© Algebra and Discrete Mathematics
Volume **36** (2023). Number 2, pp. 225–242
DOI:10.12958/adm2157 RESEARCH ARTICLE

The presentation of deterministic and strongly deterministic graphs

O. S. Senchenko and M. I. Prytula

Communicated by A. V. Zhuchok

Abstract. The paper proposes a presentation of D-graphs and SD-graphs by a defining pair of words in the alphabet of their vertex labels. We present an algorithm that, given an arbitrary pair of sets, either constructs a D-graph for which this pair is the defining pair or informs that it is impossible to do so. We also present an algorithm for constructing a canonical defining pair for a D-graph and find some numerical estimates of this pair.

Introduction

A presentation of various mathematical structures is one of the most important and useful ways to specify them in practice. The most well-known are the presentations of groups, semigroups, and finite automata by generators and defining relations between them. In the context of the presentation theory, there are various problems, many of which have significant application value. Faithful presentations of endomorphism semigroups of some graphs and hypergraphs were considered, e.g., in [1], [2]. At the same time, for non-classical algebraic structures it is known that trioids and ordered doppelsemigroups can be represented by planar trees and binary relations, respectively (see, e.g., [3], [4], [5]).

The authors are partially supported by the Grant EFDS-FL2-08 of the found The European Federation of Academies of Sciences and Humanities (ALLEA)

2020 Mathematics Subject Classification: *68R10, 05C62, 05C78.*

Key words and phrases: *deterministic graph, presentation, defining pair.*

In the research of the presentation of finite automata without output in [6], a special presentation of the automaton (the canonical system of defining relations) was proposed, and the problem of characterization (comparing this automaton with the reference automaton only by its presentation) was solved. This solution used a special procedure for transforming the presentation to the canonical system of defining relations of the reference automaton.

It is natural to want to extend the results to other objects that are similar to automata. In our opinion, it would be logical to try to extend the results to deterministic graphs [7] (including their subclass, strongly deterministic graphs) for the following reasons:

1) Labeled graphs are widely used in computer science to describe and model various computational processes. The most studied are finite directed graphs with labeled edges (Labeled Transition System [8], weighted automata [9], finite automata etc). There are many computational processes in programming, robotics [10], model verification [11], which are naturally presented by graphs with labeled vertices, including deterministic graphs.

2) Such graphs have a certain resemblance to finite automata with no output.

3) To study the structure of objects that can be modeled by vertex-labeled graphs (including deterministic and strongly deterministic graphs), one or more mobile agents with limited memory are often used and placed in the object under study. This method of presentation of a graph is very convenient for such a research.

It should be noted that no generally accepted concepts and apparatus for presenting deterministic graphs have been developed so far. Some attempts to do so were made in [12], so to develop, generalize, and systematize these results, this paper attempts to develop a theory of presentation of deterministic and strongly deterministic graphs.

In this paper, we propose a presentation of deterministic and strongly deterministic graphs using two sets of words, the first set describes the cycles of graph and the second set describes their leaf vertices. We separated the functions of these two sets: each cycle must be described by at least one word from the first set, and each leaf vertex must be described by at least one word from the second set. We propose an algorithm that, given a pair of word sets that meet certain conditions, either construct a deterministic graph or reports that it is impossible to do so (in other words, the pair of sets is not correct). We also found a criterion for sets of words, according to which, if this algorithm builds a graph, this graph will necessarily be strongly deterministic. Similarly to the canonical system of defining relations [6], we propose a special defining pair, which is called canonical. The numerical characteristics of the power of the components of the canonical defining pair and the lower bounds on the volume of these components are found. We also propose a subclass of strongly deterministic graphs for which the volume of the first component of the canonical defining pair is most likely to be maximal.

1. Main definitions

In this paper, we consider undirected, finite, non-empty, connected, vertex-labeled simple graphs [13] $G = (V, E, X, \xi(V))$, where V is the set of vertices of the graph, E is the set of its edges, $\xi(V) : V \to X$ is a total function for labeling the vertices of a graph by symbols of a finite alphabet $X = \{x_1, \dots, x_p\}$. The value of $\xi(V)$ for a vertex v is called the label of v. The set of all words of finite length in the alphabet X (including the empty word ε) is denoted by X^*. Let $p, q \in X^*$, then the concatenation of p and q is denoted by pq. Let $p = x'_1 \dots x'_k$ ($x'_i \in X$), then the length of the word p is denoted by $d(p)$. Let $E(v)$ denote the set of vertices adjacent to vertex v: $v' \in E(v) \leftrightarrow (v, v') \in E$, the degree of a vertex v is the number $|E(v)|$. A vertex of degree 1 is called a leaf or a leaf vertex.

In this paper, we consider deterministic and strongly deterministic graphs. A labeled graph G is called deterministic [7] (or a D-graph) if all vertices in the open neighborhood of every its vertex have different labels. A D-graph G is called strongly deterministic [7] (or a SD-graph) if all vertices in the closed neighborhood of every its vertex have different labels. In other words, a D-graph can have two adjacent vertices with the same label, but this cannot happen in a SD-graph. It is not difficult to see that SD-graphs are the graphs with distance-2 colorings. In our paper we will consider SD-graphs as a special case of D-graphs, which are not "colored" in the classical sense (see, e.g., [13]), since in D-graphs two adjacent vertices can have the same labels. Let $\mathcal{G}_{n,m}$ denote the class of D-graphs with n vertices and m edges.

Let's fix some vertex $v_0 \in V$ of the D-graph $G = (V, E, X, \xi(V))$, which we will call the root vertex, this vertex will be highlighted in the notation of the D-graph if necessary: $G = (V, E, X, \xi(V), v_0)$. Let $G = (V_1, E_1, X, \xi_1(V_1), v_0)$ and $H = (V_2, E_2, X, \xi_2(V_2), v_0)$. If $V_1 \subseteq V_2$,

$E_1 \subseteq E_2$ and $\xi_1(V_1) \subseteq \xi_2(V_2)$, then G is a subgraph of H (and H a supergraph of G), written as $G \subseteq H$. If $G \subseteq H$ and $G \neq H$, then G is a proper subgraph of H.

