Elementary classes of graphs in a language without equality predicate

Vladimir N. Remeslennikov, Alexander V. Treier
Sobolev Institute of Mathematics of SB RAS, Omsk, Russia
E-mail: vnremesl@gmail.com, alexander.treyer@gmail.com

March 2020

Abstract. In this paper we describe equivalence classes generated by finite graphs in the graph theory language without equality predicate. These classes coincide with classes of so called hard extensions of graphs.

1. Introduction

In this paper we will continue studying model theoretic problems for so called graph compression that we started in [1]. The notion of graph compression was defined earlier in [2].

In the category of simple graphs two finite graphs are isomorphic to each other if and only if they are elementary equivalent. From this point of view the language of graph theory $\mathcal{L}(E(x,y),=)$ has big expressive power. In the paper we deal with a weakened version of the graph theory language such that doesn’t contain equality predicate. On the other hand a language without equality still expressive enough. In the paper we describe an elementary class generated by an arbitrary finite simple graph in the mentioned language. The description is based on the notions of \(\perp\) and \(\circ\) graph compression that we introduce here. Our approach can be applied for various structures with binary relations and without equality predicate and also can be used for a description of universal classes and quasi-varieties.

Algebraic structures without equality predicate have studied in many papers. We can’t mention all of them here but we refer the reader to [3] for general approach to model theory without equality predicate. In general such languages are used in computer science particularly in the theory of logical programming, for example see [4].

2. Graph operators \(\perp\) and \(\circ\)

In this section we will define graph operators \(\perp\) and \(\circ\) that will be a basis for definition of graph compression. These notions were defined in [2] but here we plan to introduce
Elementary classes of graphs in a language without equality predicate

them using general approach that may be used in an arbitrary category of algebraic structures with binary relation.

Let $\mathcal{L}$ be the standard language of simple graphs with equality predicate and binary adjoining predicate $E(x, y)$. Recall that simple graph means that we consider graphs without loops i.e. $E(x, x)$ is always false and edges between vertices are undirected i.e. $E(x, y) = E(y, x)$ for any vertex $x$ and $y$.

We notice that the loopless condition isn’t necessary for our definitions and they also correct for graphs with loops. Let $\Gamma$ be a finite undirected graph with a vertex set $X$. Define $\perp$ and $\circ$ operators.

Definition 1 Let $x \in X$ then $x^\perp = \{y \in X | E(x, y)\} \cup \{x\}$ and $x^\circ = \{y \in X | E(x, y)\} \setminus \{x\}$.

It isn’t hard to check that our definitions of $\perp$ and $\circ$ are the same that were given in [2].

3. Graph compression and hard extensions

The operators $\perp$ and $\circ$ may be used to define equivalence relations on the vertex set $X$ of a graph $\Gamma$. For $x, y \in X$, $x \sim_\perp y$ if and only if $x^\perp = y^\perp$ and $x \sim_\circ y$ if and only if $x^\circ = y^\circ$. Denote by $[x]_\perp$ and $[x]_\circ$ the $\perp$ and $\circ$ equivalence classes of $x$ respectively.

The following Lemma establishes some basic properties of these equivalences.

Lemma 1 ([2])

(i) $[x]_\perp$ is clique for any $x \in X$.

(ii) $[x]_\perp \cap [x]_\circ = \{x\}$ for any $x \in X$.

(iii) If $|[x]_\perp| \geq 2$ then $|[x]_\circ| = 1$.

(iv) If $|[x]_\circ| \geq 2$, then $|[x]_\perp| = 1$.

The notion of graph compression was defined in [2]. In this article we want to introduce two type of graph compression such that may be considered as partial compression of the graph compression defined in [2].

Up to the end of the section let $\sim$ be the $\sim_\perp$ or $\sim_\circ$ equivalence and we put the symbol $t = \perp$ or $t = \circ$ respectively as type of chosen operator. We will define $\perp$ or $\circ$ graph compression by the following way

Definition 2 The $t$-compression of the graph $\Gamma$ is the graph $\Gamma^t$ with vertices $X^t = \{[x] | x \in X\}$ and edge joining vertices $[x]$ and $[y]$ if and only if $x$ and $y$ are incident in $\Gamma$.

Directly from the definitions we see that $\Gamma^t$ is a graph and we obtain the following fact.

