On Algebra of Languages Representable by Vertex-Labeled Graphs

Igor Grunsky\textsuperscript{a,c}, Igor Potapov\textsuperscript{b} Elena Pryanichnikova\textsuperscript{c}

\textsuperscript{a} Institute of Applied Mathematics and Mechanics, Ukrainian National Academy of Sciences, 74 R. Luxemburg St, Donetsk, Ukraine

\textsuperscript{b} Department of Computer Science, University of Liverpool, Ashton Building, Ashton Street, Liverpool L69 3BX, U.K.

\textsuperscript{c} State University of Informatics and 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 generated by vertex-labeled graphs and provides a natural translation from vertex-labeled graphs to regular expressions and vice versa.

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

1 Introduction

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

* 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).
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 automata. 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 and they are used to represent an algorithm or a computational process representing steps as boxes of various kinds, and their order by connecting them with arrows [1]. In robotics vertex-labeled graphs are used to describe a topological environment for robot 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 a robot walking on a 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) [2]”. Also such graphs have been widely used in model checking to represent the behaviour of a system and are known as Kripke structures. In this case vertices of a graph represent the reachable states of the system and whose edges represent state transitions and a labeling function maps each vertex to a set of properties that hold in the corresponding state [3,4].

The wide applicability of vertex-labeled graphs indicates the need for detailed analysis of such structures and in particular the study of languages generated by these graphs. The characterization of languages representable by vertex-labeled graphs was already investigated in [1,5] and the problem of vertex minimization in a graph preserving its language was studied in [6]. In this work the class of languages representable by vertex-labeled graphs is studied by introducing an algebra similar to well-known Kleene algebra. The proposed algebra is equipped with three operations: the union of languages, the merging of languages and the iteration (that is also based on merging). In contrast to Kleene algebra it can adequately represent many properties of languages generated by vertex-labeled graphs and provides a natural translation from vertex-labeled graphs to regular expressions and vice versa.

The paper is organized as follows. In Section 1 we provide main definitions, introduce a new algebra and investigate its properties. In Section 2 we first prove that it is possible to solve a system of linear equations in this algebra similarly to the case of Kleene algebra. Then we show how for any vertex-labeled graph to construct a regular expression representing the same language. In Section 3 we introduce the matrix representation of vertex-labeled graphs to prove that for any regular expression in proposed algebra we can construct a graph representing the same language.
2 Basic definitions and properties of the algebra \( \langle 2^{X^+}, \circ, \cup, \otimes, \emptyset, X \rangle \)

Let \( X \) be a finite alphabet, \( X^+ \) be the set of all non-empty finite words in \( X \) and \( 2^{X^+} \) be the set of all languages in \( X \) excluding the empty word. Let us denote the empty set by \( \emptyset \) 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 \( L \cdot R = \{ uv | u \in L, v \in R \} \).

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** The operations \( L \cup R, L \circ R, L^+, L^*, L^\otimes \) on \( L, R \subseteq X^+ \) are defined as follows:

1. \( L \cup R = \{ w | w \in L \text{ or } w \in R \} \);
2. \( L \circ R = \{ w_1 \circ w_2 | w_1 \in L, w_2 \in R \} \);
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. \( L^\otimes = L_{\text{beg}} \circ L^* \circ L_{\text{end}} \), where \( L_{\text{beg}} = \{ x | x w \in L, x \in X, w \in X^* \} \); \( L_{\text{end}} = \{ x | x w \in L, x \in X, w \in X^* \} \).

**Lemma 2** The operation \( \circ \) in the algebra \( \langle 2^{X^+}, \circ, \cup, \otimes, \emptyset, X \rangle \) is associative:

\[
P \circ (R \circ Q) = (P \circ R) \circ Q.
\]

**PROOF.** Let us show first that \( P \circ (R \circ Q) \subseteq (P \circ R) \circ Q \). Assume that a word \( w \) belongs to \( P \circ (R \circ Q) \). Then following the definition of \( \circ \)-operation we have that the word \( w \) is in the form \( w_1 x w_2 y w_3 \), where \( w_1 \in P, w_2 \in R, w_3 \in Q, x, y \in X \). Thus, \( w \) also belongs to the language \( (P \circ R) \circ Q \). The proof of another inclusion can be constructed in a similar way. Since \( P \circ (R \circ Q) \subseteq (P \circ R) \circ Q \) \( (P \circ R) \circ Q \subseteq P \circ (R \circ Q) \) for any word \( w \in X^+ \), then \( P \circ (R \circ Q) = (P \circ R) \circ Q \).