Let $G_1 = (V_1, E_1, X, \xi_1(V_1), v')$ and $G_2 = (V_2, E_2, X, \xi_2(V_2), v'')$ be D-graphs. A bijective mapping $\varphi : V_1 \to V_2$ is called an isomorphism if $\varphi(v') = v''$, $\forall v \in V_1 \quad \xi_1(v) = \xi_2(\varphi(v))$ and $\forall v_1, v_2 \in V_1 \quad (v_1, v_2) \in E_1$ if and only if $(\varphi(v_1), \varphi(v_2)) \in E_2$. An isomorphism of graphs G_1 and G_2 is denoted by $G_1 \cong G_2$. Let v_1' be some vertex of the D-graph $G = (V, E, X, \xi(V))$. Since the path $p = v_1' \dots v_k'$ uniquely corresponds to the word $\xi(p) = \xi(v_1') \dots \xi(v_k')$, then we will consider the path p as $\xi(p)$ and denote it by the equality $v_1'\xi(p) = v_k'$. The path (or the word) $p = x_1'x_2' \dots x_k'$ is called valid for a vertex $v_1' \in V$ if $\xi(v_1') = x_1'$, and there exist vertices $v_2', \dots, v_k' \in V$ such that $\xi(v_2') = x_2', \dots, \xi(v_k') = x_k'$, and $(v_1', v_2'), \dots, (v_{k-1}', v_k') \in E$. Let $p = x_1' \dots x_k'$. The inverse of p $(x_k' \dots x_1')$ is denoted by p^{-1}. We call a word q a prefix of a word p if there exists w such that $p = qw$, this fact is denoted by $q \subseteq p$.

All vertices of a given D-graph $G = (V, E, X, \xi(V), v_0)$ can be divided into two classes by the following procedure: remove all leaves vertices from G other than the root vertex, along with the edges incident to these vertices. This procedure will be repeated as long as possible. The graph $B(G)$ obtained in this way is called the base of the graph G, the vertices included in $B(G)$ are called base vertices, and those vertices of G that are not included in $B(G)$ are called free vertices. It is easy to see that the base of the graph is constructed uniquely in a finite number of steps, the base of $B(G)$ may consist of one root vertex (in the case when G is a tree), and if $B(G)$ contains at least one vertex other than the root vertex, then it must contain at least one simple cycle. It follows that every base vertex that is different from the root vertex v_0 is adjacent to at least two base vertices.

2. Defining pair for D-graphs and SD-graphs

As mentioned above, the structure of objects that can be presented as D-graphs is often studied with the using of agents that are placed in the object under study (hereafter we associate this object with the D-graph). These agents can communicate certain information to the observer and/or other agents about the local neighborhoods of the vertices they are currently located on. Based on this information and/or instructions from the observer, agents can move along the edges of the graph.

The movements of the agents determine the sequences of labels of the vertices they have visited. In this regard, D-graphs are very convenient for such a study, since if the observer has a map of the graph (the sets V, E, X and the function $\xi(V)$) and knows which vertex of the studied D-graph the agent was placed on at the beginning of the study (the root vertex), then the trajectory of the agent's movement along the graph can be uniquely restored from this sequence of labels.

If the observer does not know the map of the graph under study, the movements of the agents can be organized by him in such a way that, based on their analysis, the observer receives the desired information about the graph structure (for example, drawing a map of the graph, searching for the shortest paths in it, comparing the given graph with the reference graph). Such an analysis can greatly simplify our presentation of a D-graph, by which we mean the mapping of one or more finite sets of words in the alphabet of graph vertex labels that describe possible (and/or impossible) trajectories in the graph. For such a presentation, you need to determine the number of such sets of words (preferably as few as possible) and the function that each of these word sets execute in the graph (which structural elements of the graph each set describes). At the same time, these sets should be chosen in such a way that a universal algorithm can be invented that either unambiguously constructs a D-graph that matches these sets of words or informs that it is impossible to do so. It is also necessary to invent an algorithm for transition from any D-graph to the above sets of words.

At the same time, it is desirable that such a presentation satisfies the following conditions:

- The presentation of graphs should be similar to the presentation of automata [6].

- The presentation of SD-graphs must be similar to the presentation of D-graphs, but the concepts, algorithms, etc. proposed by us must preserve the strong determinism of graphs (i.e., all transformations performed on a graph known to be strongly deterministic must not deprive it of its strong determinism).

Further, we assume that all paths in the graph start from the root vertex. By the term "pair" $\{C, L\}(x')$ we mean two finite sets of words C and L that satisfy the conditions:

1) Each word of the set C begins and ends with the symbol x'.

2) Each word of the set L begins with the symbol x'.

3) The length of each word of C is greater than 2; the length of each word of L is greater than 1.

The sets C and L are called the components of the pair $\{C, L\}(x')$. If each word of each component of the pair $\{C, L\}(x')$ does not contain a sequence of two identical symbols, then such a pair is called strong.

Let's state the requirements that a D-graph $G = (V, E, X, \xi(V), v_0)$, whose alphabet X coincides with the set of all symbols that appear in words from all components of a pair $\{C, L\}(x')$, is considered a presentation of this pair:

a) $\xi(v_0) = (x')$.

b) All words from C and L are valid for v_0.

c) Each word $p \in C$ describe the cycle $v_0 p = v_0$ in G.

d) For each word $q \in L$ the vertex $v_0 q$ is a leaf in G and $v_0 q \neq v_0$.

e) For each leaf vertex $v \in V$ that is different from the root vertex v_0, there exists at least one word $q \in L$ such that $v_0 q = v$.

f) For any pair $\{C, L\}(x')$, either no presentation exists or this presentation is uniquely determined.

At the same time, several different pairs can correspond to the same presentation.

