Hardness Results for Isomorphism and Automorphism of Bounded Valence Graphs

Fabian Wagner [⋆]

Institut für Theoretische Informatik,
Universität Ulm, 89073 Ulm, Germany
fabian.wagner@uni-ulm.de

Abstract. In a bounded valence graph every vertex has $O(1)$ neighbours. Testing isomorphism of bounded valence graphs is known to be in P [15], something that is not clear to hold for graph isomorphism in general. We show that testing isomorphism for undirected, directed and colored graphs of valence 2 is logspace complete. We also prove the following: If a special version of bounded valence GI is hard for $\mathrm{Mod_k L}$ then it is also hard for #L. All results are proved with respect to DLOGTIME uniform AC^0 many-one reductions.

1 Introduction

The graph isomorphism problem (GI) is to decide whether there is a bijection between the vertices of two graphs, preserving the edge relations. Whether GI is in P is a big open question. The NP-completeness for GI would cause a collapse of the polynomial time hierarchy to its second level, see [5],[20].

For some graph classes, efficient algorithms for testing isomorphism are known, e.g. for planar graphs [11] [18], planar three-connected graphs [21], trees [14] [7], graphs of bounded eigenvalue multiplicity [3], graphs with bounded color-class size [16] or trivalent graphs [15].

With respect to lower bounds, GI is hard for DET under AC^0 many-one reductions [22]. The same lower bounds hold for isomorphism testing of tournament graphs [23].

It is open whether GI restricted to bounded valence graphs is as hard as general GI [17], [10]. Tutte proved that the automorphism group of a trivalent graph which stabilizes some edge e is a 2-group (cf. [10]). Luks gives for bounded valence GI a polynomial time algorithm [15].

The motivation for studying the complexity status of bounded valence GI is that there is a huge gap between the best known upper and lower bounds. We prove that isomorphism for undirected, directed and colored valence-2 graphs is complete for logspace. A special version of prefix-GA is introduced. We say that the prefix-GA problem has the fixed vertex property if automorphisms of the given graph, which satisfy the prefixes, are required to map at least one vertex to itself. We also prove the following: If bounded valence prefix-GA with fixed

⋆ Supported by DFG grant TO 200/2-1

vertex property is hard for $\mathrm{Mod_kL}$ then it is also hard for #L. In this case, bounded valence-GI would be hard for #L.

2 Preliminaries

We assume familiarity with basic notions of complexity and graph theory such as can be found in standard textbooks.

Complexity classes. NL is nondeterministic logspace, #L is defined [2] as the class of functions $f : \Sigma^* \to \mathbb{N}$ that counts the number of accepting paths of a NL machine on a input. Based on #L functions the following classes are defined:

$\mathrm{PL} = \{A : \exists p \in Poly, f \in \#L,\ \ \forall x \in \Sigma^*\ x \in A \Leftrightarrow f(x) \geq 2^{p(|x|)}\}$ [9] [19]
$\mathrm{C_{=}L} = \{A : \exists p \in Poly, f \in \#L,\ \ \forall x \in \Sigma^*\ x \in A \Leftrightarrow f(x) = 2^{p(|x|)}\}$ [1]
$\mathrm{Mod_kL} = \{A : \exists f \in \#L,\ \ \forall x \in \Sigma^*\ x \in A \Leftrightarrow f(x) \equiv 1 \mod k\}$ [6]

$\mathrm{Mod_k}$ circuits ($k \geq 2$) are circuits with input variables over $\mathbb{Z}_k$ and gates computing addition in $\mathbb{Z}_k$. The evaluation problem for such circuits (given fixed values for the inputs, testing if the output is 1) is complete for $\mathrm{Mod_kL}$ under $\mathrm{AC^0}$ many-one reductions. We prove our hardness results for the DLOGTIME uniform $\mathrm{AC^0}$ many-one reducibility (in short $\mathrm{AC^0}$ reducibility). A set A is $\mathrm{AC^0}$ reducible to another set B if there is a family of circuits $\{C_n | n \in \mathbb{N}\}$ where each circuit C_n contains only AND, OR and NOT gates, has size $n^{\{O(1)\}}$ and depth $O(1)$, and for each x of length n $x \in A \Leftrightarrow C_n(x) \in B$.

