On Algebra of Languages Representable by Vertex-Labeled Graphs

Igor Grunsky [a,c] Igor Potapov [b] Elena Pryanichnikova [c]

[a] *Institute of Applied Mathematics and Mechanics, Ukrainian National Academy of Sciences, 74 R. Luxemburg St, Donetsk, Ukraine*

[b] *Department of Computer Science, University of Liverpool, Ashton Building, Ashton Street, Liverpool L69 3BX, U.K.*

[c] *The Institute of Artificial Intelligence, 84 B.Khmelnitsky, Donetsk, Ukraine*

Abstract

In this paper we introduce and study an algebra of languages representable by vertex-labeled graphs. The proposed algebra is equipped with three operations: the union of languages, the merging of languages and the iteration. In contrast to Kleene algebra, which is mainly used for edge-labeled graphs, it can adequately represent many properties of languages defined by vertex-labeled graphs and provides a natural translation from vertex-labeled graphs to regular expressions and vice versa.

Key words: Vertex-labeled graphs, formal languages, Kleene algebra.

1 Introduction

Graphs are among the most widely used structures in computer science for describing and modeling a variety of computational processes, and the most studied ones in this context are finite oriented graphs with labeled edges (also known as finite state automata).

This paper is devoted to the study of oriented graphs with labeled vertices, which can be seen as a dual class in relation to the class of finite state au-

⋆ The work of authors was supported in part by NATO Collaborative Linkage Grant 983162.

Email addresses: grunsky@iamm.ac.donetsk.ua (Igor Grunsky), potapov@liverpool.ac.uk (Igor Potapov), ginger701@mail.ru (Elena Pryanichnikova).

tomata. There are many examples where computational processes can be translated more naturally into vertex-labeled graphs rather than into edge-labeled graphs. For example, in programming such graphs are known as flowcharts that represent an algorithm or a computational process [8]. In robotics vertex-labeled graphs are used to describe a topological environment navigation problems. One of such problems is the map validation problem: "Given an input map (described by a vertex-labeled graph) and the current position of a robot, determine by the robot walking on the graph whether this map is correct". Another problem is the self-location problem: "Given only a map of the environment, determine the position of the robot (i.e., the correspondence between edges of the map and edges in the world at the robot's position) [5]". Vertex-labeled graphs have been widely used in model checking to represent the behaviour of a system and are known as Kripke structures. In this case the vertices of a graph represent the reachable states of the system, the edges represent state transitions and a labeling function maps each vertex to a set of properties that hold in the corresponding state [1,2].

Languages defined by vertex-labeled graphs have been already studied in the past. In particular, the characterization of languages that can be represented by different types of vertex-labeled graphs was given in [8,7] and the problem of vertex minimization in a graph preserving its language was studied in [10].

In this work the class of languages representable by vertex-labeled graphs is studied by introducing an algebra similar to the well-known Kleene algebra. The proposed algebra is equipped with three operations: the union, the merging and the iteration. In contrast to Kleene algebra it can adequately represent many properties of languages defined by vertex-labeled graphs and it provides a natural translation from vertex-labeled graphs to regular expressions and vice versa.

The paper is organized as follows. In Section 2 we provide the main definitions, introduce a new algebra and investigate its properties. In Section 3 we show how to construct for any vertex-labeled graph a regular expression representing the same language. In Section 4 we introduce the matrix representation of vertex-labeled graphs to prove that for any regular expression we can construct a graph representing the same language.

2 Basic definitions

Let X be a finite alphabet, X^+ be the set of all non-empty finite words over X and 2^{X^+} be the set of all languages over X not containing the empty word. Let us denote the empty set by $\varnothing$ and the concatenation of two words $u \in X^+$, $v \in X^+$ by uv. The concatenation of two languages $L, R \in 2^{X^+}$ is denoted by

$L \cdot R$ or LR and it is defined as $L \cdot R = \{uv \mid u \in L, v \in R\}$.

The partial binary operation $\circ$ of merging two words is defined on the set X^+ as follows:

$$w_1 x \circ y w_2 = \begin{cases} w_1 x w_2, & \text{if } x = y; \\ \text{undefined}, & \text{otherwise} \end{cases}$$

for any $w_1, w_2 \in X^+$ and any $x, y \in X$.

Definition 1 *For any L, $R \subseteq X^+$, we define*

(1) $L \cup R = \{w \mid w \in L \text{ or } w \in R\}$;
(2) $L \circ R = \{w_1 \circ w_2 \mid w_1 \in L,\ w_2 \in R\}$;
(3) $L^+ = \bigcup_{i=1}^{\infty} L^i$, *where* $L^1 = L$; $L^{n+1} = L^n \circ L$ *for all* $n \geq 1$;
(4) $L^* = \bigcup_{i=0}^{\infty} L^i$, *where* $L^0 = X$; $L^{n+1} = L^n \circ L$ *for all* $n \geq 0$;
(5) $L^{\circledast} = \bigcup_{i=0}^{\infty} L^i$, *where* $L^0 = L_{beg} \circ L_{end}$; $L^{n+1} = L^n \circ L$ *for all* $n \geq 0$, $L_{beg} = \{x \in X \mid xw \in L \text{ for some } w \in X^*\}$; $L_{end} = \{x \in X \mid wx \in L \text{ for some } w \in X^*\}$.