Fix on X the linear order $<$: $x_1 < x_2 < \cdots < x_p$. We define the linear order $\preceq$ on X^*:

1) for all $x_i \in X$, where $1 \le i \le p$, let $x_i \preceq x_i$;

2) for all $p, q \in X^*$, if $d(p) < d(q)$, then $p \preceq q$;

3) for all $p, q \in X^*$, if $p = x'_1 \ldots x'_s$, $q = x''_1 \ldots x''_s$, $x'_1 = x''_1$, $\ldots$, $x'_{k-1} = x''_{k-1}$ and $x'_k < x''_k$, then $p \preceq q$.

Let $G = (V, E, X, \xi(V), v_0)$ be some vertex-labeled graph. A reduction of G is a procedure that transforms G into a graph $[G]$ using the following algorithm (the AR-algorithm):

0) Set $V' = V, E' = E, \xi'(V') = \xi(V)$, denote by $G' = (V', E', X, \xi'(V'), v_0)$.

1) For all $v \in V'$, we find the shortest word w in $\preceq$ that corresponds to the path from v_0 to v. Associate v with w.

2) Sort the vertices of the graph G' by their associated words in order $\preceq$. The current vertex v_c is the first vertex in the specified order, and the current label is $x = x_1$.

3) If there are such different vertices $v'_1, \ldots, v'_q$ for which the conditions $v'_1, \ldots, v'_q \in E'(v_c)$ and $\xi'(v'_1) = \ldots = \xi'(v'_q) = x$ are simultaneously satisfied, then we denote $U = \{v'_1, \ldots, v'_q\}$ and perform the following sequence of actions:

3.1. add a new vertex v' to V' and set $\xi'(v') = x$;

3.2. remove the edges (v_i, v_j) from E', where $v_i, v_j \in U$;

3.3. for all (v_i, v_j), where $v_i \in U$, add (v', v_j) to E';

3.4. remove the edges (v_i, v_j), where $v_i \in U$, from E', remove v from V', where $v \in U$;

3.5. if $v_0 \in U$, then the vertex v' is renamed to v_0 and we consider this vertex as root;

3.6. remove the repeating edges, preserving only one copy of each;

3.7. go to step 1.

4) If there exists $x' \in X$ that is next to x in the order $<$ (i.e., $x \neq x_p$), then set $x = x'$ and go to step 3.

5) If there exists v that is next to v_c, then set $v_c = v$, $x = x_1$ and go to step 3, otherwise the AR-algorithm finishes and $[G] = G'$.

It is easy to see that the execution of the AR-algorithm ends in a finite number of steps, its result is uniquely determined, and this result is a D-graph.

Remark 1. When executing this algorithm, there is some ambiguity with the names of the new vertices (except in the case when the newly created vertex v' was renamed to v_0 in step 3.5), but the names of all other vertices (except the root one) in the graph are not important to us. In this circumstance, if step 3 of the AR-algorithm has been executed, we can say that $[G]$ is isomorphic to some graph. It is easy to see that $[G] = G$ if and only if G is a D-graph. In addition, if there are no edges between any vertex labeled x'_1 and any vertex labeled x'_2 in G, then there are no edges between vertices with these labels in $[G]$. This fact will be used below when finding a criterion for constructing a SD-graph.

We define an AP-algorithm that, given a pair $\{C, L\}(x')$, either construct the D-graph $G(\{C, L\}(x'))$ or shows that it is impossible to construct a D-graph that meets conditions (a) – (e).

0) Initially, the graph $G(\{C, L\}(x'))$ consists of a single vertex v_0 labeled $\xi(v_0) = x'$.

1) For each word $p^i = x'x_1^i \dots x_n^i x' \in C$ we add vertices $v_1^i, \dots, v_n^i$ with labels $x_1^i, \dots, x_n^i$ and edges $(v_0, v_1^i), (v_1^i, v_2^i), \dots, (v_{n-1}^i, v_n^i)$, (v_n^i, v_0). After each such addition, we reduce the resulting graph by the AR-algorithm.

2) For each word $p^j = x'x_1^j \dots x_n^j \in L$, we add vertices $v_1^j, \dots, v_n^j$ with labels $x_1^j, \dots, x_n^j$, edges $(v_0, v_1^j), \dots$, (v_{n-1}^j, v_n^j) and reduce the resulting graph by the AR-algorithm.

3) We consider all the words of L: if there exists $p \in L$ such that the vertex v_0p is not leaf or $v_0p = v_0$, then we assume that the graph $G(\{C, L\}(x'))$ does not exist.

4) For each leaf vertex $v \in G(\{C, L\}(x'))$ other than v_0, we consider the words of the component L: if there is no such $p \in L$ that $v = v_0p$, then we assume that the graph $G(\{C, L\}(x'))$ does not exist.

5) For each $p \in C \bigcup L$ if the path p is not valid for v_0 in the resulting graph, then we assume that the graph $G(\{C, L\}(x'))$ does not exist.

If, as a result of this procedure, it is possible to construct the graph $G(\{C, L\}(x'))$ from a pair $\{C, L\}(x')$, then such a pair is called correct. In the AP-algorithm, the first and second stages create a certain graph, which, if the checks in stages $3 - 5$ are successful, is the graph $G(\{C, L\}(x'))$. If at least one of the checks in stages $3 - 5$ fails, then we assume that the graph $G(\{C, L\}(x'))$ does not exist.

From the fact that vertices and edges of $G(\{C, L\}(x'))$ are constructed exclusively by words from the sets C and L, words from a strongly pair do not have a sequence of two identical labels, and, therefore, at any step of constructing $G(\{C, L\}(x'))$ there are no adjacent vertices with the same label, and the reduction procedure will not lead to the appearance of adjacent vertices with the same label, it follows

Lemma 1. *If $\{C, L\}(x')$ is a correct strong pair, then the graph $G(\{C, L\}(x'))$ is strongly deterministic.*

It is easy to see that if the pair $\{C, L\}(x')$ is correct, the graph $G(\{C, L\}(x'))$ satisfies the requirements (a) – (e), any proper subgraph of $G(\{C, L\}(x'))$ does not satisfy the requirement (b), and any proper supergraph of $G(\{C, L\}(x'))$ satisfies the requirements (a) – (c), but may not satisfy the requirements (d) and (e). So, if the pair $\{C, L\}(x')$ is correct, then the graph $G(\{C, L\}(x'))$ is the smallest inclusion graph that meets the requirements (a) – (e), so we consider it a presentation by the pair $\{C, L\}(x')$.