Graph Isomorphism Problems. Let $G = (V, E)$ be a *graph* with a set of *vertices* $V = V(G)$ and *edges* $E = E(G)$. Let $G[X]$ be a subgraph of G *induced* on vertex set X. Let H be a subgraph of G then $G \backslash H = G[V(G) \backslash V(H)]$.

For shorter notations we write $[k, n] = \{k, \ldots, n\}$ for integers $k < n$ and $[n]$ if $k = 1$. The set $Sym(V)$ is the *symmetric group* over a set V and $S_n = Sym([n])$.

An *automorphism* of graph G is a permutation $\phi : V(G) \mapsto V(G)$ preserving adjacency: $(u, v) \in E(G) \Leftrightarrow (\phi(u), \phi(v)) \in E(G)$. A *rigid* graph contains no nontrivial automorphisms. The *graph automorphism problem* (GA) decides, whether a graph is rigid or not. The *automorphism group* $Aut(G)$ is the set of automorphisms of G. The *prefix automorphism problem* (prefix-GA) as denoted in [13] is given $x_1, \ldots, x_k, y_1, \ldots, y_k \in V(G)$, find an automorphism $\phi \in Aut(G)$ such that $\phi(x_i) = y_i\ \forall i \in [k]$.

An *isomorphism* between graphs G and H is a bijective mapping of vertices in G onto vertices in H that preserves adjacency. If G and H are isomorphic, we write $G \cong H$. The *graph isomorphism problem* (GI) is defined as

$$\mathrm{GI} = \{\, (G, H) \mid G \cong H \,\}.$$

Let k be a fixed integer. A *coloring* of a graph G is a function $f : V(G) \mapsto [k]$. For any isomorphism between colored graphs, the color relations have to be preserved. The decision problem is called the *isomorphism problem for colored graphs* (color-GI).

The *degree* or *valence* of a vertex v in a graph G is the number of edges incident to v. The *valence of a graph* is the maximum valence of its vertices. The *valence-k graph isomorphism problem* is the same as GI with the input-graphs restricted to valence-k graphs.

3 Complexity of Bounded Valence Graph Isomorphism

In this section we discuss upper and lower bounds for valence-k GI for different values of k.

3.1 Valence-2 Graph Isomorphism

We discuss the complexity of valence-2 GI and prove that Valence-2 GI of directed, undirected and colored graphs is complete for logspace.

Theorem 1. *Valence-2 GI is complete for* L *under* AC^0 *many-one reductions.*

Proof. For hardness we reduce the logspace complete problem ORD to valence-2 GI. Etessami proved that this problem is L-complete via quantifier-free projections. Recall the following definition.

Order between Vertices (ORD) [8], [12]
Given: A digraph $G = (V, E)$ with $|V| = n$ that is a line and two designated nodes $v_i, v_j \in V(G)$.
Problem: Decide whether $v_i < v_j$ in the total order induced on $V(G)$.

For the reduction we construct two graphs X, Y such that $X \cong Y$ if and only if $v_i < v_j$ in the order induced by G.

- Construct two new graphs X and Y, also see Figure 1.
- X is the undirected version of G (replace arcs by undirected edges) but with the following change: $\{v_j, v_{j+1}\} \notin E(X)$.
- Y is the undirected version of G but with the following changes: $\{v_i, v_{i+1}\}$, $\{v_j, v_{j+1}\} \notin E(Y)$, but $\{v_j, 1\} \in E(Y)$.
- If $i < j$ then $Y = Y_1$ is isomorphic to X. But if $j < i$ then $Y = Y_2$ contains a cycle and is not isomorphic to X.

This reduction is an AC^0 many-one reduction. Since ORD is L-complete via quantifier-free projections, valence-2 GI is hard for L under AC^0 many-one reductions.

For completeness we give a logspace Turing machine which descides isomorphism between valence-2 graphs. Let G, H be two valence-2 graphs. The machine works as follows:

- Count and compare the number of isolated vertices in G and H.
- For each value $k \in [n]$ compare the number of paths of length k in G and H.