The algebra \( \langle 2^{X^+}, \circ, X \rangle \) is a monoid, since by the definition of \( \circ \)-operation the following equalities \( R \circ X = X \circ R = R \) hold for any \( R \in 2^{X^+} \). The operation \( \cup \) coincides with the set-theoretic operation of union and therefore it is commutative and associative. Thus, \( \langle 2^{X^+}, \cup, \emptyset \rangle \) is idempotent commutative monoid due to the fact that \( R \cup \emptyset = \emptyset \cup R = R, R \cup R = R \) for any \( R \in 2^{X^+} \).
Lemma 3 Both operations $\circ$ and $\cup$ satisfy to distributivity law:

\[
R \circ (Q \cup P) = R \circ Q \cup R \circ P \\
(R \cup Q) \circ P = R \circ P \cup Q \circ P
\]

PROOF. Let us show that $R \circ (Q \cup P) \subseteq R \circ Q \cup R \circ P$. Let $w \in R \circ (Q \cup P)$ then $w$ can be represented as $w = w_1 \circ w_2$, where $w_1 \in R$, $w_2 \in Q \cup P$ and if $w_2 \in Q \cup P$, then $w_2 \in Q$ or $w_2 \in P$ holds. In the first case where $w_2 \in Q$ we have $w = w_1 \circ w_2$, $w_1 \in R$, $w_2 \in Q$ and therefore $w \in R \circ Q$. In the second case where $w_2 \in P$, we have $w = w_1 \circ w_2$, $w_1 \in R$, $w_2 \in P$ and therefore $w \in R \circ P$. Now from $w \in R \circ Q \cup R \circ P$ follows that $R \circ (Q \cup P) \subseteq R \circ Q \cup R \circ P$. The proof of another inclusion can be done by analogy and from two inclusion follow that $R \circ (Q \cup P) = R \circ Q \cup R \circ P$. The fact that $(R \cup Q) \circ P = R \circ P \cup Q \circ P$ can be derived in a similar way.

From the above definitions follows that the algebra $\langle 2^{X^+}, \circ, \cup, \varnothing, X \rangle$ is idempotent semiring. Set $2^{X^+}$ is an ordered set in relation to inclusion and $R \leq Q$ holds if and only if $Q \subseteq R = Q$. Assuming that $R \leq Q$, we have that $R \circ P \leq Q \circ P$, $R \cup P \leq Q \cup P$, $R^* \leq Q^*$, i.e. the relation $\leq$ is the partial order relation and it is monotonic in respect to operations of the algebra $\langle 2^{X^+}, \circ, \cup, \varnothing, \varnothing, X \rangle$.

The main feature of proposed algebra is that, in a contrast to Kleene algebra, the operation $\circ$ is partially defined. In Kleene algebra $R \cdot Q = \varnothing$ holds if and only if $R = \varnothing$ or $Q = \varnothing$ and in the algebra $\langle 2^{X^+}, \circ, \cup, \varnothing, \varnothing, X \rangle$ it is not always true, for example let $R = \{ab\}$ and $Q = \{cd\}$ then we have that $R \circ Q = \varnothing$.

Another diversity of proposed algebra is that in Kleene algebra a set $R^*$ always contains the empty word and is infinite for any $R$ [7], but in algebra $\langle 2^{X^+}, \circ, \cup, \varnothing, \varnothing, X \rangle$ the result of applying $\circ$ can be infinite, finite or the empty set:

- if $R = \{aba\}$ then $R^* = \{a, aba, ababa, ...\}$,
- if $R = \{ab\}$, then $R^* = \{ab\}$,
- if $R = \{\varnothing\}$, then $R^* = \{\varnothing\}$.