The algebra $\langle 2^{X^+}, \circ, X\rangle$ is a monoid, since the operation $\circ$ is associative and the equalities $X \circ L = L \circ X = L$ hold for any language $L \subseteq X^+$. The operation $\cup$ is the union. Obviously, $\langle 2^{X^+}, \cup, \varnothing\rangle$ is an idempotent commutative monoid. The operation $\circ$ is distributive over $\cup$ both from the left and from the right. From the above definition it follows that the algebra $\langle 2^{X^+}, \cup, \circ, \varnothing, X\rangle$ is an idempotent semiring.

The operations $*$ and $+$ are not exactly the same as the star operator and the plus operator in Kleene algebra of regular languages [6]. In particular, for the usual star a set L^* is infinite for any $L \neq \emptyset$, but the result of applying our star operator for $L \neq \emptyset$ can be finite. The $\circledast$ is a variant of the star operator, obtained by setting $L^0 \subseteq X$. It is intended to represent the intuitive meaning of iteration for vertex-labeled graphs.

In the next proposition we folmulate some basic properties of the operations $*$, $+$ and $\circledast$. These facts will be used later in the paper and they are directly follow from the above definitions.

Proposition 2 *For any $L, R \subseteq X^+$, we have*

$$L^* = X \cup L^+ = (X \cup L)^+ \tag{1}$$

$$L^{\circledast} = L_{beg} \circ L_{end} \cup L^+ = (L_{beg} \circ L_{end} \cup L)^+ \tag{2}$$

$$L^* = L^* \circ L^* \tag{3}$$

$$L^{\circledast} = L^{\circledast} \circ L^{\circledast} \cup L \tag{4}$$

$$L^+ = L \circ L^* = L^* \circ L \tag{5}$$

$$L^+ = L^{\circledast} \circ L \cup L = L \circ L^{\circledast} \cup L \tag{6}$$

$$(R \cup L)^* = R^* \circ (L \circ R^*)^* \tag{7}$$

$$(R \circ L)^* \circ R = R \circ (L \circ R)^* \tag{8}$$

$$L^{\circledast} = L_{beg} \circ (L^*) \circ L_{end} \tag{9}$$

The main property of the algebra $\langle 2^{X^+}, \cup, \circ, \circledast, \varnothing, X \rangle$ is formulated in the following Lemma.

Lemma 3 *For any $R, Q \subseteq X^+$, if $X \cap R = \varnothing$ then the equation $Y = R \circ Y \cup Q$ has the unique solution $Y = R^{\circledast} \circ Q \cup Q$ in the algebra $\langle 2^{X^+}, \cup, \circ, \circledast, \varnothing, X \rangle$.*

Proof. It is straightforward to verify that the language $R^{\circledast} \circ Q \cup Q$ is indeed a solution.

The proof of the uniqueness will be based on the fixed point theorem which states: an equation $x = f(x)$, where f is a continuous function from any complete partially ordered set (CPO) to itself, has a minimal solution [4].

An ordered set $P = (P, \leq)$ is a CPO if P has a bottom element and $sup(D)$ exists for each directed subset D of P[8]. A function $f : P \to Q$ is continuous if it preserves all existing suprema of directed sets[8]. The set $(2^{X^+}, \subseteq)$ is CPO since any directed sequence of languages $L_1 \subseteq L_2 \subseteq ... \subseteq L_n \subseteq ...$ has a supremum, which is the union of all languages in this sequence, and $\varnothing$ is the bottom element of $(2^{X^+}, \subseteq)$.

Due to the facts that the operations $\circ$ and $\cup$ are continuous and the composition of continuous functions is continuous, the right part of the equation $Y = R \circ Y \cup Q$ defines a continuous mapping $f : 2^{X^+} \to 2^{X^+}$, $f(L) = R \circ L \cup Q$.

By the fixed point theorem, the least fixed point of f is $Y = \bigcup_{n \geq 0}^{\infty} f^n(\varnothing)$ where $f^0(x) = x$ and $f^n(x) = f(f^{n-1}(x))$ for any $n > 0$.

