A Complexity Theory for Implicit Graph Representations

Maurice Chandoo

Leibniz Universität Hannover, Theoretical Computer Science, Appelstr. 4, 30167 Hannover, Germany
chandoo@thi.uni-hannover.de

Abstract

In an implicit graph representation the vertices of a graph are assigned short labels such that adjacency of two vertices can be algorithmically determined from their labels. This allows for a space-efficient representation of graph classes that have asymptotically fewer graphs than the class of all graphs such as forests or planar graphs. A fundamental question is whether every smaller graph class has such a representation. We consider how restricting the computational complexity of such representations affects what graph classes can be represented. A formal language can be seen as implicit graph representation and therefore complexity classes such as P or NP can be understood as set of graph classes. We investigate this complexity landscape and introduce complexity classes defined in terms of first order logic that capture surprisingly many of the graph classes for which an implicit representation is known. Additionally, we provide a notion of reduction between graph classes which reveals that trees and interval graphs are 'complete' for certain fragments of first order logic in this setting.

1998 ACM Subject Classification G.2.2 Graph Theory, F.1.3 Complexity Measures and Classes

Keywords and phrases adjacency labeling schemes, complexity of graph properties

We call a graph class small if it has at most $2^{\mathcal{O}(n \log n)}$ graphs on n vertices. Many important graph classes such as interval graphs are small. Using adjacency matrices or lists to represent such classes is not space-efficient since this requires n^2 bits (resp. $n^2 \log n$ bits if the class contains dense graphs) whereas a representation that uses only $\mathcal{O}(n \log n)$ bits is possible. In an implicit representation, also called adjacency labeling scheme, each vertex is assigned a label of length $\mathcal{O}(\log n)$ such that adjacency of vertices can be inferred from their two labels. In contrast to an adjacency matrix or list, adjacency of the labels needs to be defined before the graph is known. The representation of a graph is the (multi)set of its labels and requires $n \cdot \mathcal{O}(\log n)$ bits as desired. Let us consider a labeling scheme for interval graphs as example. Recall that a graph G is an interval graph if there exists a function M that maps the vertices of G to intervals on the real line such that u, v are adjacent in G iff $M(u)$ and $M(v)$ have non-empty intersection. The image of M is called interval model of G. An implicit representation for G can be obtained by writing down its interval model, enumerating the endpoints of the intervals from left to right and labeling the vertices of G according to their intervals as shown in Figure 1. Notice that two numbers from $[2n]$ constitute a label and hence a label is $\log(4n^2)$ bits long.

▶ **Definition 1** (Labeling scheme). A labeling scheme is a tuple $S = (F, c)$ where $F \subseteq \{0,1\}^* \times \{0,1\}^*$ is called a label decoder and $c \in \mathbb{N}$ is the label length. A graph G on n vertices is in the class of graphs spanned by S, denoted by $G \in \mathrm{gr}(S)$, if there exists a labeling $\ell \colon V(G) \to \{0,1\}^{c \log n}$ such that for all $u, v \in V(G)$:

$$(u, v) \in E(G) \Leftrightarrow (\ell(u), \ell(v)) \in F$$

We say a graph class $\mathcal{C}$ is represented by (or has) a labeling scheme S if $\mathcal{C} \subseteq \mathrm{gr}(S)$.

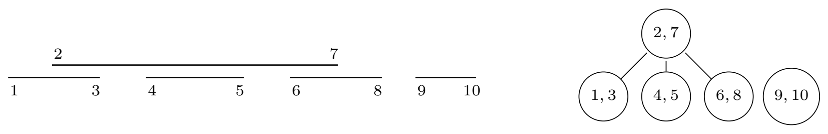

Figure 1 Interval model and the resulting labeling of the interval graph

The label decoder in our example can be defined as follows. A tuple of labels (x_1x_2, y_1y_2) with $|x_1| = |x_2| = |y_1| = |y_2|$ is in the label decoder if neither x_2 is (lexicographically) smaller than y_1 nor y_2 is smaller than x_1. This means that none of the two intervals ends before the other starts and therefore they intersect, assuming that $x_1 \leq x_2$ and $y_1 \leq y_2$. The label length in this case is $c = 4$ because $\log(4n^2) \leq 4\log(n)$ for all $n \geq 2$.

An important aspect of a labeling scheme is the computational complexity of its label decoder as it directly corresponds to the resources required to query an edge and thus should be as small as possible. The following interpretation of a binary language as label decoder allows us to classify labeling schemes in terms of classical complexity classes.