Proposition 1 The mapping $t : X \rightarrow X^t$ given by $t(x) = [x]$, $x \in X$ induces a surjective graph homomorphism $t : \Gamma \rightarrow \Gamma^t$. 
Elementary classes of graphs in a language without equality predicate

Examples. On the examples below original graphs are on the left and compressed graphs are on the right. We omit brackets [ ] for those vertices of compressed graphs such that their equivalence classes contain only one vertex.

Let us introduce the notion of so called $t$-hard extensions of simple graph, where the symbol $t$ means $\perp$ or $\circ$. Let $\Gamma$ be a finite simple graph such that $\Gamma' \cong \Gamma$ as simple graphs. Then we call a graph $H$ as a $t$-hard extension of $\Gamma$ if and only if $H' \cong \Gamma$.

Example 1. Below the graph $H$ is a $\circ$-hard extension of the graph $\Gamma$. 
Elementary classes of graphs in a language without equality predicate

As it was before let $t \in \{\bot, \circ\}$. Let a finite simple graph $\Gamma$ be such that $\Gamma = \Gamma^t$. Denote by $\mathcal{K}_t^\Gamma$ the class of all hard $t$-extensions of the graph $\Gamma$. In the next section we will show relations between hard extensions of graphs and elementary classes of graph in special graph theory languages without the equality predicate.

4. Elementary classes of graphs in the language without equality predicate

Let $\mathcal{L} = \langle E(x, y), = \rangle$ be the standard language of simple graphs. Let $\Gamma$ be a finite simple graph with vertex set $X = \{x_1, \ldots, x_n\}$. All logical formulas consisting of the equality predicate $=$ and the adjoin predicate $E(x, y)$ which hold on the graph $\Gamma$ are called an elementary theory of the language $\mathcal{L}$ generated by $\Gamma$ and noted by $Th_{\mathcal{L}}(\Gamma)$. Two graphs are elementary equivalent to each other if their elementary theories coincide.

We refer the reader to [5] for detailed definitions of formulas, elementary equivalence, elementary classes and diagram formulas.

Informally speaking $\mathcal{L}$ has strong expressive power since many properties of graphs can be expressed with formulas of the language $\mathcal{L}$. Moreover any finite simple graph can be defined by the following $\exists \forall$ formula:

$$\phi_{\Gamma} = \exists x_1, \ldots x_n \forall y D_{\Gamma}(x_1, \ldots, x_n) \land (y = x_1 \lor \ldots \lor y = x_n),$$  \hspace{1cm} (1)

where $D_{\Gamma}(v_1, \ldots, v_n)$ is a diagram formula constructed by $\Gamma$ (see [5] for detailed definition of diagram formulas).

In this section we will consider two non standard graph languages. The first one is the language of simple graphs without the equality predicate. We put $\mathcal{L}^*$ as a notation for this language. The second one is also equality free but we put that $E(x, x)$ holds for all vertices. In other words we virtually add a loop to each vertex in this category of graphs. We will use $\mathcal{L}'$ notation for this language. Of course new languages are less expressive than $\mathcal{L}$ but they still contain the main attribute of graph theory: the adjoining predicate $E(x, y)$. Therefore any simple graph can be considered as model of three languages: $\mathcal{L}, \mathcal{L}^*$ and $\mathcal{L}'$. 
**Elementary classes of graphs in a language without equality predicate**

On the other hand the formula 1 doesn’t belong to \( \mathcal{L}^s \) as well as \( lI \) and an elementary class generated by the graph \( \Gamma \) differs from the case of the standard graph language. How elementary class generated by the finite graph \( \Gamma \) looks in the languages elementary class generated by the finite graph \( \Gamma \) differs from the case of the standard graph \( \mathcal{L} \) and \( \mathcal{L}^s \)? The answer is given in the following theorem.

**Theorem 1** Let \( \Gamma \) be a finite simple graph, \( \Gamma^\circ \) and \( \Gamma^\perp \) are \( o \) and \( \perp \) compressed graphs respectively. Let \( \mathcal{K}_{T^\circ}^\circ \) be the class of \( o \)-hard extensions of \( \Gamma^\circ \) and \( \mathcal{K}_{T^\perp}^\perp \) be the class of \( \perp \)-hard extensions of \( \Gamma^\perp \). Denote by \( T_{\mathcal{L}^s} (T_{\mathcal{L}^t}) \) the elementary theory of \( \Gamma \) in the language \( \mathcal{L}^s (\mathcal{L}^t) \). Then the following holds:

(i) \( \Gamma \equiv_{\mathcal{L}^*} \Gamma^\circ \) and \( \text{Mod}(T_{\mathcal{L}^*}) = \mathcal{K}_{T^\circ}^\circ \);

(ii) \( \Gamma \equiv_{\mathcal{L}^t} \Gamma^\perp \) and \( \text{Mod}(T_{\mathcal{L}^t}) = \mathcal{K}_{T^\perp}^\perp \).