Since $f^0(\varnothing) = \varnothing$, $f^1(\varnothing) = Q$, $f^2(\varnothing) = R \circ Q \cup Q$, $f^3(\varnothing) = R \circ R \circ Q \cup R \circ Q \cup Q$ (by distributive law), and generally for any $n \geq 2$ $f^n(\varnothing) = (R^{n-1} \cup R^{n-2} \cup ... \cup R \circ R \cup R) \circ Q \cup Q$ we have the following

$$\begin{aligned} Y = \bigcup_{n \geq 0}^{\infty} f^n(\varnothing) = \bigcup_{n \geq 1}^{\infty} ((R^{n-1} \cup R^{n-2} \cup ... \cup R \circ R \cup R)) \circ Q \cup Q) = \\ = R^+ \circ Q \cup Q. \end{aligned} \tag{10}$$

For any languages P and R,

$$P \subseteq X \text{ implies } P \circ R \subseteq R \tag{11}$$

Since $R_{beg}, R_{end} \subseteq X$, it follows by (11) that $R_{beg} \circ R_{end} \subseteq X$, and hence that

$$R^+ \circ Q \cup Q = R^+ \circ Q \cup Q \cup R_{beg} \circ R_{end} \circ Q = (R^+ \cup R_{beg} \circ R_{end}) \circ Q \cup Q \tag{12}$$

By (2) we have that for any language

$$R^{\circledast} = R^{+} \cup R_{beg} \circ R_{end} \tag{13}$$

Applying (13) we can now express the minimal solution as

$$Y = R^{\circledast} \circ Q \cup Q. \tag{14}$$

Let us show that if $X \cap R = \varnothing$ then the solution $R^{\circledast} \circ Q \cup Q$ is unique.

Let Y' be a solution of the equation $Y = R \circ Y \cup Q$. Since $R^{\circledast} \circ Q \cup Q$ is the minimal solution, it follows that $R^{\circledast} \circ Q \cup Q \subseteq Y'$.

By a sequence of iterative substitutions we express Y' as follows:

$$\begin{aligned} Y' &= R \circ Y' \cup Q = R \circ (R \circ Y' \cup Q) \cup Q = \\ &= R \circ R \circ Y' \cup R \circ Q \cup Q = R \circ R \circ (R \circ Y' \cup Q) \cup R \circ Q \cup Q = \\ &= R^{m} \circ Y' \cup R^{m-1} \circ Q \cup ... \cup R \circ Q \cup Q. \end{aligned} \tag{15}$$

Consider a word $w \in Y'$ of the length m. Assuming that $X \cap R = \varnothing$, all words of the language R^{m} are at least of length $m+1$ and therefore $w \in R^{m-1} \circ Q \cup ... \cup R \circ Q \cup Q$ and hence $w \in R^{\circledast} \circ Q \cup Q$ and $Y' \subseteq R^{\circledast} \circ Q \cup Q$. Thus $Y' = R^{\circledast} \circ Q \cup Q$ and $R^{\circledast} \circ Q \cup Q$ is the only solution of the equation $Y = R \circ Y \cup Q$. $\square$

Definition 4 *The regular $\circ$-expressions over an alphabet X and the language $L(P)$ represented by such an expression P are defined inductively as follows:*

(1) $\varnothing$ is a regular $\circ$-expression and $L(\varnothing)$ is the empty set;
(2) any letter $x \in X$ is a regular $\circ$-expression and $L(x) = \{x\}$;
(3) if $x \in X$ and $y \in X$, then xy is a regular $\circ$-expression and $L(xy) = \{xy\}$;
(4) if R and Q are regular $\circ$-expressions, then $(R \circ Q)$, $(R \cup Q)$, $(R^{\circledast})$ are also regular $\circ$-expressions and $L(R \circ Q) = L(R) \circ L(Q)$, $L(R \cup Q) = L(R) \cup L(Q)$, $L(R^{\circledast}) = (L(R))^{\circledast}$.

Definition 5 *A regular $\circ$-expressions P is equal to a regular $\circ$-expressions Q if $L(P) = L(Q)$. The equality of regular $\circ$-expressions P and Q is denoted by $P = Q$.*

3 Representation of graph languages by regular $\circ$-expressions

Let us define a vertex-labeled graph as a quadruple $G = (V, E, X, \mu)$, where V is a finite set of vertices, $E \subseteq V \times V$ is a set of edges, X is a set of vertex labels and $\mu : V \rightarrow X$ is a mapping from vertices to labels.

A path in graph G is a finite sequence of adjacent vertices $l = v_1v_2...v_k$, where $(v_i, v_{i+1}) \in E$ for $i = 1, ..., k-1$; k is the length of the path; v_1 is the initial vertex of the path and v_k is the final vertex. The label of this path is $x = \mu(v_1)\mu(v_2)...\mu(v_k) = x_1x_2..x_k \in X^+$.

Let I and F are subsets of V, called the set of initial vertices and the set of final vertices, respectively. A path $v_1v_2...v_k$ is called successful if $v_1 \in I$ and $v_k \in F$.