▶ **Definition 2.** A language $L \subseteq \{0,1\}^*$ induces a label decoder F_L where for all $x, y \in \{0,1\}^*$ with $|x| = |y|$ it holds that $(x, y) \in F_L \Leftrightarrow xy \in L$. For a set of languages A we say that a graph class $\mathcal{C}$ is in GA if there exists a language $L \in \mathsf{A}$ and $c \in \mathbb{N}$ such that $\mathcal{C}$ is represented by the labeling scheme (F_L, c).

If a graph class $\mathcal{C}$ is in GP then it has a labeling scheme with a polynomial-time computable label-decoder. This means that querying an edge requires only polylogarithmic time w.r.t. the number of vertices, which makes such a representation useful in practice. This prompts the question whether every small graph class is in GP. In general this is not the case [4, Thm. 2.1]. However, an arbitrary graph class can lack the structure one usually expects from natural graph classes and therefore a reasonable notion of uniformity should be imposed: a graph class is hereditary if it is closed under deleting vertices. Then we can ask the previous question for small, hereditary graph classes. This leads to the implicit graph conjecture which was originally posed as question by Kannan, Naor and Rudich in 1992 and rephrased as conjecture by Spinrad.

▶ **Conjecture 3** (IGC [2, 4]). *Every small, hereditary graph class is in* GP.

We approach this conjecture from a complexity-theoretic point of view. The following is a fruitful example of this approach. Originally the label decoder of an implicit representation was required to be computable [3] whereas later this requirement was tightened to polynomial-time computability [2]. However, until recently it was not known whether these two definitions coincide. In our terminology this is the question of whether $\mathsf{GP} = \mathsf{GR}$ where R is the class of decidable languages. Observe that proving separations in this context is even harder than in classical complexity since there can be different label decoders (languages) that represent the same graph class. For example, the fact that $\mathsf{P} \subsetneq \mathsf{EXP}$ does not directly imply that $\mathsf{GP} \subsetneq \mathsf{GEXP}$. Nonetheless, by applying diagonalization as known from the hierarchy theorems we were able to prove a similar hierarchy for implicit representations which implies $\mathsf{GP} \subsetneq \mathsf{GR}$ thereby resolving the previous question.

▶ **Theorem 4** ([1]). $\mathsf{GEXP} \subsetneq \mathsf{G2EXP} \subsetneq \mathsf{G3EXP} \subsetneq \cdots \subsetneq \mathsf{GR}$.

When trying to classify graph classes that are known to have a labeling scheme one will notice that in many cases their label decoders have a very low complexity. More specifically, the label decoders can be expressed as boolean formulas with arithmetic conditions as propositions. Such label decoders can be naturally seen as first-order formulas which motivates the following complexity classes.

Formulas are evaluated with respect to the following structures. For $n \geq 1$ let $\mathcal{N}_n$ be the structure that has $[n]$ as universe, the order relation $<$ on $[n]$ and addition as well as multiplication as functions:

$$+(x,y) = \begin{cases} x+y & \text{, if } x+y \leq n \\ 1 & \text{, if } x+y > n \end{cases} \quad , \quad \times(x,y) = \begin{cases} xy & \text{, if } xy \leq n \\ 1 & \text{, if } xy > n \end{cases}$$

For $\sigma \subseteq \{<,+,\times\}$ let $\mathsf{FO}_k(\sigma)$ be the set of first-order formulas with boolean connectives $\neg, \vee, \wedge$, quantifiers $\exists, \forall$ and k free variables using only equality and the relation and function symbols from σ.

▶ **Definition 5.** A (quantifier-free) logical labeling scheme is a tuple $S = (\varphi, c)$ with a (quantifier-free) first-order formula $\varphi \in \mathsf{FO}_{2k}$ and $c, k \in \mathbb{N}$. A (c,k)-labeling for a set V with n elements is a function $\ell \colon V \to [n^c]^k$ and induces the graph G^ℓ_S with vertex set V and edges (u,v) if $\mathcal{N}_{n^c}, (\ell(u), \ell(v)) \models \varphi$. Then a graph G is in $\mathrm{gr}(S)$ if there exists a (c,k)-labeling ℓ for $V(G)$ such that $G = G^\ell_S$.