Definition 4 The regular expressions in the algebra $\langle 2^{X^+}, \circ, \cup, \varnothing, \varnothing, X \rangle$ are defined recursively in the following way:

(1) The empty set $\varnothing$ is a regular expression and represents the language $L(\varnothing) = \varnothing$.
(2) $x$ is a regular expression and represents the language $L(x) = \{x\}$ for all $x \in X$. 

4
(3) $xy$ is a regular expression representing the language $L(xy) = \{xy\}$ for all $x \in X$ and $y \in X$.

(4) If $R$ and $Q$ are regular expressions representing languages $L(R)$ and $L(Q)$ correspondingly, then expressions $(R \circ Q)$, $(R \cup Q)$, $(R \circ \emptyset)$ are also regular and $L(R \circ Q) = L(R) \circ L(Q)$, $L(R \cup Q) = L(R) \cup L(Q)$, $L(R \circ \emptyset) = (L(R))^\circ$.

**Definition 5** Two regular expressions are equivalent if and only if their corresponding languages coincide.

Here we introduce several notations about vertex-labeled graph that will be used in the rest of the paper.

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

Let $I \subseteq Q$ be a set of initial graph vertices and $F \subseteq Q$ be a set of final vertices. Two vertices $q_1$ and $q_2$ are adjacent if there is the edge $(q_1, q_2) \in E$. Then a path in a graph $G$ is defined as a finite sequence of adjacent vertices $l = q_1q_2...q_k$, where $(q_i, q_{i+1}) \in E$, $k$ denotes a length of the path, $q_1$ is the initial vertex of the path and $q_k$ is the final vertex. The label of this path will be denoted by $x = \mu(q_1)\mu(q_2)...\mu(q_k) = x_1x_2..x_k \in X^+$. We also define a path of a zero length for each vertex $q$, i.e. a path which starts and immediately finishes in it. The label of such zero length path from a vertex $q$ will be $x \in X$, where $\mu(q) = x$.

A set of labels for all paths from an initial vertex $q_i$ into a final vertex $q_k \in F$ we denote as a language $L_i$ generated by a vertex $q_i$ in a graph $G$. Then a set of labels for all paths with the initial vertices from a set $I$ and the final vertices from a set $F$ is denoted as a language $L(G)$ generated by a graph $G$.

In automata theory one of the fundamental results is the Kleene’s theorem on the coincidence of regular and rational languages, representable by regular expressions in Kleene algebra [7]. The main aim of this work is to prove a similar theorem for languages generated by vertex labeled graphs.

### 3 Representation of Graph Languages in $\langle 2^{X^+}, \circ, \cup, \circ, \emptyset, X \rangle$.

**Theorem 6** Any language defined by a vertex-labeled graph can be represented by a regular expression in the algebra $\langle 2^{X^+}, \circ, \cup, \circ, \emptyset, X \rangle$.

Let us show that for any graph $G$, generating a language $L(G)$, exists a regular expression $R$ in the algebra $\langle 2^{X^+}, \circ, \cup, \circ, \emptyset, X \rangle$, such that $L(R) = L(G)$.
Note that a language $L_i$ contains all words of $L_j$ which are extended by the left concatenation of symbol $\mu(q_i)$ for each edge $(q_i, q_j) \in E$. If the vertex $q_i$ is a final vertex of $G$ then a symbol $\mu(q_i)$ belongs to a language $L_i$ as a single word. So a language $L_i$ can be represented by the following equation:

$$L_i = \mu(q_i) \cup L_1 \cup \mu(q_i) \cup \mu(q_i) \cup L_n \cup \alpha \cup \mu(q_i), \quad (3)$$

where $n$ is a number of vertices in $G$ and if a vertex $q_i$ is final then $\alpha = X$, otherwise $\alpha = \emptyset$.

Using the above representation a language generated by a graph $G$ can be defined as $L(G) = \bigcup_{q_i \in I} L_i$. Thus $L(G)$ can be derived by solving the following system of equations:

$$L_1 = r_{11} \cup L_1 \cup r_{12} \cup L_2 \cup \ldots \cup L_n \cup p_1$$

$$L_2 = r_{21} \cup L_1 \cup r_{22} \cup L_2 \cup \ldots \cup L_n \cup p_2$$

$$\ldots$$

$$L_n = r_{n1} \cup L_1 \cup r_{n2} \cup L_2 \cup \ldots \cup L_n \cup p_n,$$  

(4)

where $p_i \in \{X \cup \emptyset\}$, and all $r_{ij}$ are in the form $xy$, $x, y \in X$.