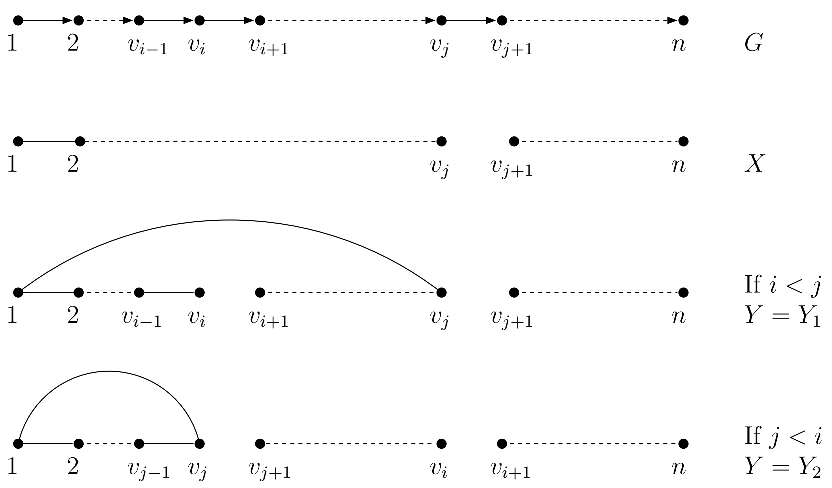

Fig. 1. Graph gadgets for simulation of ORD.

- For each value $k \in [n]$ compare the number of cycles of length k in G and H. Observe that the length of paths and cycles can be computed in logspace.
- Space requirements: We need counters for the variable k and to compare the number of counted isomorphic components.
- As we run through all isolated vertices, paths and cycles we do not have to remember explicitely which components have been visited yet. Instead, we run through all vertices sorted by their labels, call this vertex *actual*. Thereby, we just consider the components where the actual vertex has the minimum label. Observe, that testing the minimum property for each component can be done in logspace. Thus, we reach every component and consider it exactly once.

□

3.2 Coloring and Bounded Valence

For general graphs there is no difference between the complexity of GI and the complexity of color-GI. That is, instead of coloring a vertex v, a special tree is connected to v. This tree has the property that different colors correspond to nonisomorphic trees (cf. [13]). In this section we will prove that this also holds for bounded valence GI.

First we consider the complexity of colored and directed valence-2 graphs. The situation here is different since the standard techniques can not be adapted for valence less than three. With minor changes to the standard techniques we also prove that valence-k color-GI is reducible to valence-k GI for $k \geq 3$ under AC^0 many-one reductions.

Theorem 2. *Valence-2 color-GI is complete for* L.

Proof. Hardness for L follows since valence-2 GI without coloring is hard for L.

For completeness we give a logspace Turing machine which descides isomorphism between colored valence-2 graphs. Let G, H be two such graphs.

- For each color c count and compare the number of isolated vertices with color c in G and H.
- For each value $k \in [n]$ compare the number of paths of length k with the same sequence of colors in G and H. The length of the paths can be tested in logspace. Let P_1, P_2 be two paths of length k. Find one end v in P_1 and both ends w, w' in P_2. Start with v and test whether $\mathit{color}(v) = \mathit{color}(w)$. If it is so then compare the color of the neighbours at distance i for all $i \in [n]$ until we reach the other end of the paths. If we distinguished the paths then do the same with vertices v and w', that is we run through the reverse of P_2. If we cannot distinguish the sequences of colors in one of both tests then both colored paths are isomorphic.
- For each value $k \in [n]$ compare the number of cycles of length k with the same sequence of colors in G and H. The length of the cycles can be tested in logspace. Let C_1, C_2 be two cycles of length k. Start with the actual vertex v in C_1 and let $v', v'' \in \Gamma(v)$ with v' the neighbour with minimum label. For each vertex w in C_2 and each neighbour $w', w'' \in \Gamma(w)$ do the following. Start with $v, v', \ldots, v'', v$ and $w, w', \ldots, w'', w$ and pairwise compare the colors of the corresponding vertices v with w and v' with w' and so on. If we distinguish the sequence of colors then do the same with $v, v', \ldots, v'', v$ and $w, w'', \ldots, w', w$. If we cannot distinguish them then both cycles are isomorphic.
- Space requirements and the technique to traverse the components in both graphs are similar as in proof of Theorem 1.