A correct pair $\{C, L\}(x')$ is called *defining* for a D-graph G if $G(\{C, L\}(x')) \cong G$.

To summarize the above, we can consider the AP-algorithm to be a partial map of the set of pairs to the set of D-graphs, whereby the correct pair corresponds to the D-graph constructed by the AP-algorithm.

Let's consider the operation of individual stages of the AP-algorithm using the following example.

Example 1. Let $C = \{12341, 142451\}$, $L = \{152125423, 14523\}(1)$. Figure 1 (a) shows the graph obtained after executing step 1, Figure 1 (b) shows the graph obtained after executing step 2. In this graph, the vertices $v_0 152125423$ and $v_0 14523$ are leaves, so the check in step 3 is successful. For the selected vertex v with the label 1, there is no word $p \in L$ such that $v_0 p = v$, i.e., the check in step 4 fails, so the graph $G(\{C, L\}(1))$ does not exist.

Note that for a pair $\{C, L'\}(1)$, where $L' = L \bigcup \{1521\}$, the graph $G(\{C, L'\}(1))$ exists, it is shown in Figure 1 (b).

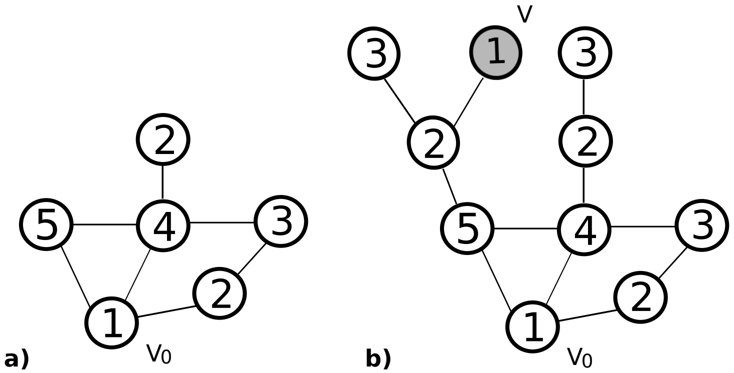

Figure 1. Illustration of the procedure for constructing a D-graph for a given pair

3. Canonical defining pair

Let $G = (V, E, X, \xi(V), v_0)$ be a D-graph and $V = \{v_0, \ldots, v_{n-1}\}$. Let's define the auxiliary set of words in the alphabet X. The reachability basis of $\mathcal{V}_G$ is the set of words $\{w_1, w_2, \ldots, w_n\}$, where for each vertex v_i there exists a word $w_i \in \mathcal{V}_G$, such that $v_0 w_i = v_i$, and for any $w \neq w_i$ from $v_0 w = v_i$, it follows $w_i \preceq w$. The spanning tree of a graph G, defined by the basis $\mathcal{V}_G$, is denoted by $T(\mathcal{V}_G)$ or $T(G, v_0)$. For each vertex $v \in V$, we denote by sp_v a word from $\mathcal{V}_G$ such that $v_0 sp_v = v$.

Let's describe the algorithm (the AC-algorithm) for constructing the defining pair $\{\Sigma_G, \Lambda_G\}$ for a D-graph G, which, due to some of its properties, we call canonical.

First, we set $\Sigma_G = \emptyset$ and $\Lambda_G = \emptyset$. If the graph G consists of a single vertex v_0, then we set $\Sigma_G = \emptyset$, $\Lambda_G = \emptyset$ and $\mathcal{V}_G = \{\xi(v_0)\}$.

Suppose a graph G contains more than one vertex. First, we add to the set Λ_G all words $w \in \mathcal{V}_G$ such that the vertex $v_0 w$ is a leaf vertex of G. After that, for each pair of words $p, q \in \mathcal{V}_G \setminus \Lambda_G$, if neither of them is a prefix of the other and $v_0 p q^{-1} = v_0$, then we add one of the two words

pq^{-1} or qp^{-1} to the set Σ_G, which is lower in the order $\preceq$ (hereafter, the vertices v_0p and v_0q^{-1} are called the generators for the word pq^{-1} or qp^{-1} that was added to Σ_G).

Let's describe some properties of the canonical defining pair that directly follow from the AP- and AC-algorithms.

Theorem 1. *Let $\{\Sigma_G, \Lambda_G\}$ be the canonical defining pair of a D-graph $G = (V, E, X, \xi(V), v_0)$.*

1) If at least one component $\{\Sigma_G, \Lambda_G\}$ is not an empty set, then for each vertex $v \in V$ there exist such $z_1, z_2, z_3 \in X^*$ that at least one of the statements is true: a) $sp_v z_1 \in \Lambda_G$; b) $sp_v z_2 \in \Sigma_G$; c) $z_3(sp_v)^{-1} \in \Sigma_G$.

2) Let (v_1, v_2) be an edge of G, $\xi(v_1) = x_1'$ and $\xi(v_2) = x_2'$. Then there exists $z_1, z_2 \in X^*$ such that at least one of the following statements is true: a) $sp_{v_1}x_2'z_1 \in \Sigma_G \bigcup \Lambda_G$; b) $sp_{v_2}x_1'z_2 \in \Sigma_G \bigcup \Lambda_G$.

3) If $\Sigma_G \neq \emptyset$, then for every $\sigma \in \Sigma_G$ there exist $p, q \in X^*$ such that $pq = \sigma$, and $p \in \mathcal{V}_G$ and $q^{-1} \in \mathcal{V}_G$.

4) Let (v_1, v_2) be an edge of the graph G that does not belong to the spanning tree $T(G, v_0)$. Then there exists a unique word $\sigma \in \Sigma_G$ such that the edge (v_1, v_2) is included in the path $v_0\sigma$.