The set of labels of all successful paths in a graph G form the language $L(G)$.

In automata theory one of the fundamental results is Kleene's theorem. It states that the class of languages recognized by finite automata coincides with the class of regular languages, which are given by regular expressions [6]. The main aim of this work is to prove a similar theorem for languages defined by vertex labeled graphs.

Theorem 6 *Any language defined by a vertex-labeled graph can be represented by a regular $\circ$-expression.*

Proof. Let us show that for any graph G there exists a regular $\circ$-expression R such that $L(R) = L(G)$.

Let L_i be the set of labels for all paths from a vertex v_i into final vertices in G. Then $L(G) = \bigcup_{v_i \in I} L_i$.

For any vertices v_i and v_j the language L_i contains all words of L_j extended by the left concatenation of symbol $\mu(v_i)$ for each edge $(v_i, v_j) \in E$. If the vertex v_i is a final vertex of G then the symbol $\mu(v_i)$ belongs to the language L_i. So each language L_i can be represented by the following equation:

$$L_i = M_{i1} \circ L_1 \cup M_{i2} \circ L_2 \cup ... \cup M_{in} \circ L_n \cup B_i, \tag{16}$$

where n is the number of vertices,

$$M_{ij} = \begin{cases} \mu(v_i)\mu(v_j), & \text{if } (v_i, v_j) \in E; \\ \emptyset, & \text{otherwise;} \end{cases}$$

$$B_i = \begin{cases} \mu(v_i), & \text{if } v_i \in F; \\ \emptyset, & \text{otherwise.} \end{cases}$$

Thus $L(G)$ can be derived by solving the following system of equations:

$$\begin{array}{l} L_1 = M_{11} \circ L_1 \cup M_{12} \circ L_2 \cup ... \cup M_{1n} \circ L_n \cup B_1 \\ L_2 = M_{21} \circ L_1 \cup M_{22} \circ L_2 \cup ... \cup M_{2n} \circ L_n \cup B_2 \\ \dots\dots\dots\dots \\ L_n = M_{n1} \circ L_1 \cup M_{n2} \circ L_2 \cup ... \cup M_{nn} \circ L_n \cup B_n. \end{array} \tag{17}$$

Since every equation of the system (17) has the form $Y = R \circ Y \cup Q$, we can solve the system of equations using the Lemma 3 as in the Kleene algebra $\langle 2^{X^*}, \cdot, \cup, *, \varnothing, \lambda \rangle$ [8]. Applying (14) we can eliminate L_n from the right part of the final equation in the system (17). Then the value of L_n can be substituted into the previous equation of the system and using the same technique to get a value for L_{n-1} that will be only expressed via $L_1 .... L_{n-2}$. By continuing such substitutions for all equations in the system we can get values for all L_i $(i = 2, ..., n)$ expressed via $L_1 .... L_{i-1}$. To get the value of L_i (which are regular $\circ$-expressions based on Lemma 3) one should substitute into it the final forms of L_1, L_2,..,L_{i-1}.

From the above it follows that by solving the equations (17) for any graph G we can find a regular $\circ$-expression R, such that $L(R) = L(G)$.

This ends the proof of the Theorem 6. □

4 Matrix representation of graph languages

Let us consider a family of matrices over the algebra $\langle 2^{X^+}, \circ, \cup, \circledast, \varnothing, X \rangle$. First we define the following two operations $\cup$ and $\circ$ on matrices:

(1) If A and B are $m \times n$ matrices, then $A \cup B = C$, where C is an $m \times n$ matrix and $C_{ij} = A_{ij} \cup B_{ij}$ for all $i, j = 1, ..., n$;

(2) If A is an $n_1 \times m$ matrix and B an $m \times n_2$ matrix, then $A \circ B = C$, where C is $n_1 \times n_2$ matrix and $C_{ij} = \bigcup_{k=1}^{m} A_{ik} \circ B_{kj}$.

Thus operations $\circ$ and $\cup$ are defined in a similar way as the standard matrix addition and multiplication, where addition and multiplication of values are understood as corresponding operations in algebra $\langle 2^{X^+}, \circ, \cup, \circledast, \varnothing, X \rangle$.

The transpose of an $m \times n$ matrix A is defined as the $n \times m$ matrix A^T where $[A^T]_{ij} = A_{ji}$. The zero matrix Z_n is defined as the square $n \times n$ matrix where all elements are equal to $\varnothing$. The identity matrix E_n is defined as the square $n \times n$ matrix where all elements on the main diagonal are equal to X and all other elements are equal to $\varnothing$.

Now we consider matrix representations for vertex-labeled graphs by analogy with matrix representations for finite state automata [9].