Since every $i$-th equation of the system (4) has a form $L_i = r \cup L_i \cup q$, where $r = r_{ii}$, $q = \bigcup_{j \neq i} r_{ij} \cup L_j \cup p_i$ we can solve the system of equations by applying the following result:

**Lemma 7** The equation $Y = r \cup Y \cup q$ with one unknown in the algebra $(2^X, \cup, \cup, \smallfrown, \emptyset, X)$ has the unique minimal solution $Y = r^* \cup q$ if $X \cap r = \emptyset$.

**PROOF.** The proof will be based on the fixed point theorem which states: an equation $x = f(x)$, where $f$ is continuous function from any closed semiring to itself, has a minimal solution [1].

The idempotent semiring $(2^X, \cup, \smallfrown, \emptyset, X)$ is closed, since any ordered sequence of languages $L_1, L_2, \ldots, L_n, \ldots$ has a supremum, which is the union of all languages in this sequence, and the operation $\cup$ preserves supremums of sequences.

Due to the facts that operations $\smallfrown$ and $\cup$ are continuous and supremum of continuous functions is continuous, the right part of the equation $Y = r \cup Y \cup q$ is a continuous mapping $f : 2^X \to 2^X$.

By the fixed point theorem, the least fixed point of $Y = r \cup Y \cup q$ will be $Y = \bigcup_{n \geq 0} f^n(\emptyset)$, where $\emptyset$ is the least element of a set $2^X$, $f^0(x) = x$ and $f^n(x) = f(f^{n-1}(x))$ for any $n > 0$. 

6
Taking into account that 

\[ f^0(\emptyset) = \emptyset, \]
\[ f^1(\emptyset) = q, \]
\[ f^2(\emptyset) = r \circ q \cup q, \]
\[ f^3(\emptyset) = r \circ r \circ q \cup r \circ q \cup q \text{ (by distributed law according to (2))} \]

... 

\[ f^n(\emptyset) = (r^{n-1} \cup r^{n-2} \cup \ldots \cup r \circ r \cup r) \circ q \cup q, \]
we have the following

\[ Y = \bigcup_{n \geq 0} f^n(\emptyset) = \left( \bigcup_{n \geq 0} (r^{n-1} \cup r^{n-2} \cup \ldots \cup r \circ r \cup r) \right) \circ q \cup q = r^+ \circ q \cup q. \quad (5) \]

By the definition of \( \circ \) we have that for a regular expressions \( p \) such that \( L(p) \subseteq X \) and any regular expressions \( q \) the following inclusion holds:

\[ L(p \circ q) \subseteq L(q). \quad (6) \]

Since \( L(r_{beg}) \subseteq X \) and \( L(r_{end}) \subseteq X \) for any regular expression \( r \), we have the inclusion \( L(r_{beg} \circ r_{end}) \subseteq X \).

Now by definition of \( \cup \), a word \( w \) belongs to a language representable by a regular expression \( r_{beg} \circ r_{end} \circ q \cup q \) if \( w \in L(r_{beg} \circ r_{end} \circ q) \) or \( w \in L(q) \). According to (6) \( L(r_{beg} \circ r_{end} \circ q) \subseteq L(q) \) and therefore \( w \in L(q) \). From that follows that for any \( q \) and \( r \) we have

\[ r_{beg} \circ r_{end} \circ q \cup q = q. \quad (7) \]

In this case the expression \( Y = r^+ \circ q \cup q \) from (5) can be substituted by

\[ Y = r^+ \circ q \cup r_{beg} \circ r_{end} \circ q \cup q = (r^+ \cup r_{beg} \circ r_{end}) \circ q \cup q. \quad (8) \]