□

One consequence of Theorem 2 is that considering directed graphs keeps the complexity of valence-2 GI unchanged.

Corollary 1. *Valence-2 GI* $\leq_{\mathrm{m}}^{\mathrm{AC}^0}$ *Valence-2 directed GI* $\leq_{\mathrm{m}}^{\mathrm{AC}^0}$ *Valence-2 color-GI.*

Proof. First, Valence-2 GI $\leq_{\mathrm{m}}^{\mathrm{AC}^0}$ valence-2 directed GI. Let $\{u, v\}$ be a undirected edge. Replace it and insert a vertex w in between and arcs $(u, w), (v, w)$. Second, valence-2 directed GI $\leq_{\mathrm{m}}^{\mathrm{AC}^0}$ Valence-2 color-GI. Let (u, v) be a directed edge. Replace it by a path of length three, that is $\{u, w_1\}$,$\{w_1, w_2\}$,$\{w_2, v\}$. Apply special colors c_1 to w_1 and c_2 to w_2. □

We have shown that the following problems are complete for L under AC^0 many-one recudibility: Valence-2 undirected GI, valence-2 directed GI, valence-2 color GI. Concerning graphs of higher valence, standard techniques as in [13] suffice to proof valence-k directed GI $\equiv_{\mathrm{m}}^{\mathrm{AC}^0}$ valence-k undirected GI for $k \geq 3$. Now, we discuss coloring for graphs of higher valence.

Lemma 1. *Valence-k color-GI $\leq_{\mathrm{m}}^{\mathrm{AC}^0}$ Valence-k GI for all $k \geq 3$.*

Proof. The proof idea as in [13] would increase the valence of vertices from k to $k+1$. Let G be a valence-k graph for some $k \geq 3$. We construct now a valence-k graph H as follows. Replace each edge by a path of length three. Let vertex $v \in V(G)$ be labeled with color $c_i \in \{c_1, \ldots, c_k\}$. Connect each vertex $v' \in \Gamma(v)$ in H with a path $u_1, \ldots, u_{2n+3}$ such that u_j is at distance j to v' and connect to u_{n+2} another path of length i (cf. [13]). Observe, that the neighbours of vertex v are of degree two and thus, we can connect them with copies of the new graph gadget. Thus, H is also of valence k. □

Let $\{x_1, \ldots, x_k\}, \{y_1, \ldots, y_k\} \subseteq V(G)$ be vertex sets of graph G. Let ϕ be a mapping with $\phi(x_i) = y_i, i \in [k]$ such that (G, ϕ) is an instance for prefix-GA. Adapting the proof of prefix-GA $\leq_m^{\mathrm{AC}^0}$ GI [13], we obtain the following chain of reductions.

Corollary 2. *For $k \geq 2$ valence-k prefix-GA $\leq_m^{\mathrm{AC}^0}$ valence-k color-GI $\leq_m^{\mathrm{AC}^0}$ valence-k GI.*

3.3 Valence-*k* Graph Isomorphism

GI is known to be hard for $\mathrm{Mod_kL}$ under AC^0 many-one reductions, for all $k \geq 2$ [22]. In this proof, the valence of the graphs are bounded by $k+1$.

An NC^1 circuit can be simulated by a balanced DLOGTIME uniform family of circuits with fan-out 1, logarithmic depth, polynomial size and alternating layers of and-gates and or-gates [4]. For simulation of NC^1 circuits graph gadgets were used as shown in proof of Lemma 3.1 in [12] for simulation of AND- and OR-gates in circuits. These graph gadgets have designated root vertices and are recursively connected again to root vertices of another graph gadgets. Before we present the main result of this section we define a special version of the prefix-GA problem.

Prefix-GA Problem with fixed vertex property:
Given: $G = (V, E)$ a connected graph and let $U \subseteq V$ and $\phi : U \mapsto U$ be a mapping which fixes at least one vertex.
Problem: Can ϕ be extended to an automorphism of G?

