for disjoint V, V_1 as the graph $G \oslash G_1 = (V \cup V_1, E \cup E_1 \cup K_{V,V_1})$ where K_{V,V_1} is the complete bipartite graph on the parts V, V_1 . The clique number of $G \oslash G_1$ obviously equals the sum of the clique numbers of G and G_1 and this gluing is $(k + n_1)$-regular when G is k-regular, G_1 is k_1-regular and $n + k_1 = n_1 + k$. Hence, in case if our algorithm works out on a certain sufficiently large fraction of q-regular graphs with m vertices for a certain set of values of the ratio q:m (containing NP-complete cases), we can also conjecture that such a random gluing may be quite capable of resolving, via the proposed "gravitational" algorithm, the most hard cases of instances with a sufficiently high (for being a polynomial-time randomized computational circuit) probability of success. However, the general direction of research regarding the above-stated gravitation contraction model may, of course, be rather related to attempting to understand the behavior of its differential equation's solution and even trying, on the basis of such understanding, to reduce the algorithm's polynomial-time complexity through modifying its vertex selection criterion for to take, at each global step, not just one pair, but a much bigger set of vertices as supposedly belonging to a maximum clique.

References:

* Aho, Alfred V.; Hopcroft, John; Ullman, Jeffrey D. (1974), The Design and Analysis of Computer Algorithms, Reading, MA: Addison-Wesley.

* Arvind, Vikraman; Köbler, Johannes (2000), "Graph isomorphism is low for ZPP(NP) and other lowness results.", Proceedings of the 17th Annual Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Computer Science, 1770, Springer-Verlag, pp. 431–442, doi:10.1007/3-540-46541-3_36, ISBN 3-540-67141-2, MR 1781752.

* Arvind, Vikraman; Kurur, Piyush P. (2006), "Graph isomorphism is in SPP", Information and Computation, 204 (5): 835–852, doi:10.1016/j.ic.2006.02.002, MR 2226371.

* Babai, László (1980), "On the complexity of canonical labeling of strongly regular graphs", SIAM Journal on Computing, 9 (1): 212–216, doi:10.1137/0209018, MR 0557839.

* Babai, László; Codenotti, Paolo (2008), "Isomorphism of hypergraphs of low rank in moderately exponential time" (PDF), Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008), IEEE Computer Society, pp. 667–676, doi:10.1109/FOCS.2008.80, ISBN 978-0-7695-3436-7.

* Babai, László; Grigoryev, D. Yu.; Mount, David M. (1982), "Isomorphism of graphs with bounded eigenvalue multiplicity", Proceedings of the 14th Annual ACM Symposium on Theory of Computing, pp. 310–324, doi:10.1145/800070.802206, ISBN 0-89791-070-2.

* Babai, László; Kantor, William; Luks, Eugene (1983), "Computational complexity and the classification of finite simple groups", Proceedings of the 24th Annual Symposium on Foundations of Computer Science (FOCS), pp. 162–171, doi:10.1109/SFCS.1983.10.

* Babai, László; Luks, Eugene M. (1983), "Canonical labeling of graphs", Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing (STOC '83), pp. 171–183, doi:10.1145/800061.808746, ISBN 0-89791-099-0.

* Babai, László (2015), Graph Isomorphism in Quasipolynomial Time, arXiv:1512.03547, Bibcode:2015arXiv151203547B * Baird, H. S.; Cho, Y. E. (1975), "An artwork design verification system", Proceedings of the 12th Design Automation Conference (DAC '75), Piscataway, NJ, USA: IEEE Press, pp. 414–420.

* Blum, Manuel; Kannan, Sampath (1995), "Designing programs that check their work", Journal of the ACM, 42 (1): 269–291, doi:10.1145/200836.200880.

* Bodlaender, Hans (1990), "Polynomial algorithms for graph isomorphism and chromatic index on partial k-trees", Journal of Algorithms, 11 (4): 631–643, doi:10.1016/0196-6774(90)90013-5, MR 1079454. * Booth, Kellogg S.; Colbourn, C. J. (1977), Problems polynomially equivalent to graph isomorphism, Technical Report, CS-77-04, Computer Science Department, University of Waterloo.