5) If $\Sigma_G \neq \emptyset$, then for each $\sigma \in \Sigma_G$ inequality $d(\sigma) \geq 4$ is satisfied.

6) If $\Sigma_G \neq \emptyset$, then for each $\sigma \in \Sigma_G$ there are $p = p'x_1$ $(x_1 \in X)$, $q \in X^*$ $x_2, x_3 \in X$, $x_2 \neq x_3$ such that $\sigma = px_2qx_3p^{-1}$ and $v_0p = v_0px_2qx_3x_1$.

7) If v_1, v_2 are the generators of some word $\sigma \in \Sigma_G$, then $(v_1, v_2) \notin T(G, v_0)$ and $\mid d(sp_{v_1}) - d(sp_{v_2}) \mid \leq 1$.

In other words, the first and the second statements state that the pair $\{\Sigma_G, \Lambda_G\}$ contains certain information about each vertex and edge of G, the third statement states that each $\sigma \in \Sigma_G$ can be divided into two parts such that the first part and the reverse of the second part are the shortest by $\preceq$ words for the corresponding vertices of G, the fourth statement indicates that every edge of G not belonging to the spanning tree $T(\mathcal{V}_G)$ is described by a some word from Σ_G, the fifth statement shows that each $\sigma \in \Sigma_G$ contains a simple cycle; which, as the sixth statement establishes, cannot be presented as pxp^{-1} for any $p \in X^*$, $x \in X$.

From the AP-algorithm and the AC-algorithm, we can see that the pair $\{\Sigma_G, \Lambda_G\}$ is correct.

Theorem 2. $G(\{\Sigma_G, \Lambda_G\}) \cong G$.

To prove this theorem, we investigated the step-by-step construction of the graph $G(\{\Sigma_G, \Lambda_G\})$ by words from the sets Σ_G and Λ_G. To do this, let's $\Sigma_G = \{\sigma_1, \ldots, \sigma_k\}$ and $\Lambda_G = \{\lambda_1, \ldots, \lambda_q\}$. Let's introduce the family of sets M_i $(0 \le i \le k+q)$ as follows: $M_0 = \varnothing$, $M_1 = \{\sigma_1\}$, $M_2 = \{\sigma_1, \sigma_2\}$, …, $M_k = \{\sigma_1, \ldots, \sigma_k\} = \Sigma_G$, $M_{k+1} = \{\sigma_1, \ldots, \sigma_k, \lambda_1\}$, …, $M_{k+q} = \{\sigma_1, \ldots, \sigma_k, \lambda_1, \ldots, \lambda_q\} = \Sigma_G \bigcup \Lambda_G$. Given this family of sets, two families of graphs are introduced: G_i and H_i. The graphs of the family G_i are subgraphs of the graph G, defined by the vertices and edges of the words from the corresponding set M_i, and the graphs of the family H_i are constructed from the words from the corresponding set M_i using the AP-algorithm. According to this algorithm, $H_{k+q} = G(\{\Sigma_G, \Lambda_G\})$, and the equality $G_{k+q} = G$ follows from the first and the second statements of Theorem 1. By induction on i $(0 \le i \le k+q)$, we show that $H_i \cong G_i$.

4. Metric properties of the components of a canonical defining pair

Let's further assume that the D-graph $G = (V, E, X, \xi(V), v_0)$ belongs to the class $\mathcal{G}_{n,m}$. This section presents estimates of the power (i.e., in our case, the number of elements) and volume (i.e., in our case, the sum of the lengths of all words) of each component of a canonical defining pair: $|\Sigma_G|, |\Lambda_G|, \|\Sigma_G\|, \|\Lambda_G\|$. Hereafter, the notation $\lceil x \rceil$ denotes the smallest natural number, that is not less than x.

Theorem 3. $|\Sigma_G| = m - n + 1$.

Proof. Since each member of the class $\mathcal{G}_{n,m}$ is a connected simple graph, $n-1 \le m \le \frac{n(n-1)}{2}$. The proof will be done by induction on the number of edges.

Let $m = n-1$. Then any member of the class $\mathcal{G}_{n,n-1}$ is a tree, i.e., it does not contain cycles, which implies that $|\Sigma_G| = 0$. In this case, $n-1-n+1 = 0$, which proves the base of induction.

Suppose that for any member G of the class $\mathcal{G}_{n,k}$ $(n-1 \le k < \frac{n(n-1)}{2})$, $|\Sigma_G| = k-n+1$. We prove that for any member G' of the class $\mathcal{G}_{n,k+1}$ it is satisfied $|\Sigma_{G'}| = k-n+2$.

Let $G' = (V, E, X, \xi(V), v_0) \in \mathcal{G}_{n,k+1}$. Since $n \le k+1$, the D-graph G' contains at least one simple cycle. Let $(v_1, v_2) \in E$ belongs to some simple cycle of G' and does not belong to the spanning tree $T(G', v_0)$. Then the graph $G = G' - (v_1, v_2)$ [13] is connected and contains k edges. It is easy to see that this graph is a D-graph. By inductive assumption, $|\Sigma_G| = k-n+1$.

Let $p_1, p_2 \in \mathcal{V}_G$ and $v_0 p_1 = v_1$ and $v_0 p_2 = v_2$. Then, since the vertices v_1 and v_2 are not adjacent in the graph G, neither of two words $p_1 (p_2)^{-1}$ and $p_2 (p_1)^{-1}$ do not belong to Σ_G. In this case, according to the AC-algorithm, one of two words $p_1 (p_2)^{-1}$ or $p_2 (p_1)^{-1}$ (lesser in the order $\preceq$) belongs to $\Sigma_{G'}$. Let $p_1 (p_2)^{-1} \in \Sigma_{G'}$.