Since this version of prefix-GA is a special case of prefix-GA it also reduces to GI, even if we consider bounded valence graphs, then it also reduces to bounded valence GI. Let C be a $\mathrm{Mod_k}$ circuit with input values x. In Theorem 3.3 in [22] the author proved that there is an AC^0 computable function which computes for circuit C a valence-$k+1$ graph $G(C)$ and prefixes ϕ such that $(G(C), \phi)$ can be extended to an automorphism iff C outputs 1. Observe that this graph is connected and can be modified such that ϕ contains fixed vertices. For example, transform C into a circuit C' that adds 0 to the output value. Then some of the input vertices according to the new output-gate are fixed in ϕ and are of valence at most k. Thus, $(G(C'), \phi')$ is in Prefix-GA with fixed vertex property iff C (and C') outputs 1. But, the valence of $G(C')$ depends on k.

If there would be a different construction such that $(H(C'), \psi')$ with the same properties as $(G(C'), \phi')$ but with valence of $H(C')$ bounded by a constant $c \geq 3$, then valence-c GI would be hard for $\mathrm{Mod_kL}$. Then the following theorem says that valence-c GI would also be hard for #L.

Theorem 3. *If there exists a constant $c \geq 3$ such that for every $k \in \mathbb{N}$ every instance of the* $\mathrm{Mod_k}$ *circuit value problem can be* AC^0 *reduced to an instance of valence-c Prefix-GA with the fixed vertex property, which is of size polynomial in the circuit size (and independent of k) then valence-c GI is hard for* NL, #L, $\mathrm{C_{=}L}$ *and* PL *under* AC^0 *many-one reductions.*

Proof. We give a sketch of the important proof steps. Observe that $G'(C')$ is a connected valence-$k+1$ graph. Take two copies of this graph, say G_1, G_2 and apply prefixes as coloring to these graphs as in [22] and obtain H_1, H_2. Observe that by Lemma 1 coloring vertices does not change the valence of the graph. Because of fixed vertex property of ϕ there is at least one designated vertex v in H_1 and H_2 which has its own unique color in both graphs and is of valence k. Such graph pairs H_1, H_2 can be connected as input to the graph gadgets, encoding an NC^1 circuit in a similar way as in Theorem 4.4 in [22]. In this theorem different graph tuples are needed simulating AND and OR-functions. We can replace these graph gadgets by those in proof of Lemma 3.1 in [12] in order to keep the degrees of vertices bounded by a constant. □

3.4 Extensions

In Theorem 1 we proved that the logspace-complete problem ORD is AC^0 many-one reducible to valence-2 GI. We can adapt the proof to obtain the following result. Let *circle-GI* be graph isomorphism where the input graphs are given as a set of circles.

Theorem 4. *Circle-GI is logspace-complete under* AC^0 *many-one reductions.*

Proof. We reduce from ORD, which is logspace-complete via first-order projections [8]. Recall the definition in proof of Theorem 1. That is given a graph that is a directed line of size n and two designated vertices v_i, v_j. For the reduction we construct two graphs X, Y such that $X \cong Y$ if and only if $v_i < v_j$ in the order induced by G.

- X is an undirected cycle of size n.
- Y is the undirected version of G but with the following changes: $\{v_i, v_{i+1}\}$, $\{v_j, v_{j+1}\} \notin E(Y)$, but $\{v_1, v_j\}, \{v_{j+1}, v_i\}, \{v_{i+1}, v_n\} \in E(Y)$.
- If $i < j$ then Y consists of one circle of size n and is isomorphic to X. But if $j < i$ then Y consists of three circles and is not isomorphic to X.

This reduction is an AC^0 many-one reduction. Since ORD is L-complete via quantifier-free projections, circle-GI is hard for L under AC^0 many-one reductions. Completeness follows since circle-GI is a special case of the logspace-complete problem valence-2 GI. □

We observe that this result also generalizes to line-GI.