Following the definition of \( \ast \) we can derive that

\[ r^\ast = r_{beg} \circ (r^\ast) \circ r_{end} = r_{beg} \circ (X \cup r \cup r \circ r \cup r \circ r \cup r \ldots) \circ r_{end} = \]
\[ = r_{beg} \circ r_{end} \cup r_{beg} \circ r \circ r_{end} \cup r_{beg} \circ r \circ r \circ r_{end} \cup r_{beg} \circ r \circ r \circ r \circ r_{end} \ldots = r_{beg} \circ r_{end} \cup r^+. \]

Applying (8) we can now define a solution as

\[ Y = r^\ast \circ q \cup q. \quad (9) \]

Let us show that if \( X \cap r = \emptyset \) then the solution is unique. Let \( Y' \) be a solution of the equation \( Y = r \circ Y \cup q \) and a word \( p \in Y' \) is of the length \( m \). By a
sequence of iterative substitutions of the right part of equation \( Y = r \circ Y \cup q \) instead of \( Y \) we express \( Y \) as follows:

\[
Y = r \circ Y \cup q = r \circ (r \circ Y \cup q) \cup q = r \circ r \circ Y \cup r \circ q \cup q = \ldots \]

Assuming that \( X \cap r = \emptyset \), all words of the language \( r^m \) are at least of length \( m + 1 \) and therefore \( p \in r^{m-1} \circ q \cup \ldots \cup r \circ q \cup q \) and \( p \in r^\circ \circ q \cup q \). Thus, \( Y = r^\circ \circ q \cup q \) is the only solution of the equation \( Y = r \circ Y \cup q \).

Considering how the system (4) is constructed, it can be noticed that in all equations and for all coefficients the following holds: \( r_{ij}\text{beg} = r_{ij}\text{end} = p_{\text{beg}} \) and \( r_{ij} \cap X = \emptyset \). According to the definition of \( \circ \) we have that \( r^\circ \circ q = r_{\text{beg}} \circ r^* \circ r_{\text{end}} \circ q \), and therefore if \( r_{\text{beg}} = r_{\text{end}} = q_{\text{beg}} \), then \( r^\circ \circ q = r_{\text{beg}} \circ r_{\text{end}} \circ q \cup r \circ q \cup r \circ q \cup q \cup \ldots = r^\circ \circ q \cup q \).

Thus, if the equation \( L_i = r \circ L_i \cup q \) represents a language generated by a vertex \( q_i \) in a graph \( G \), its unique solution can be defined as

\[
L_i = r^\circ \circ q.
\]

In order to solve a system (4) we use a kind of Gaussian elimination by analogy to solving a system of equations with regular coefficients in algebra Kleene \( \langle 2^X, \cdot, \cup, \circ, \emptyset, \lambda \rangle [1] \).

Applying (11) we can eliminate \( L_n \) from the right part of the final equation in the system (4). 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 \( L_1, \ldots, L_{n-2} \). By continuing such substitutions for all equations in the system we can get values for all \( L_i \), which are regular expressions in the algebra \( \langle 2^X, \circ, \cup, \emptyset, \emptyset, X \rangle \) based on Lemma 7.

From the above follows that by solving the equation (4) for any graph \( G \) we can find a regular 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 algebra \( \langle 2^X, \circ, \cup, \emptyset, X \rangle \). First we define the following two operations \( \cup \) and \( \circ \) on matrices:
(1) Let $A$ and $B$ be $m \times n$ matrices, then $A \cup B = C$, where $C$ is a $m \times n$ matrix and $C_{ij} = A_{ij} \cup B_{ij}$ for all $i, j \in \{1, \ldots, n\}$.

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

As it can be seen, both 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 $(\mathcal{P}(X), \cup, \circ, \emptyset, X)$.

The transpose of an $m \times n$ matrix $A$ is defined as an $n \times m$ matrix $A^T$, which is formed by turning all the row of a given matrix into columns and vice-versa, i.e. $A^T_{ij} = A_{ji}$. The zero matrix $Z_n$ is defined as a square $n \times n$ matrix where all elements are equal to $\emptyset$. In the identity matrix $E_n$ all elements on the main diagonal are defined as $E_{nj} = \bigcup_{x \in X} x$ and all other elements are equal to $\emptyset$.

