Linguistic presentation of deterministic and strongly deterministic graphs

Oleksii S. Senchenko, Mykola I. Prytula

IAMM of the NAS of Ukraine, Slovyansk
November, 2023

Supported by the Grant EFDS-FL2-08 of The European Federation of Academies of Sciences and Humanities (ALLEA)

Motivations

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 presentation theory, there are various problems, many of which have significant application value.

In the research of the linguistic presentation of finite automata without output, 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.

Motivations

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, including their subclass, strongly deterministic graphs, for the following reasons:

- 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, weighted automata, finite automata etc). There are many computational processes in programming, robotics, model verification, which are naturally presented by graphs with labeled vertices, including deterministic graphs.
- Such graphs have a certain resemblance to finite automata with no output.
- 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.

Results

1. We propose a linguistic 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.
2. We propose an algorithm (AP-algorithm) that, given a pair of word sets that meet certain conditions, either construct a strongly deterministic graph or reports that it is impossible to do so (in other words, the pair of sets is not correct).
3. 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.
4. Similarly to the canonical system of defining relations, we propose a special defining pair, which is called canonical.
5. 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.
6. We also propose subclass of strongly deterministic graphs for which the volume of the first component of the canonical defining pair is most likely to be maximal.

Main definitions

We consider undirected, finite, non-empty, connected, vertex-labeled simple graphs $G = (V, E, X, \xi(V))$, 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 $X = \{x_1, \ldots, x_p\}$. 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 $p = x'_1 \ldots x'_k$ $(x'_i \in X)$, 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 leaf.

We consider deterministic and strongly deterministic graphs. A labeled graph G is called deterministic (or D-graph) if all vertices in open neighborhood of every its vertex have different labels. A deterministic graph G is called strongly deterministic (or SD-graph) if all vertices in closed neighborhood of every its vertex have different labels. In other words, a deterministic graph can have two adjacent vertices with the same label, but this cannot happen in an strongly deterministic graph.

Main definitions

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: $G = (V, E, X, \xi(V), v_0)$. Since the path $p = v'_1 \dots v'_k$ uniquely corresponds to a 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 word) $p = x'_1x'_2 \dots x'_k$ is called valid for 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}.

All the 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.

Result 1 - defining pair for D-graphs and SD-graphs

It is desirable that such a presentation satisfies the following conditions:

- The presentation of graphs should be similar to the presentation of automata.
- 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 sets C and L are called 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.

Result 1 - defining pair for D-graphs and SD-graphs

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 both components of a pair $\{C, L\}(x')$, is considered a presentation of this pair:

- $\xi(v_0) = (x')$.
- All words from C and L are valid for v_0.
- Each word $p \in C$ describe the cycle $v_0 p = v_0$ in G.
- For each word $q \in L$ the vertex $v_0 q$ is leaf in G.
- 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 $v_0 q = v$.
- 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.

Result 2 - linear order $\preceq$

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 $1 \leq i \leq p$, 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 $p = x'_1 \dots x'_s$, $q = x''_1 \dots x''_s$, $x'_1 = x''_1$, $\dots$, $x'_{k-1} = x''_{k-1}$ and $x'_k < x''_k$, then $p \preceq q$.

Result 2 - AR-algorithm

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 (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:

Result 2 - AR-algorithm

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 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 $[G] = G'$.

Result 2 - AP-algorithm

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 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 AR-algorithm.

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

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

Result 2

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 and 4 are successful, is the graph $G(\{C, L\}(x'))$. If at least one of the checks in stages 3 and 4 fails, then we assume that the graph $G(\{C, L\}(x'))$ does not exist.

It is easy to see that if the pair $\{C, L\}(x')$ is correct, the graph $G(\{C, L\}(x'))$ satisfies requirements (a) – (e), any proper subgraph of $G(\{C, L\}(x'))$ does not satisfy requirement (b), and any proper supergraph of $G(\{C, L\}(x'))$ satisfies requirements (a) – (c), but may not satisfy 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 $G(\{C, L\}(x')) \cong G$.

Result 2 - example

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 $v_0152125423$ and v_014523 are leaves, so the check in step 3 is successful. For the selected vertex v with label 1, there is no word $p \in L$ such that $v_0p = v$, 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).

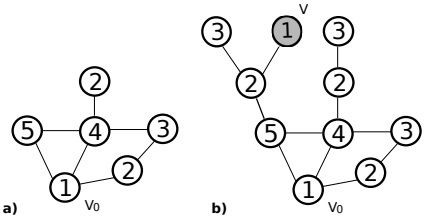

Figure 1. Illustration of the procedure for constructing a D-graph for a given pair

Result 3

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

Result 4 - 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 $v_0 sp_v = v$.

Result 4 - AC-algorithm

Let's describe the algorithm (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 $v_0 p$ and $v_0 q^{-1}$ are called the generators for the word pq^{-1} or qp^{-1} that was added to Σ_G).

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

Result 5 - metric properties of the components of a canonical defining pair

Let's further assume that the D-graph $G = (V, E, X, \xi(V), v_0)$ has n vertices and m edges. 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. $|\Sigma_G| = m - n + 1$.

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

Theorem 5. Let G 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.

Theorem 6. Let G is not a tree. Then $\|\Sigma_G\| \geq 4(m - n + 1)$, and this estimate is attainable.

Result 6 - flower graph

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 has n vertices and m edges, which we call the «flower graph».

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 the vertices of the graph F into two classes. The first class (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. 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 (we call this vertex in the inflorescence the 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.

Result 6 - flower 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 2.

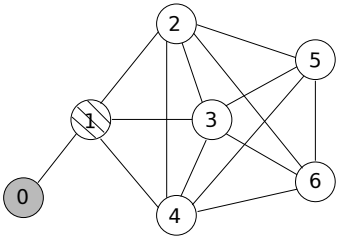

Figure 2. Flower graph F, that has 7 vertices and 14 edges.

Result 6 - flower graph

For the flower graph shown in Figure 2, $n = 7$, $m = 14$, $\delta(n, m) = 6$, $\mu(n, m) = 2$, the stem consists of the root vertex 0, the inflorescence includes vertices 1, ..., 6, the receptacle is the vertex 1, and the inflorescence lacks edges (1, 5) and (1, 6).
Let's find $\|\Sigma_F\|$ for a flower graph F, that has n vertices and m edges.

Theorem 7. Let F is flower graph with n vertices and m edges. Then:

$$\|\Sigma_F\| = (\delta(n, m) - \mu(n, m) - 1)(\delta(n, m) - \mu(n, m) - 2)(n - \delta(n, m) + 2) +$$
$$+\mu(n, m)(\mu(n, m) - 1)(n - \delta(n, m) + 3) +$$
$$+\mu(n, m)(\delta(n, m) - \mu(n, m) - 2)(2n - 2\delta(n, m) + 5).$$

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 software prototypes of the algorithms proposed in this presentation. 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.

Publications

O.S. Senchenko, M.I. Prytula, *The presentation of deterministic and strongly deterministic graphs*, Algebra and Discrete Mathematics, **36**, 2023 - Submitted.

Thank you for your attention!