Since the edge $(v_1, v_2) \notin T(G', v_0)$, $\mathcal{V}_{G'} = \mathcal{V}_G$, so, according to the AC-algorithm, $\Sigma_G = \Sigma_{G'} \setminus \{p_1 (p_2)^{-1}\}$, so $|\Sigma_{G'}| = (k-n+1)+1 = k-n+2$, which proves the inductive step. □

Theorem 4. *If $m = n-1$ (i.e., G is a tree), then $1 \le |\Lambda_G| \le n-1$. If $m > n-1$, then $0 \le |\Lambda_G| \le n - \lceil \frac{3}{2} + \sqrt{\frac{9}{4} - 2 \cdot n + 2 \cdot m} \rceil$, and all estimates are attainable.*

Proof. Consider the first case, where G is a tree. G can contain from two (for a line graph [13]) to $n-1$ (for a star graph [13]) leaf vertices. Since, according to the AC-algorithm, $|\Lambda_G|$ is equal to the number of leaf vertices in the graph G that are different from the root vertex v_0, the largest value of $|\Lambda_G|$ is in the star graph, where the central vertex is the root vertex (Figure 2 a), and this value is $n-1$, and the smallest value of $|\Lambda_G|$ is in the line graph, where the root vertex is one of the two leaf vertices (Figure 2 b), and this value is 1, which proves the first part of the theorem.

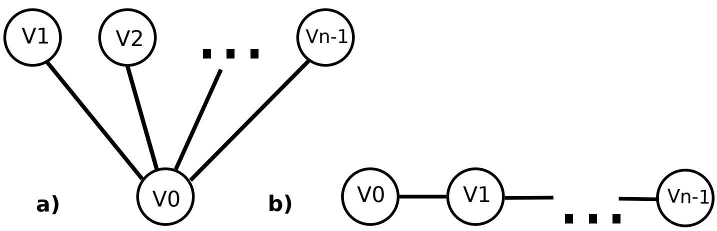

Figure 2. The trees for which the maximum and minimum estimates $|\Lambda_G|$ are attainable

Consider the case when G is not a tree, i.e. $n-1 < m \le \frac{n \cdot (n-1)}{2}$. In this case, there always exists a graph from the class $\mathcal{G}_{n,m}$ such that all vertices are the part of some simple cycle (i.e., there are no leaf vertices in G), so the smallest value of $|\Lambda_G|$ is 0.

Find an upper bound of $|\Lambda_G|$. Let's describe the structure of a graph G from the class $\mathcal{G}_{n,m}$, for which the number of leaf vertices is maximized. Let $|B(G)|$ vertices belonging to the base of G. It is easy to see that the maximum number of leaf vertices in G is $n - |B(G)|$, which is possible in the case when all free vertices of the graph are leaves (for example, all free vertices of such a graph are adjacent to the root vertex). Denote by $\delta(n,m)$ the minimum possible number of vertices for $G \in \mathcal{G}_{n,m}$ belonging to $B(G)$. Then the number $n - \delta(n,m)$ is an upper bound of $|\Lambda_G|$.

Let's find $\delta(n,m)$ for $G \in \mathcal{G}_{n,m}$. It is easy to see that the largest power of the first component of the canonical defining pair for a graph G' with y vertices is that of the complete graph K_y. By Theorem 3, $|\Sigma_G| = m - n + 1$, so $\delta(n,m)$ is equal to the minimum value of y such that $m - n + 1$ does not exceed $|\Sigma_{K_y}|$. Since K_y has $\frac{y\cdot(y-1)}{2}$ edges, then $|\Sigma_{K_y}| = \frac{y\cdot(y-1)}{2} - y + 1 = (\frac{y}{2} - 1)\cdot(y-1)$. In other words, the value of $\delta(n,m)$ is the minimal natural number y that satisfies the inequality $m - n + 1 \leq (\frac{y}{2} - 1)\cdot(y-1)$. Since m, n and y are natural numbers, $\delta(n,m) = \lceil \frac{3}{2} + \sqrt{\frac{9}{4} - 2\cdot n + 2\cdot m}\rceil$, so the upper bound of $|\Lambda_G|$ is $n - \lceil \frac{3}{2} + \sqrt{\frac{9}{4} - 2\cdot n + 2\cdot m}\rceil$. □

Below we provide all the estimates of the volume of the components of the canonical defining pair found to date.

Theorem 5. *Let $G \in \mathcal{G}_{n,m}$ be a tree. Then:*

1) $\|\Sigma_G\| = 0$*;*

2) $n \leq \|\Lambda_G\| \leq \lceil \frac{n^2+2n}{4} \rceil$*, and these estimates are attainable.*

Proof. 1) Since G is a tree, $\Sigma_G = \emptyset$, so $\|\Sigma_G\| = 0$.

2) Given the second statement of Theorem 1, for every edge (v_1, v_2) of G, there must exist a word $\lambda \in \Lambda_G$ such that (v_1, v_2) is in the path $v_0\lambda$. Since a path length of k edges is $k+1$, the smallest possible total number of symbols in Λ_G required to include all $n-1$ edges of G in the words of Λ_G is $n - 1 + 1$, which proves the lower bound. This estimate is attainable for the line graph shown in Figure 2 (b).

Let's prove the upper bound. Suppose that a graph G has q leaf vertices other than the root vertex v_0, and, accordingly, $|\Lambda_G| = q$ $(q = \overline{1, \ldots, n-1})$. It is easy to see that the maximum possible volume of the second component of the canonical defining pair will be for a tree with at most one vertex of degree greater than 2 (v'). This vertex can either coincide with the root vertex or not, then it is adjacent to some vertex that belongs to the shortest path from v_0 to v'. Also all leaf vertices of the tree other than the root vertex are necessarily adjacent to the vertex v'. The vertex v' is not adjacent to other vertices in the tree. An example of such a tree is shown in Figure 3, where vertex v' is adjacent to the vertex v'' and to all leaf vertices $\{v'_1, \ldots, v'_q\}$ other than the root vertex.