Let us represent a graph G with n vertices by an $n \times n$ transition matrix M and vectors s and t of length n such that

$$M_{ij} = \begin{cases} \mu(v_i)\mu(v_j), & \text{if } (v_i, v_j) \in E; \\ \varnothing, & \text{otherwise;} \end{cases}$$

$$s_i = \begin{cases} \mu(v_i), & \text{if } v_i \in I; \\ \varnothing, & \text{otherwise;} \end{cases}$$

$$t_i = \begin{cases} \mu(v_i), & \text{if } v_i \in F; \\ \varnothing, & \text{otherwise;} \end{cases}$$

Define the operation $\circledast$ for the transition matrix M as
$M^{\circledast} = M^0 \cup M \cup M \circ M \cup M \circ M \circ M \cup ...$ where

$$[M^0]_{ij} = \begin{cases} \mu(v_i), & \text{if } i = j; \\ \varnothing, & \text{if } i \neq j. \end{cases}$$

Using the matrix representation for graphs with labeled vertices we can define the system (17) as a matrix equation

$$l = M \circ l \cup t, \tag{18}$$

and represent the language defined by a graph G as $L(G) = s^T \circ l$. By lemma 3 $l = M^{\circledast} \circ t \cup t$. Since $M^0 \circ t = t$ for any graph G, $l = M^{\circledast} \circ t \cup M^0 \circ t = (M^{\circledast} \cup M^0) \circ t = M^{\circledast} \circ t$, $L(G) = s^T \circ M^{\circledast} \circ t$.

The element M_{ij} contains the label for a path of the length 2 from a vertex v_i into a vertex v_j; $[M \circ M]_{ij} = \bigcup_{k=1}^{n} M_{ik} \circ M_{kj}$ corresponds to the sum of all paths of length 3 from v_i into v_j; $[M \circ M \circ M]_{ij}$ is the sum of all paths of length 4, and so on; $[M^0]_{ij}$ contains the label for a path of the length 1 from v_i into v_j. Thus $M^{\circledast}$ is a matrix where each element $[M^{\circledast}]_{ij}$ is a regular $\circ$-expression which represents all possible paths in G from v_i into v_j, so the language defined by G can be represented as

$$L(G) = s^T \circ M^{\circledast} \circ t. \tag{19}$$

In order to compute the values of $M^{\circledast}$ we can use a recursive method that is similar to the one applied in [3] for transition matrices of finite state automata.

Recursive method

Given a graph G with n vertices.

(1) If $n = 1$, then obviously $M^{\circledast} = M_{11}^{\circledast}$;

(2) If $n > 1$, let us
   (a) take any value n_1 such that $0 < n_1 < n$,
   (b) split $M = \begin{pmatrix} M_{11} & \dots & M_{1n} \\ \vdots & \ddots & \vdots \\ M_{n1} & \dots & M_{nn} \end{pmatrix}$ into the blocks $\begin{array}{|c|c|} \hline M_1 & M_3 \\ \hline M_4 & M_2 \\ \hline \end{array}$ where

$$M_1 = \begin{pmatrix} M_{11} & \dots & M_{1n_1} \\ \vdots & \ddots & \vdots \\ M_{n_1 1} & \dots & M_{n_1 n_1} \end{pmatrix}, M_2 = \begin{pmatrix} M_{(n_1+1)(n_1+1)} & \dots & M_{(n_1+1)n} \\ \vdots & \ddots & \vdots \\ M_{n(n_1+1)} & \dots & M_{nn} \end{pmatrix},$$

$$M_3 = \begin{pmatrix} M_{1(n_1+1)} & \dots & M_{1n} \\ \vdots & \ddots & \vdots \\ M_{n_1(n_1+1)} & \dots & M_{n_1 n} \end{pmatrix}, M_4 = \begin{pmatrix} M_{(n_1+1)1} & \dots & M_{(n_1+1)n_1} \\ \vdots & \ddots & \vdots \\ M_{n1} & \dots & M_{nn_1} \end{pmatrix},$$

   (c) compute $M_1{}^{\circledast}$ and $M_2{}^{\circledast}$;
   (d) $M^{\circledast}$ is computed as

$(M_1 \cup M_3 \circ M_2{}^{\circledast} \circ M_4)^{\circledast}$	$(M_1 \cup M_3 \circ M_2{}^{\circledast} \circ M_4)^{\circledast} \circ M_3 \circ M_2{}^{\circledast}$
$M_2{}^{\circledast} \circ M_4 \circ (M_1 \cup M_3 \circ M_2{}^{\circledast} \circ M_4)^{\circledast}$	$M_2{}^{\circledast} \circ M_4 \circ (M_1 \cup M_3 \circ M_2{}^{\circledast} \circ M_4)^{\circledast} \circ M_3 \circ M_2{}^{\circledast} \cup M_2{}^{\circledast}$