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

Let us for a graph $G$ with $n$ vertices put in correspondence a transition matrix of size $n \times n$ which elements are defined as follows:

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

A vector $u$ of size $n$ will represent a set of initial vertices of a graph

$$u_i = \begin{cases} 
\mu(q_i), & \text{if } q_i \in I; \\
\emptyset, & \text{otherwise.}
\end{cases}$$

and a vector $v$ of size $n$ stands for a set of final vertices

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

Also let $l = (l_1, l_2, \ldots, l_n)$ be a vector of size $n$, where each $l_i$ be a language generated by a vertex $q_i$ in a graph $G$.

Let us define the operation $+$ for transition matrix $M$ as $M^+ = M \cup M \circ M \cup M \circ M \circ M \ldots$. Let $M$ be a transition matrix of a graph $G$ and its element $M_{ij}$ contains a label for a path of the length 1 from a vertex $q_i$ into a vertex $q_j$, then $[M \circ M]_{ij} = \bigcup_{k=1}^{m} M_{ik} \circ M_{kj}$ corresponds to a sum of all path of the length 2 from $q_i$ to $q_j$; $[M \circ M \circ M]_{ij}$ is a sum of all such paths of the length
3, and so on. Thus, $M^+$ is a matrix, where every element $[M^+]_{ij}$ is a regular expression of algebra $\langle 2^X, \circ, \cup, \emptyset, \emptyset, X \rangle$ and which represents all possible paths of a nonzero length in a graph $G$ from a vertex $q_i$ into a vertex $q_j$.

Let $M^* = E_n \cup M^+$, $M^\emptyset = M' \cup M^+$, where $M'$ be a matrix representing all zero length paths in a graph $G$:

$$M' = \begin{cases} \mu(q_i), & \text{if } i = j; \\ \emptyset, & \text{if } i \neq j, \end{cases}$$

and $M^\emptyset_{ij}$ defines all paths in a graph $G$ from a vertex $q_i$ into a vertex $q_j$, which length is greater or equal to zero. From the above definitions follows that $M^\emptyset = (M' \cup M)^+$. Using the matrix representation for graphs with labeled vertices and basic properties of matrix operations we can define a system (4) as a matrix equation

$$l = M \circ l \cup v,$$  \hspace{1cm} (12)

and to represent a language generated by a graph $G$ as $L(G) = u^T \circ l$.

Since the algebra of matrices with elements over the algebra $\langle 2^X, \circ, \cup, \emptyset, \emptyset, X \rangle$ is closed idempotent semiring we can apply fixed point theorem to show that the equation $Y = R \circ Y \cup Q$, where $R, Q$ are given matrices, has the least fixed point solution $Y = R^\emptyset \circ Q \cup Q$.

Since all elements of $j$'s-column in a matrix $M$ are paths into a vertex $q_j$ from other vertices we have that for any $i$: $M_{ij\text{end}} = \mu(q_j)$, where $v_j = \emptyset$ or $v_j = \mu(q_j)$ and for all diagonal elements of $M$ $M_{i\text{beg}} = M_{i\text{end}} = v_i$, where $v_i \neq \emptyset$. From it follows that $M' \circ v = v$. Then the solution of equation (12) can be represented in the form:

$$l = M^+ \circ v \cup v = M^+ \circ v \cup M' \circ v = (M^+ \cup M') \circ v = M^\emptyset \circ v = (M' \cup M)^\emptyset \circ v,$$  \hspace{1cm} (13)

and the language generated by a graph $G$ as

$$L(G) = u^T \circ M^\emptyset \circ v = u^T \circ (M \cup M')^\emptyset \circ v.$$  \hspace{1cm} (14)

In order to compute the values of a matrix $M^\emptyset$ we can use the same method of eliminations which was applied for solving the system (4). However, for further computations it will be more convenient to use a recursive method that is similar to the one applied in [8] for transition matrices of finite state automata.