Informally, one assigns k small numbers to each vertex and adjacency is determined by checking whether the $2k$ numbers of two vertices satisfy the formula. Notice that k numbers from $[n^c]$ can be encoded as a binary string of length $ck \log n$. For example, the representation for interval graphs that we have given in the beginning can be phrased as logical labeling scheme $S = (\varphi, 2)$ with $\varphi(x_1, x_2, y_1, y_2) = \neg(x_2 < y_1 \vee y_2 < x_1)$.

▶ **Definition 6.** Let $\sigma \subseteq \{<,+,\times\}$. A graph class $\mathcal{C}$ is in $\mathsf{GFO}(\sigma)$ if there exist $c, k \in \mathbb{N}$ and $\varphi \in \mathsf{FO}_{2k}(\sigma)$ such that $\mathcal{C}$ is represented by the logical labeling scheme (φ, c). We write $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ to denote the quantifier-free analogue. If $\sigma = \{<,+,\times\}$ we write GFO and $\mathsf{GFO}_{\mathrm{qf}}$.

Be aware that the eponymous complexity class from descriptive complexity which we denote by FO_{D} leads to the class of graph classes $\mathsf{GFO}_{\mathrm{D}} = \mathsf{GAC}^0$ which is presumably different from our class GFO. The expressiveness of our logical classes is bounded by:

▶ **Theorem 7** ([1]). $\mathsf{GFO} \subseteq \mathsf{GPH}$ *and* $\mathsf{GFO}_{\mathrm{qf}} \subseteq \mathsf{GTC}^0$.

Proof sketch. To simulate the label decoder of a logical labeling scheme one needs to consider the complexity of evaluating a fixed formula with $2k$ numbers as input. Every atom of φ corresponds to a polynomial (in)equation with $2k$ variables. After computing the truth value of every atom and plugging them into φ one is left with checking a boolean formula. If the formula is quantifier-free then evaluating the polynomial inequations is the most difficult part which can be done in TC^0. If φ contains quantifiers then they can be simulated using the non-determinism of the polynomial-time hierarchy. ◀

Consider that querying an edge in a labeling scheme from $\mathsf{GFO}_{\mathrm{qf}}$ takes only constant time in the RAM model with word length $\mathcal{O}(\log n)$ and comparison, addition and multiplication as unit operations. This makes $\mathsf{GFO}_{\mathrm{qf}}$ a practically relevant class.

Interestingly, many graph classes can be found in the even smaller fragment $\mathsf{GFO}_{\mathrm{qf}}(<)$, which is also a subset of GAC^0. Every graph class that is bounded with respect to one of the following parameters resides in $\mathsf{GFO}_{\mathrm{qf}}(<)$: degree, tree width, arboricity, edge clique cover number, boxicity and interval number. Also, every graph class closed under minors (e.g. planar graphs) is in $\mathsf{GFO}_{\mathrm{qf}}(\emptyset)$, except the class of all graphs. Therefore it is interesting to consider the following variant of the implicit graph conjecture:

▶ **Conjecture 8** (Weak IGC). *Every small, hereditary graph class is in* $\mathsf{GFO}_{\mathrm{qf}}$.

The fact that so many graph classes already lie in $\mathsf{GFO}_{\mathrm{qf}}$ makes it worthwhile to investigate whether this holds for other hereditary graph classes as well. Considering the restricted nature of q.f. logical labeling schemes as opposed to polynomial-time computable label decoders, trying to establish lower bounds seems to be a reasonable endeavor here. An important tool for this is a reduction notion between graph classes. Roughly, we say a class $\mathcal{C}$ is reducible to another class $\mathcal{D}$ if for every graph $G \in \mathcal{C}$ its vertices can be mapped to a subset of vertices of a polynomially larger graph $H \in \mathcal{D}$, i.e. $f\colon V(G) \to \mathcal{P}(V(H))$, such that adjacency of u, v in G can be determined from the induced subgraph of H on $f(u), f(v)$.

▶ **Definition 9.** Given two graphs G, H, $k \in \mathbb{N}$ and a boolean function $f\colon \{0,1\}^{k^2} \to \{0,1\}$. We say G has an (H, f)-representation if there exists a mapping $\ell\colon V(G) \to V(H)^k$ such that for all $u, v \in V(G)$ it holds that $(u, v) \in E(G)$ iff $f(x_{1,1}, x_{1,2}, \dots, x_{1,k}, x_{2,1}, \dots, x_{k,k}) = 1$ with $x_{i,j} = 1$ iff $(\ell(u)_i, \ell(v)_j) \in E(H)$ for all $i, j \in [k]$.