Let us prove the correctness of this method. For the case $n > 1$ let us define the equation (18) using block matrices and block vectors. First let us split vectors l and t into blocks l^{top}, l^{bottom} and t^{top}, t^{bottom}, respectively :

$$\begin{array}{|c|} \hline l^{top} \\ \hline l^{bottom} \\ \hline \end{array} \; \begin{array}{|c|} \hline t^{top} \\ \hline t^{bottom} \\ \hline \end{array}, \text{ where}$$

$$l^{top} = \begin{pmatrix} l_1 \\ \vdots \\ l_{n_1} \end{pmatrix}, l^{bottom} = \begin{pmatrix} l_{(n_1+1)} \\ \vdots \\ l_n \end{pmatrix}, t^{top} = \begin{pmatrix} t_1 \\ \vdots \\ t_{n_1} \end{pmatrix}, t^{bottom} = \begin{pmatrix} t_{(n_1+1)} \\ \vdots \\ t_n \end{pmatrix}.$$

Now we can express equation (18) in the following way:

$$\begin{array}{|c|} \hline l^{top} \\ \hline l^{bottom} \\ \hline \end{array} = \begin{array}{|c|c|} \hline M_1 & M_3 \\ \hline M_4 & M_2 \\ \hline \end{array} \circ \begin{array}{|c|} \hline l^{top} \\ \hline l^{bottom} \\ \hline \end{array} \cup \begin{array}{|c|} \hline t^{top} \\ \hline t^{bottom} \\ \hline \end{array}$$

or as the system of equations:

$$\begin{aligned} l^{top} &= M_1 \circ l^{top} \cup M_3 \circ l^{bottom} \cup t^{top} \\ l^{bottom} &= M_4 \circ l^{top} \cup M_2 \circ l^{bottom} \cup t^{bottom}. \end{aligned} \tag{20}$$

Then applying Lemma 3 and the elimination method we have:

$$\begin{aligned} l^{top} =& (M_1 \cup M_3 \circ M_2{}^{\circledast} \circ M_4)^{\circledast} \circ M_3 \circ M_2{}^{\circledast} \circ t^{bottom} \cup \\ & \cup (M_1 \cup M_3 \circ M_2{}^{\circledast} \circ M_4)^{\circledast} \circ t^{top} \\ l^{bottom} =& M_2{}^{\circledast} \circ M_4 \circ (M_1 \cup M_3 \circ M_2{}^{\circledast} \circ M_4)^{\circledast} \circ M_3 \circ M_2{}^{\circledast} \circ t^{bottom} \cup \\ & \cup M_2{}^{\circledast} \circ M_4 \circ (M_1 \cup M_3 \circ M_2{}^{\circledast} \circ M_4)^{\circledast} \circ t^{top} \cup M_2{}^{\circledast} \circ t^{bottom} \end{aligned} \tag{21}$$

or in the block form:

$$\begin{array}{|c|} \hline l^{top} \\ \hline l^{bottom} \\ \hline \end{array} = \begin{array}{|c|} \hline M^{\circledast} \\ \hline \end{array} \circ \begin{array}{|c|} \hline t^{top} \\ \hline t^{bottom} \\ \hline \end{array},$$ where $M^{\circledast}$ is the matrix defined in the recursive method.

Theorem 7 *Any regular $\circ$-expression R represents a language defined by a vertex-labeled graph.*

Proof.

We show by induction on the number n of operators appearing in R how to construct a graph G such that $L(G) = L(R)$.

If $n = 0$, there are the following three cases to consider.

Case 1. If $R = x$ for some $x \in X$, then G contains only a single vertex with the label x, which is the initial and the final at the same time. Clearly, the language of this graph is $L(G) = x = L(R)$.

Case 2. If $R = xy$ for some $x, y \in X$, then G contains two vertices, one of them is initial and its label is x, the other one is final and its label is y, so we have that $L(G) = s^T \circ M^{\circledast} \circ t = xy = R(L)$, where $M = \begin{pmatrix} \varnothing & xy \\ \varnothing & \varnothing \end{pmatrix}$ is the transition matrix, $s = \begin{pmatrix} x \\ \varnothing \end{pmatrix}$ is a vector of initial vertices and $t = \begin{pmatrix} \varnothing \\ y \end{pmatrix}$ is a vector of final vertices.

Case 3. If $R = \emptyset$ then G does not have any vertices and $L(G) = \varnothing = L(R)$.

Assuming that $n > 0$, we have the following three cases to consider.

Case 1. If $R = R' \cup R''$ where R' and R'' are regular $\circ$-expressions with at most $n-1$ operations then the disjoint sum of two graphs G_1 and G_2 such that $L(G_1) = L(R')$, $L(G_2) = L(R'')$ defines the union of the corresponding languages.

Let the transition matrices for the graphs G_1 and G_2 are denoted by M_1 and M_2, the vectors of initial vertices by s' and s'' and the vectors of final vertices by t' and t'' respectively. The graph G for which $L(G) = L(R)$ can be defined by a matrix M as follows.