* Booth, Kellogg S.; Lueker, George S. (1979), "A linear time algorithm for deciding interval graph isomorphism", Journal of the ACM, 26 (2): 183–195, doi:10.1145/322123.322125, MR 0528025.

* Boucher, C.; Loker, D. (2006), Graph isomorphism completeness for perfect graphs and subclasses of perfect graphs (PDF), Technical Report, CS-2006-32, Computer Science Department, University of Waterloo.

* Abello, J.; Pardalos, P. M.; Resende, M. G. C. (1999), "On maximum clique problems in very large graphs" (PDF), in Abello, J.; Vitter, J., External Memory Algorithms, DIMACS Series on Discrete Mathematics and Theoretical Computer Science, 50, American Mathematical Society, pp. 119–130, ISBN 0-8218-1184-3.

* Alon, N.; Boppana, R. (1987), "The monotone circuit complexity of boolean functions", Combinatorica, 7 (1): 1–22, doi:10.1007/BF02579196. * Alon, N.; Krivelevich, M.; Sudakov, B. (1998), "Finding a large hidden clique in a random graph", Random Structures & Algorithms, 13 (3–4): 457–466, doi:10.1002/(SICI)1098-2418(199810/12)13:3/4<457::AID-RSA14>3.0.CO;2-W.

* Alon, N.; Yuster, R.; Zwick, U. (1994), "Finding and counting given length cycles", Proceedings of the 2nd European Symposium on Algorithms, Utrecht, The Netherlands, pp. 354–364.

* Amano, Kazuyuki; Maruoka, Akira (2005), "A superpolynomial lower bound for a circuit computing the clique function with at most (1/6)log log N negation gates", SIAM Journal on Computing, 35 (1): 201–216, doi:10.1137/S0097539701396959, MR 2178806.

* Arora, Sanjeev; Lund, Carsten; Motwani, Rajeev; Sudan, Madhu; Szegedy, Mario (1998), "Proof verification and the hardness of approximation problems", Journal of the ACM, 45 (3): 501–555, doi:10.1145/278298.278306, ECCC TR98-008. Originally presented at the 1992 Symposium on Foundations of Computer Science, doi:10.1109/SFCS.1992.267823.

* Arora, S.; Safra, S. (1998), "Probabilistic checking of proofs: A new characterization of NP", Journal of the ACM, 45 (1): 70–122, doi:10.1145/273865.273901. Originally presented at the 1992 Symposium on Foundations of Computer Science, doi:10.1109/SFCS.1992.267824.

* Balas, E.; Yu, C. S. (1986), "Finding a maximum clique in an arbitrary graph", SIAM Journal on Computing, 15 (4): 1054–1068, doi:10.1137/0215075.

* Barrow, H.; Burstall, R. (1976), "Subgraph isomorphism, matching relational structures and maximal cliques", Information Processing Letters, 4 (4): 83–84, doi:10.1016/0020-0190(76)90049-1.

* Battiti, R.; Protasi, M. (2001), "Reactive local search for the maximum clique problem", Algorithmica, 29 (4): 610–637, doi:10.1007/s004530010074. * Bollobás, Béla (1976), "Complete subgraphs are elusive", Journal of Combinatorial Theory, Series B, 21 (1): 1–7, doi:10.1016/0095-8956(76)90021-6, ISSN 0095-8956.

* Boppana, R.; Halldórsson, M. M. (1992), "Approximating maximum independent sets by excluding subgraphs", BIT Numerical Mathematics, 32 (2): 180–196, doi:10.1007/BF01994876.

* Bron, C.; Kerbosch, J. (1973), "Algorithm 457: finding all cliques of an undirected graph", Communications of the ACM, 16 (9): 575–577, doi:10.1145/362342.362367.

* Carraghan, R.; Pardalos, P. M. (1990), "An exact algorithm for the maximum clique problem", Operations Research Letters, 9 (6): 375–382, doi:10.1016/0167-6377(90)90057-C.

* Cazals, F.; Karande, C. (2008), "A note on the problem of reporting maximal cliques" (PDF), Theoretical Computer Science, 407 (1): 564–568, doi:10.1016/j.tcs.2008.0