**Recursive method**
Given a graph $G$ with $n$ vertices. The recursive method of finding a matrix $T^\circ$, where $T=\bar{M} \cup M'$ is described below:

1. If $n = 1$, then $T^\circ = (T_{11})^\circ$;

2. If $n > 1$,
   (a) take any value $n_1$ such that $0 < n_1 < n$,
   (b) split a matrix $T = \begin{pmatrix} T_{11} & \cdots & T_{1n} \\ \vdots & \ddots & \vdots \\ T_{n1} & \cdots & T_{nn} \end{pmatrix}$ into the following blocks
      \[ T_1 = \begin{pmatrix} T_{11} & \cdots & T_{1n_1} \\ \vdots & \ddots & \vdots \\ T_{n_1 n_1} & \cdots & T_{nn_1} \end{pmatrix}, \quad T_2 = \begin{pmatrix} T_{(n_1 + 1)(n_1 + 1)} & \cdots & T_{(n_1 + 1)n} \\ \vdots & \ddots & \vdots \\ T_{nn} & \cdots & T_{nn} \end{pmatrix}, \]
      \[ T_3 = \begin{pmatrix} T_{1(n_1 + 1)} & \cdots & T_{1n} \\ \vdots & \ddots & \vdots \\ T_{n_1(n_1 + 1)} & \cdots & T_{nn} \end{pmatrix}, \quad T_4 = \begin{pmatrix} T_{(n_1 + 1)1} & \cdots & T_{(n_1 + 1)n_1} \\ \vdots & \ddots & \vdots \\ T_{n_1} & \cdots & T_{nn_1} \end{pmatrix}, \]
   (c) compute $T_1^\circ$ and $T_2^\circ$
   (d) define $T^\circ$ as

\[
\begin{array}{c}
(T_1 \cup T_3 \circ T_2^\circ \circ T_4)^\circ \\
T_2^\circ \circ T_4 \circ (T_1 \cup T_3 \circ T_2^\circ \circ T_4)^\circ \\
(T_1 \cup T_3 \circ T_2^\circ \circ T_4)^\circ \circ T_3 \circ T_2^\circ \\
T_2^\circ \circ T_4 \circ (T_1 \cup T_3 \circ T_2^\circ \circ T_4)^\circ \circ T_3 \circ T_2^\circ \cup T_2^\circ
\end{array}
\]

Let us proof the correctness of this method. In the first case where $n = 1$, we have that $T^\circ = (T_{11})^\circ$ by the definition of $\circ$-operation. For the second case $n > 1$ let us define the equation (12) using block matrices and block vectors. First let us split vectors $l$ and $v$ on blocks $l^{\text{top}}, l^{\text{bottom}}$ and $v^{\text{top}}, v^{\text{bottom}}$, respectively:

\[
\begin{array}{c}
l^{\text{top}} \\
l^{\text{bottom}} \\
v^{\text{top}} \\
v^{\text{bottom}}
\end{array}
\text{where}
\]
Now equation (12) can be defined in the following way:

\[
\begin{bmatrix}
T_1 & T_3 \\
T_4 & T_2
\end{bmatrix} = \begin{bmatrix}
l_1 \\
\vdots \\
l_n
\end{bmatrix} \circ \begin{bmatrix}
T_1 & T_3 \\
T_4 & T_2
\end{bmatrix} \cup \begin{bmatrix}
l_1 \\
\vdots \\
l_n
\end{bmatrix}
\]

or as the system of equations:

\[
\begin{align*}
T_{\text{top}} &= T_1 \circ T_{\text{top}} \cup T_3 \circ T_{\text{bottom}} \cup v_{\text{top}} \\
T_{\text{bottom}} &= T_4 \circ T_{\text{top}} \cup T_2 \circ T_{\text{bottom}} \cup v_{\text{bottom}}.
\end{align*}
\]

Then applying Lemma 7 and the method of unknown elimination we have:

\[
\begin{align*}
T_{\text{top}} &= (T_1 \cup T_3 \circ T_2 \circ T_4) \circ T_3 \circ T_2 \circ v_{\text{bottom}} \cup (T_1 \cup T_3 \circ T_2 \circ T_4) \circ v_{\text{top}} \\
T_{\text{bottom}} &= T_2 \circ T_4 \circ (T_1 \cup T_3 \circ T_2 \circ T_4) \circ T_3 \circ T_2 \circ v_{\text{bottom}} \cup T_2 \circ T_4 \circ (T_1 \cup T_3 \circ T_2 \circ T_4) \circ v_{\text{top}} \cup v_{\text{bottom}}.
\end{align*}
\]