If the numbers of vertices in the graphs G_1 and G_2 are n_1 and n_2, respectively, then M is a $n \times n$ matrix where $n = n_1 + n_2$ and

$$M = \begin{array}{|c|c|}\hline M_1 & \varnothing \\ \hline \varnothing & M_2 \\ \hline \end{array} \qquad s = \begin{array}{|c|}\hline s' \\ \hline s'' \\ \hline \end{array} \qquad t = \begin{array}{|c|}\hline t' \\ \hline t'' \\ \hline \end{array}$$

Following the recursive method we compute $M^{\circledast}$:

$$M^{\circledast} = \begin{array}{|c|c|}\hline (M_1 \cup \varnothing \circ {M_2}^{\circledast} \circ \varnothing)^{\circledast} & {M_1}^{\circledast} \circ \varnothing \circ {M_2}^{\circledast} \\ \hline {M_2}^{\circledast} \circ \varnothing \circ {M_1}^{\circledast} & {M_2}^{\circledast} \cup M_2^{\circledast} \circ \varnothing \circ {M_1}^{\circledast} \circ \varnothing \circ {M_2}^{\circledast} \\ \hline \end{array} =$$

$$= \begin{array}{|c|c|}\hline {M_1}^{\circledast} & \varnothing \\ \hline \varnothing & {M_2}^{\circledast} \\ \hline \end{array}$$

Following (19) we have that $L(G) = s^T \circ M^{\circledast} \circ t = s'^T \circ {M_1}^{\circledast} \circ t' \cup s''^T \circ {M_2}^{\circledast} \circ t''$.

Finally, from the equalities $L(G_1) = s'^T \circ M_1^{\circledast} \circ t'$, $L(G_2) = s''^T \circ M_2^{\circledast} \circ t''$, $L(G_1) = L(R')$, $L(G_2) = L(R'')$ we can derive that

$$L(G) = L(R') \cup L(R'') = L(R).$$

Case 2. If $R = R' \circ R''$ then the graph G such that $L(G) = L(R)$ has the following matrix representation:

$$M = \begin{array}{|c|c|}\hline M_1 & t' \circ s''^T \circ M_2 \\ \hline \varnothing & M_2 \\ \hline \end{array} \qquad s = \begin{array}{|c|}\hline s' \\ \hline \varnothing \\ \hline \end{array} \qquad t = \begin{array}{|c|}\hline t' \circ s''^T \circ t'' \\ \hline t'' \\ \hline \end{array}$$

By the recursive method we have that

$M^{\circledast} =$

${M_1}^{\circledast}$	${M_1}^{\circledast} \circ t' \circ s''^T \circ M_2 \circ {M_2}^{\circledast}$
$\varnothing$	${M_2}^{\circledast}$

Now by inserting $M^{\circledast}$ into the expression for $L(G)$ following (19) we get
$L(G) = s^T \circ M^{\circledast} \circ t = s'^T \circ {M_1}^{\circledast} \circ t' \circ s''^T \circ t'' \cup s'^T \circ {M_1}^{\circledast} \circ t' \circ s''^T \circ M_2 \circ {M_2}^{\circledast} \circ t'' =$
$= s'^T \circ {M_1}^{\circledast} \circ t' \circ (s''^T \circ t'' \cup s''^T \circ M_2 \circ {M_2}^{\circledast} \circ t'')$.

By the definitions of matrix operations $M^{\circledast} = M^0 \cup M^+$, and then
$M^{\circledast} \circ M = (M^0 \cup M^+) \circ M = M^0 \circ M \cup M^+ \circ M = M^0 \circ M \cup M \circ M \cup M \circ M \circ M \ldots$.
Note that the matrix M is constructed in such a way that every element M_{ij} is of the form xy, where $x, y \in X$ and the matrix M^0 is diagonal, $[M^0]_{ii} \in X$ and the values of other elements are equal to $\varnothing$. So $M^0 \circ M = M$ for any M and therefore $M^{\circledast} \circ M = M \cup M \circ M \cup M \circ M \circ M \ldots = M^+$.

Finally, by inserting the computed value into the expression for $L(G)$ we have that $L(G) = s'^T \circ {M_1}^{\circledast} \circ t' \circ (s''^T \circ t'' \cup s''^T \circ {M_2}^+ \circ t'')$, and now by (19) we get

$$L(G) = s'^T \circ {M_1}^{\circledast} \circ t' \circ s''^T \circ {M_2}^{\circledast} \circ t'' = L(R') \circ L(R'') = L(R).$$

Case 3. If $R = R'^{\circledast}$ then according to (19):
$L(G_1) = s'^T \circ {M_1}^{\circledast} \circ t' = s'^T \circ {M_1}^+ \circ t' \cup s'^T \circ t' = s'^T \circ {M_1}^* \circ t'$. Let us consider a graph G with a transition matrix $M = M_1 \cup t' \circ s'^T \circ M_1$, a vector of initial vertices $s = s'$ and a vector of final vertices $t = t'$. The language of this graph is $L(G) = s^T \circ M^{\circledast} \circ t = s^T \circ (M \cup M^0)^{\circledast} \circ t$.