4 Hardness Results for valence-k Graph Automorphism

The decision problem valence-2 GA is trivial since it consists of isolated vertices, paths and cycles. Each path and cycle has nontrivial automorphisms. The only graph without nontrivial automorphisms consists of one isolated vertex. The situation is different for valence-k GA for $k \geq 3$. In this section we discuss hardness results for this problem.

GA is many-one hard for the $\mathrm{Mod_kL}$ hierarchy (Theorem 5.1 [22]). In this proof the same circuit graph $G(C)$ is used as in Theorem 3.3 [22]. In detail, prefixes were defined, that is two subsets of vertices in $G(C)$. Take two copies G_1 and G_2 of graph $G(C)$ and apply colorings $Col(G_1), Col(G_2)$ to the vertices which represent the input and output values of the circuit in order to encode these prefixes. Thus the graph $G_1 \cup G_2$ has a nontrivial automorphism, iff the output of the original circuit is 1.

Since the proof uses similar graphs as for proving GI is hard for $\mathrm{Mod_kL}$, the valence of the graphs is bounded the same way. Since coloring does not change the valence of the graphs G_1, G_2 we obtain the following corollary.

Corollary 3. *Valence-k+1 GA is hard for* $\mathrm{Mod_kL}$*, $k \geq 2$, under* AC^0 *many-one reductions.*

An important property of $G(C)$ is that this graph is rigid and coloring does not change rigidity. We will formulate Theorem 3 for bounded valence GA. In the following Theorem, by *GA with the fixed vertex property* we mean GA restricted to graphs wich contain fixed vertices in their automorphism group.

Theorem 5. *If there exists a constant $c \geq 3$ such that for every $k \in \mathbb{N}$ every instance of the* $\mathrm{Mod_k}$ *circuit value problem can be* AC^0 *reduced to an instance of valence-c GA with the fixed vertex property, which is of size polynomial in the circuit size (and independent of k) then valence-c GA is hard for* NL, #L, $\mathrm{C_{=}L}$ *and* PL *under* AC^0 *many-one reductions.*

Proof. We give a sketch of the important proof steps. Basically, we construct similar graphs as in Theorem 3. The graphs encoding a circuit with modulo addition gates can be connected as input to the graph gadgets, encoding an NC^1 circuit in a similar way as in Theorem 4.4 in [22]. In this theorem graph tuples are needed simulating AND and OR-functions but the vertices in the construction have no bounded degree. We can replace the graphs simulating an AND by those in proof of Lemma 3.1 in [12] in order to keep the degrees of vertices bounded by a constant. For a rigid graph gadget simulating an OR-function, we need a different construction, see figure 2. The graph gadget G^2 is the same as defined in Definition 3.1 in [22]. G^2 simulates a parity gate, that is an automorphism which maps x_i onto $x_{i\oplus a}$ and y_i onto $y_{i\oplus b}$ then maps z_i onto $z_{i\oplus a\oplus b}$ for all $i, a, b \in \{0, 1\}$. Observe, that this graph gadget preserves the rigidity conditions as desired.

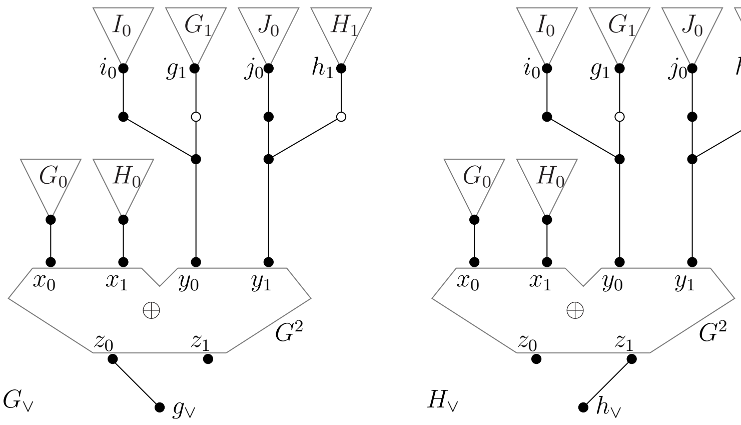

Fig. 2. Rigid graph gadgets for simulation of an OR-function.