▶ **Definition 10.** Given two graph classes $\mathcal{C}$ and $\mathcal{D}$. We say $\mathcal{C}$ is reducible to $\mathcal{D}$, in symbols $\mathcal{C} \leq \mathcal{D}$, if there exist $c, k \in \mathbb{N}$ and a boolean function $f\colon \{0,1\}^{k^2} \to \{0,1\}$ such that for all $n \in \mathbb{N}$ and every graph $G \in \mathcal{C}$ on n vertices there exists a graph $H \in \mathcal{D}$ on n^c vertices such that G has an (H, f)-representation.

Notice that $\mathcal{C} \subseteq \mathcal{D}$ implies $\mathcal{C} \leq \mathcal{D}$ because every graph $G \in \mathcal{C}$ trivially has a (G, id)-representation where $\mathrm{id}(x) = x$. Therefore $\leq$ is reflexive. Let us consider the reduction of k-interval graphs to interval graphs as another example. Recall that a graph G is a k-interval graph if there exists a function M which maps every vertex of G to k intervals on the real line such that two vertices u, v are adjacent iff there exists an interval $I_u \in M(u)$ and $I_v \in M(v)$ such that I_u and I_v have non-empty intersection. We assume w.l.o.g. that no interval occurs twice. Let $M(G)$ denote the set of all kn intervals and $H_{M(G)}$ is the interval graph that is induced by this set of intervals. It holds that G has an $(H_{M(G)}, f)$-representation where f is the disjunction over k^2 variables and via the labeling $\ell = M$.

It is also not difficult to show that $\leq$ is transitive. The design rationale behind this reduction is that $\mathsf{GFO}(\sigma)$ and $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ are closed under it for all $\sigma \subseteq \{<, +, \times\}$. Also, every decent complexity class such as GP or GAC^0 is closed under this reduction. Therefore it satisfies all relevant properties required of a reduction.

We say a graph class $\mathcal{C}$ is complete for a complexity class $\mathsf{G}\cdot$ if $\mathcal{C}$ is in $\mathsf{G}\cdot$ and for every graph class $\mathcal{D}$ in $\mathsf{G}\cdot$ it holds that $\mathcal{D}$ is reducible to $\mathcal{C}$.

▶ **Theorem 11.** *Interval graphs are complete for* $\mathsf{GFO}_{\mathrm{qf}}(<)$.

Proof idea. Leaving technicalities aside, what lies at the heart of this proof is to represent a natural number by some intervals such that the order of two numbers can be expressed in terms of intersection of their intervals. This can be accomplished by mapping a number n to the two intervals $[1, n]$ and $[n, n]$. Then for two numbers x, y it holds that $x < y$ iff $[x, x] \cap [1, y] \neq \emptyset$ and $[1, x] \cap [y, y] = \emptyset$. ◀

▶ **Theorem 12.** *Trees are complete for* $\mathsf{GFO}_{\mathrm{qf}}(\emptyset)$.

This research direction opens up a variety of interesting, non-trivial yet seemingly solvable questions that can enhance our understanding of implicit representations. What is a complete class for $\mathsf{GFO}_{\mathrm{qf}}$? Do quantifiers enhance the expressiveness in the case of comparison, i.e. $\mathsf{GFO}_{\mathrm{qf}}(<) = \mathsf{GFO}(<)$? Also, can graph classes for which no labeling scheme is known (such as the intersection graphs of disks or line segments) be related via this reduction? And last but not least the weak implicit graph conjecture, which we deem to be the most interesting of these question.

References

1 Maurice Chandoo. On the implicit graph conjecture. In *41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Kraków, Poland*, pages 23:1–23:13, 2016. `doi:10.4230/LIPIcs.MFCS.2016.23`.

2 Sampath Kannan, Moni Naor, and Steven Rudich. Implicit representation of graphs. *SIAM Journal on Discrete Mathematics*, 5(4):596–603, November 1992.

3 John Harold Muller. *Local Structure in Graph Classes.* PhD thesis, Atlanta, GA, USA, 1988. Order No: GAX88-11342.

4 Jeremy P. Spinrad. *Efficient Graph Representations.: The Fields Institute for Research in Mathematical Sciences.* Fields Institute monographs. American Mathematical Soc., 2003.