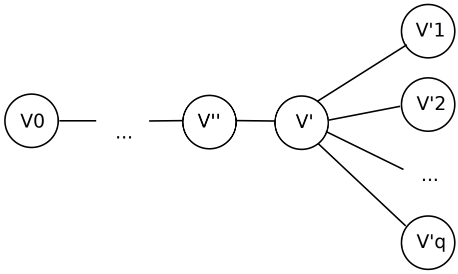

Figure 3. A tree for which the volume of the second component of the canonical defining pair is maximal

Let G be such a tree, we find $\|\Lambda_G\|$. Since $d(sp_{v'}) = n - q$, and the length of each word in Λ_G is $n-q+1$, $\|\Lambda_G\| = q(n-q+1)$. Thus, we need to find q ($q \in N$, $q \in [1, n-1]$) for which the function $f(q) = q(n-q+1)$ takes the largest value. Without taking into account the naturalness of q, we get $q = \frac{n+1}{2}$, taking into account $q \in N$, we get that the desired value is $q = \frac{n+1}{2}$ for odd n and $q = \frac{n}{2}$ or $q = \frac{n}{2} + 1$ for even n. Then $\|\Lambda_G\| = \frac{n^2+2n+1}{4}$ for odd n and $\|\Lambda_G\| = \frac{n^2+2n}{4}$ for even n. Combining these cases, we get $\|\Lambda_G\| = \lceil \frac{n^2+2n}{4} \rceil$, which proves the upper bound. □

Lemma 2. *Let $G \in \mathcal{G}_{n,m}$ is not a tree. Then $\|\Sigma_G\| \geq 4(m-n+1)$, and this estimate is attainable.*

Proof. This bound follows directly from Theorem 3 and the fifth statement of Theorem 1. This bound is attainable for a graph with all vertices adjacent to the root vertex. In such graph, Σ_G consists of $m-n+1$ words of the length 4. □

Let's consider an upper bound on the volume of the first component of the canonical defining pair for a graph that is not a tree. Let's consider the following graph $F \in \mathcal{G}_{n,m}$, which, due to its certain resemblance to a flower, we call the "flower graph", and its structural elements are given corresponding biological names.

When finding the upper bound of $|\Lambda_G|$ in Theorem 4, we found $\delta(n,m) = \lceil \frac{3}{2} + \sqrt{\frac{9}{4} - 2 \cdot n + 2 \cdot m} \rceil$, which is the minimum number of vertices in a connected simple graph G, such that $|\Sigma_G| = m - n + 1$. We divide all vertices of the graph F into two classes. The first class (the inflorescence) consists of vertices that are generators for words in Σ_F, the inflorescence contains $\delta(n,m)$ vertices. The second class (the stem) contains the remaining $n-\delta(n,m)$ vertices, in the case $n=\delta(n,m)$ all vertices of the graph F are part of the inflorescence. If $n>\delta(n,m)$, then the stem is a line with one end being the root vertex and the other end connected by an edge to one of the vertices of the inflorescence (let's call this vertex a receptacle). There are no other edges between the stem and the inflorescence. If $n=\delta(n,m)$, then the root vertex becomes the receptacle. The inflorescence can be a complete graph $K_{\delta(n,m)}$, then $\mu(n,m)=0$. Otherwise, $\mu(n,m)$ denotes the number of edges whose addition to the inflorescence makes it a complete graph.

Let's find $\mu(n,m)$. The complete graph $K_{\delta(n,m)}$ contains $\frac{\delta(n,m)(\delta(n,m)-1)}{2}$ edges, the stem contains $n-\delta(n,m)$ edges, and the graph F contains m edges. Thus $\mu(n,m)=\frac{\delta(n,m)(\delta(n,m)-1)}{2}-(m-n+\delta(n,m))$. If $\mu(n,m)>0$, then there are no edges between the receptacle and some $\mu(n,m)$ other vertices of the inflorescence, and there is an edge between any other vertices in the inflorescence. Let's illustrate the above definitions in Figure 4.

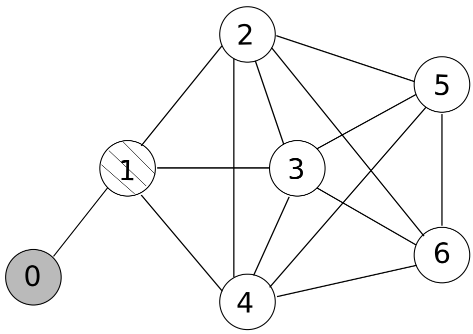

Figure 4. The flower graph $F\in\mathcal{G}_{7,14}$

For the flower graph shown in Figure 4, $n=7$, $m=14$, $\delta(n,m)=$ $=6$, $\mu(n,m)=2$, the stem consists of the root vertex with label 0, the inflorescence includes vertices with labels 1, ..., 6, the receptacle is the vertex with label 1, and the inflorescence lacks edges (1, 5) and (1, 6).

Let's find $\|\Sigma_F\|$ for a flower graph $F\in\mathcal{G}_{n,m}$. The length of the shortest path from the root vertex to the receptacle is $n-\delta(n,m)+1$. Consider two classes of vertices in the inflorescence. The class I includes those vertices of the inflorescence that are adjacent to the receptacle, and the class II includes those that are not adjacent to the receptacle (the receptacle does not belong to either class). The number of vertices of the class II is $\mu(n,m)$, the length of the shortest path to these vertices is

$n-\delta(n,m)+3$. The number of vertices of the class I is $\delta(n,m)-\mu(n,m)-1$, the length of the shortest path to these vertices is $n-\delta(n,m)+2$.