□

Acknowledgements. I would like to thank an anonymous referee for pointing out an error in the previous version of the paper and Jacobo Torán for helpfull discussions.

References

1. E. Allender, M. Ogihara, *Relationships among PL, #*L *and the determinant* RAIRO Inform. Theor. Appl., 30, 1996, pp. 1-21.
2. C. Alvarez, B. Jenner, *A very hard logspace counting class* Theoretical Computer Science 107, 1993, pp. 3-30.
3. L. Babai, E. Luks, A. Seress, *Permutation Groups in NC* Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing (STOC), 1987, pp. 409-420
4. D.A.M. Barrington, N. Immerman, H. Straubing, *On uniformity within* NC^1 J. Comput. System Sci. 41, 1990, pp. 274-306.
5. R. Boppana, J. Hastad, S. Zachos, *Does co-NP have short interactive proofs?* Inform. Process. Lett. 25, 1987, pp. 27-32.
6. G. Buntrock, C. Damm, U. Hertrampf, C. Meinel, *Structure and importance of logspace-MOD-classes* Math. System Theory 25, 1992, pp. 223-237.

7. S. R. Buss, *Alogtime algorithms for tree isomorphism, comparison, and canonization in Computational Logic and Proof Theory* Lecture Notes in Comput. Sci. 1289, Springer Verlag Berlin 1997, pp. 18-33.
8. K. Etessami, *Counting quantifiers, successor relations, and logarithmic space* in Proceedings of the 10th Structure in Complexity Theory Conference, IEEE Computer Society Press, Silver Spring MD, 1995, pp. 2-11.
9. J. Gill, *Computational complexity of probabilistic Turing machines* SIAM J. Comput. 6, 1977, pp. 675-695.
10. C. Hoffmann, *Group-Theoretic Algorithms and Graph Isomorphism* Lecture Notes in Computer Science 136, 1982.
11. J.E. Hopcroft, R.E. Tarjan, *A V^2 algorithm for determining isomorphism of planar graphs* 1971, pp. 32-34.
12. B. Jenner, J. Köbler, P. McKenzie, J. Torán , *Completeness results for graph isomorphism* J. Comput. System Sci., 66, 2003, pp. 549-566.
13. J. Köbler, U. Schöning, J. Torán , *The Graph Isomorphism Problem - Its Structural Complexity* Prog. Theor. Comp.Sci., Birkhaeuser, Boston, MA, 1993.
14. S. Lindell, *A logspace algorithm for tree canonization* in Proceedings of the 24th Annual ACM Symposium on Theory of Computing, 1992, pp. 400-404.
15. E. Luks, *Isomorphism of bounded valence can be tested in polynomial time* J. Comput. System Sci. 25, 1982, pp. 42-65.
16. E. Luks, *Parallel algorithms for perumtation groups and graph isomorphism* in Proc of the 27th IEEE Symp. on Found. of Comp. Sci. 1986, pp. 292-302.
17. G. Miller, *Graph Isomorphism, General Remarks* Journal of Computer and System Sciences vol.18(2) 1979, pp. 128-142
18. G.L. Miller, J.H. Reif, *Parallel tree contraction* Part 2: further applications SIAM Journal on Computing 20(6), 1991, pp. 1128-1147.
19. W. Ruzzo, J. Simon, M. Tompa, *Space bounded hierarchies and probabilistic computations* J. Comput. System Sci. 28, 1984, pp. 216-230.
20. U. Schöning, *Graph isomorphism is in the low hierarchy* J. Comput. System Sci. 37, 1988, pp. 312-323.
21. T. Thierauf, F. Wagner, *The Isomorphism Problem for Planar 3-Connected Graphs is in Unambigious Logspace* Electronic Colloquium on Computational Complexity ECCC, TR07-068, 2007.
22. J. Torán , *On the Hardness of Graph Isomorphism* SIAM J. Comput. Vol. 33, No. 5, 2004, pp. 1093-1108.
23. F. Wagner, *Hardness Results for Tournament Isomorphism and Automorphism* In 32nd International Symposium on Mathematical Foundations of Computer Science (MFCS) Lecture Notes in Computer Science 4708, 2007, pp. 572-583.