**Proof.** Proofs of two parts of the theorem are identically up to chosen extension type symbol (\( \perp \) or \( o \)). Therefore without loss of generality we will give a proof of the first part. To reduce heavy notations we will keep in mind that all formulas belong to \( \mathcal{L}^s \). Let’s prove \( \mathcal{K}_{T^\circ}^\circ \subseteq \text{Mod}(T) \) where \( T = Th(\Gamma) \). At first we show that any \( o \)-hard extension \( H \) of \( \Gamma^\circ \) is elementary equivalent to \( \Gamma^\circ \) in \( \mathcal{L}^s \).

The proof will be based on Ehrenfaucht-Fraisse game for \( H \) and \( \Gamma^\circ \) (see [6] for detailed description of Ehrenfaucht-Fraisse game). Let \( k \) be an arbitrary natural number and we consider \( k \)-step Ehrenfaucht-Fraisse game. The reader may look at Example 1 for a particular case of graphs \( H \) and \( \Gamma = \Gamma^\circ \) for the described game. The winning strategy of Duplicator is the following: at each step if Spoiler chose some vertex \( x \) from \( H \) then Duplicator choose its equivalence class \([x]\) in \( \Gamma^\circ \) and vice versa if Spoiler chose a vertex \( x \) from \( \Gamma^\circ \) then Duplicator choose any vertex from \( \sim_o \)-equivalence class of \( x \). Finally we get two \( k \)-tuples \( a_1, \ldots, a_k \in H \) and \( b_1, \ldots, b_k \in \Gamma^\circ \) where some elements may coincide. It is not hard to see that by definitions of \( \sim_o \)-equivalence and described strategy that for any \( i, j \) \( E(a_i, a_j) = E(b_i, b_j) \). Therefore these tuples can’t be distinguished by atomic formulas and \( H \equiv \Gamma^\circ \). Since \( \Gamma \) is also belongs to \( \mathcal{K}_{T^\circ}^\circ \), then we have \( H \equiv \Gamma^\circ \equiv \Gamma \) and \( \mathcal{K}_{T^\circ}^\circ \subseteq \text{Mod}(T) \).

Now we prove that \( \text{Mod}(T) \subseteq \mathcal{K}_{T^\circ}^\circ \). The proof will rely on the following lemma.

**Lemma 2** The class of hard extensions \( \mathcal{K}_{T^\circ}^\circ \) is elementary class and it can be axiomatized by the following formula:

\[
\exists x_1, \ldots, x_k \forall y D^\circ_T(x_1, \ldots, x_k) \land (y \sim_o x_1 \lor \ldots \lor y \sim_o x_k)
\]

(2)

Proof of the lemma is based on the following clear fact. Two vertices \( x \) and \( y \) of a finite graph are \( \sim_o \)-equivalent if and only if they can’t be distinguished by atomic formulas, i.e for any \( z \in X \) we have \( E(x, z) = E(y, z) \).

Axiom 2 holds for the graph \( \Gamma \). Hence \( Th(\mathcal{K}_{T^\circ}^\circ) \subseteq Th(\Gamma^\circ) \). Therefore \( \text{Mod}(T) \subseteq \mathcal{K}_{T^\circ}^\circ \). Q.E.D.

**Acknowledgments**

The work was supported by Russian Science Foundation grant 18-71-10028
Elementary classes of graphs in a language without equality predicate

References

[1] Remeslennikov V, Treier A 2019 Universal classes in the category of simple graphs J. Phys.: Conf. Ser. 1260 022009
[2] Duncan A J, Kazachkov I V, Remeslennikov V N 2008 Orthogonal systems in finite graphs Siberian Electronical Mathematical Reports 5 pp 151–176
[3] Elgueta Raimon 1997 Characterization Classes Defined without Equality Studia Logica. 58 pp 357-394
[4] Gallier J, Raatz S 1987 HORNLOG: A graph-based interpreter for general Horn clauses The Journal of Logic Programming 4 2 pp 119-155
[5] Daniyarova E Yu, Miasnikov A G, Remeslennikov V N 2016 Algebraic geometry over algebraic structures Publishing House SB RAS Novosibirsk
[6] Hodges W 1993 Model Theory Encyclopedia of Mathematics and its Applications, Cambridge University Press