Let's consider how the pairs of generators in the flower graph F are formed. It is easy to see that the vertices that form the stem and the receptacle aren't generators for any word in Σ_F. All possible pairs of vertices in the class I are generators for the words in Σ_F. The number of such pairs is $\binom{\delta(n,m)-\mu(n,m)-1}{2}$, so the sum of the lengths of words from Σ_F formed by such vertices is

$$\frac{(\delta(n,m)-\mu(n,m)-1)(\delta(n,m)-\mu(n,m)-2)}{2}2(n-\delta(n,m)+2)=$$

$$=(\delta(n,m)-\mu(n,m)-1)(\delta(n,m)-\mu(n,m)-2)(n-\delta(n,m)+2).$$

All possible pairs of vertices in the class II are generators for the words in Σ_F. The number of such pairs is $\binom{\mu(n,m)}{2}$, so the sum of the lengths of words from Σ_F formed by such vertices is

$$\frac{\mu(n,m)(\mu(n,m)-1)}{2}2(n-\delta(n,m)+3)=$$

$$=\mu(n,m)(\mu(n,m)-1)(n-\delta(n,m)+3).$$

In addition, each vertex of the class II forms a pair of generators with any but one vertex of the class I, so the length of the words from Σ_F formed in this way is

$$\mu(n,m)(\delta(n,m)-\mu(n,m)-2)(2n-2\delta(n,m)+5).$$

Consequently,

$$\|\Sigma_F\|=(\delta(n,m)-\mu(n,m)-1)(\delta(n,m)-\mu(n,m)-2)(n-\delta(n,m)+2)+$$

$$+\mu(n,m)(\mu(n,m)-1)(n-\delta(n,m)+3)+$$

$$+\mu(n,m)(\delta(n,m)-\mu(n,m)-2)(2n-2\delta(n,m)+5).$$

Currently, we are investigating the upper bound on the volume of the first component of the canonical defining pair for any graph. The largest such estimate we have found so far is for a flower graph. We have developed the software prototypes of the algorithms proposed in this paper. These prototypes investigate whether the pair $\{C,L\}(x')$, which is explicitly specified or automatically generated according to certain rules, is correct. If the pair $\{C,L\}(x')$ is correct, then is constructing the graph $G(\{C,L\}(x'))$ and is finding all the metric characteristics of the canonical defining pair of the graph $G(\{C, L\}(x'))$ given in the paper. The tests did not deny the hypothesis that the largest upper bound on the volume of the first component of the canonical defining pair is that of the flower graph.

Conclusion

The paper proposes the presentation of D-graphs and SD-graphs by a defining pair of words. We present an algorithm which, given an arbitrary pair of sets satisfying some conditions, either constructs a D-graph for which this pair is defining, or reports that it is impossible. We give a criterion by which this graph will be a SD-graph. We also present an algorithm that constructs a canonical defining pair for a D-graph and find some numerical estimates of it. The given representation of D-graphs and SD-graphs can be useful for solving various problems, including applied ones, among which the authors emphasize the following:

1) Finding relationships between the graph G and its reduction $[G]$.

2) Solving the problem of the pair characterization: for a given D-graph and a given pair, determine whether this pair is defining for the graph without directly constructing a graph from the pair.

3) Finding the necessary and sufficient conditions for the sets C and L, under which the pair $\{C, L\}(x')$ is correct.

4) An optimal choice of the root vertex, according to which the metric properties of the components of the canonical defining pair will be minimal.

5) An efficient construction of the shortest paths between two arbitrary vertices of the graph G by $\{\Sigma_G, \Lambda_G\}$.

References

[1] U. Knauer, M. Nieporte, *Endomorphisms of graphs I. The monoid of strong endomorphisms*, Archiv der Mathematik, **52**, 1989, pp. 607–614. DOI: 10.1007/BF01237575.

[2] Y. V. Zhuchok, *The monoid of endomorphisms of disconnected hypergraphs*, Algebra and Discrete Mathematics, **16** (1), 2013, pp. 134–150. https://admjournal.luguniv.edu.ua/index.php/adm/article/view/763/292.

[3] J.-L. Loday, M. O. Ronco, *Trialgebras and families of polytopes*, Contemporary Mathematics, **346**, 2004, pp. 369–398. DOI: 10.48550/arXiv.math/0205043.

[4] A. V. Zhuchok, Yul. V. Zhuchok, Y. V. Zhuchok, *Certain congruences on free trioids*, Communications in Algebra, **47** (12), 2019, pp. 5471–5481. DOI: 10.1080/00927872.2019.1631322.

[5] Y. V. Zhuchok, J. Koppitz, *Representations of ordered doppelsemigroups by binary relations*, Algebra and Discrete Mathematics, **27** (1), 2019, pp. 144–154. https://admjournal.luguniv.edu.ua/index.php/adm/article/view/1294/pdf.

[6] I. S. Grunskii, A. S. Senchenko, *Properties of systems of defining relations for automata*, Discrete Mathematics and Applications, **14** (6), 2004, pp. 593–601. DOI: 10.1515/1569392043272458.

[7] I. Grunskii, I. Mikhaylova, S. Sapunov, *Domination on the vertices of labeled graphs*, Algebra and Discrete Mathematics, **14** (2), 2012, pp. 174–184. https://admjournal.luguniv.edu.ua/index.php/adm/article/view/719/251.

[8] R. Gorrieri, *Process Algebras for Petri Nets. The Alphabetization of Distributed Systems*, Springer, 2017. DOI: 10.1016/C2015-0-04158-2.

[9] M. Droste, W. Kuich, H. Vogler, *Handbook of Weighted Automata*, Springer, 2009. DOI: 10.1007/978-3-642-01492-5.

[10] G. Dudek, M. Jenkin, *Computational Principles of Mobile Robotics, 2nd ed.*, Cambridge Univ. Press, 2010. DOI: 10.1017/CBO9780511780929.

[11] C. Baier, J.-P. Katoen, *Principle of Model Checking*, MIT Press, 2008. DOI: 10.5555/1373322.

[12] S. V. Sapunov, A. S. Senchenko, *Linguistic representation of vertex-labeled graphs*, Reports of the National Academy of Sciences of Ukraine, **11**, 2019, pp. 17–24. DOI: 10.15407/dopovidi2019.11.017.

[13] R. Diestel, *Graph Theory. Fifth Edition*, Springer, 2017. DOI: 10.1007/978-3-662-53622-3.

Contact information

Oleksii S. Senchenko

Institute of Applied Mathematics and Mechanics of the National Academy of Sciences of Ukraine
E-Mail: senchenko.a76@gmail.com
URL:

Mykola I. Prytula

Institute of Applied Mathematics and Mechanics of the National Academy of Sciences of Ukraine
E-Mail: elanir358@gmail.com
URL:

Received by the editors: 12.09.2023
and in final form 17.12.2023.