Since $M^0 = {M_1}^0$, we can redefine $L(G)$ as:
$L(G) = s'^T \circ ({M_1}^0 \cup M_1 \cup t' \circ s'^T \circ (M_1 \cup {M_1}^0))^{\circledast} \circ t' =$
$= s'^T \circ ((M_1 \cup {M_1}^0) \cup t' \circ s'^T \circ (M_1 \cup {M_1}^0))^* \circ t'$

Using (7) we have
$L(G) = s'^T \circ ((M_1 \cup {M_1}^0) \cup t' \circ s'^T \circ (M_1 \cup {M_1}^0))^* \circ t' =$
$= s'^T \circ (M_1 \cup {M_1}^0)^* \circ (t' \circ s'^T \circ (M_1 \cup {M_1}^0) \circ (M_1 \cup {M_1}^0)^*)^* \circ t_1$.

Applying (8) we get
$L(G) = s'^T \circ (M_1 \cup {M_1}^0)^* \circ t' \circ (s'^T \circ (M_1 \cup {M_1}^0) \circ (M_1 \cup {M_1}^0)^* \circ t')^*$.

Following the definitions of the matrices ${M_1}^*$ and ${M_1}^0$ we have that
$(M_1 \cup {M_1}^0) \circ (M_1 \cup {M_1}^0)^* = {M_1}^{\circledast}$, $(M_1 \cup {M_1}^0)^* = {M_1}^*$ and therefore

$$L(G) = s'^T \circ {M_1}^* \circ t' \circ (s'^T \circ {M_1}^{\circledast} \circ t')^* = L(G_1) \circ (L(G_1))^* = (L(G_1))^+.$$

According to (2) $R^{\circledast} = R_{beg} \circ R_{end} \cup R^+$. Therefore we can obtain a graph with the language $(L(G_1))^{\circledast}$ as follows. For all vertices $v_i \in I$ and $v_j \in F$ such that $\mu(v_i) = \mu(v_j)$ we need to add a new vertex with label $\mu(v_i)$ into G and to make it to be initial and final one at the same time.

Now we conclude that in each case for any regular $\circ$-expression R it is possible to construct a graph G such that $L(G) = L(R)$. This ends the proof of the Theorem 7. □

Theorem 8 *A language $L \subseteq X^+$ is representable by a regular $\circ$-expression if and only if it can be defined by a vertex-labeled graph.*

Proof. The necessary condition is proved in Theorem 6 and the sufficient condition is proved in Theorem 7. □

5 Conclusion

In this work we introduced the algebra $\langle 2^{X^+}, \cup, \circ, \circledast, \varnothing, X \rangle$ which is similar to Kleene algebra. It has been shown that the proposed algebra can be used for the analysis of vertex-labeled graphs in the same way as Kleene algebra is used for the analysis of edge-labeled graphs or finite state automata. Also we show in a constructive way how the new algebra provides a natural translation from vertex-labeled graphs to regular expressions and vice versa.

References

[1] C. Baier, J.-P. Katoen. Principles of Model Checkng. MIT Press, Cambridge, 2008.

[2] E. M. Clarke, O. Grumberg and D. Peled. Model Checking. MIT Press, 1999.

[3] J. H. Conway. Regular Algebra and Finite Machines. Chapman and Hall, London, 1971.

[4] M. Droste, W. Kuich, H. Vogler. Handbook of Weighted Automata. Springer-Verlag, Berlin, Heidelberg, 2009.

[5] G. Dudek, M. Jenkin, E. Milios and D. Wilkes. Map validation and robot self-location in a graph-like world // Robotics and autonomous systems, vol. 22, November 1997, pp.159-178.

[6] S. Eilenberg. Automata, Languages, and Machines. vol A. Academic Press, New York and London, 1974.

[7] I.Grunsky, O.Kurganskyy, I.Potapov. Languages Representable by Vertex-labeled Graphs. Proceedings of the 30th International Symposium on Mathematical Foundations of Computer Science, LNCS, v.3618, 2005, 435-446.

[8] B.A. Davey, H.A. Priestley. Introduction to Lattices and Order. Cambridge University Press, Cambridge,1990.

[9] D. Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. Information and Computation, 110(2):366-390, 1994.

[10] S.V. Sapunov. Equivalence of Vertex-labeled Graphs // Proceedings of IAMM NAS of Ukraine, 2002, v.7, pp.162-167.