Let us consider a square $n \times n$ matrix which is defined as follows. Assume that $T_1, T_2, T_3, T_4$ are block matrices which form $T$ and also that $T_1$ and $T_2$ are already known for two blocks $T_1$ and $T_2$. Then a matrix $X$ will have the form:

\[
\begin{bmatrix}
(T_1 \cup T_3 \circ T_2 \circ T_4) \circ T_3 \circ T_2 \\
T_2 \circ T_4 \circ (T_1 \cup T_3 \circ T_2 \circ T_4)
\end{bmatrix}
\]

Finally computing the elements of $X \circ v$, we get:

\[
\begin{bmatrix}
(T_1 \cup T_3 \circ T_2 \circ T_4) \circ v_{\text{top}} \cup (T_1 \cup T_3 \circ T_2 \circ T_4) \circ T_3 \circ T_2 \circ v_{\text{bottom}} \\
T_2 \circ T_4 \circ (T_1 \cup T_3 \circ T_2 \circ T_4) \circ v_{\text{top}} \cup T_2 \circ T_4 \circ (T_1 \cup T_3 \circ T_2 \circ T_4) \circ v_{\text{bottom}}
\end{bmatrix} = \begin{bmatrix}
T_{\text{top}} \\
T_{\text{bottom}}
\end{bmatrix}
\]

Thus, the matrix defined in (17) coincides with the matrix $T^\circ$.

**Theorem 8** Any regular expression $R$ in algebra $\langle 2^{X^+}, \circ, \cup, \otimes, \emptyset, X \rangle$ defines a language generated by a vertex-labeled graph.
PROOF. The proof of this theorem is done by induction along the structure of expression \( R \).

Let \( R \) contains \( n \) operations. If \( n = 0 \) then by definition of regular expressions of algebra \( \langle 2^{X^+}, \circ, \cup, \otimes, \emptyset, X \rangle \) we have the following cases:

1. \( R = x \), where \( x \in X \);
2. \( R = xy \), where \( x \in X \) and \( y \in X \);
3. \( R = \emptyset \).

Let us consider a graph \( G \) for which the equality \( L(G) = L(R) \) holds. In the first case a graph \( G \) contains only a single vertex with a label \( x \), which is the initial and the final label at the same time. Following (14) the language of this graph is \( L(G) = x = L(R) \).

For the second case we have that \( L(G) = u^T \circ M^{\#} \circ v = xy = R(L) \), where

\[
M = \begin{pmatrix} \emptyset & xy \\ \emptyset & \emptyset \end{pmatrix}
\]

is a transition matrix, \( u = \begin{pmatrix} x \\ \emptyset \end{pmatrix} \) is a vector of initial vertices

and \( v = \begin{pmatrix} \emptyset \\ y \end{pmatrix} \) is a vector of final vertices.

Finally, in the third case a graph \( G \) does not have any vertices and \( L(G) = \emptyset = L(R) \).

Assuming that \( n > 0 \) and following definition of regular expressions we have 3 cases to consider

1. \( R = R' \cup R'' \)
2. \( R = R' \circ R'' \)
3. \( R = R^{\otimes} \)

where \( R' \) and \( R'' \) are regular expressions with \( n - 1 \) operations.

Let us assume that for two regular expressions \( R' \) and \( R'' \) there are two graphs \( G_1 \) and \( G_2 \) such that \( L(G_1) = L(R') \), \( L(G_2) = L(R'') \). The transition matrices for graphs \( G_1 \) and \( G_2 \) are denoted by \( M_1 \) and \( M_2 \), vectors of initial vertices by \( u' \) and \( u'' \) and vectors of final vertices by \( v' \) and \( v'' \) respectively.

Case 1: \( R = R' \cup R'' \)

Let us construct a graph \( G \) for which \( L(G) = L(R) \), where \( R = R' \cup R'' \).

This graph can be defined by a matrix \( M \) as follows. If the number of vertices
in a graph $G_1$ is $n_1$ and in $G_2$ is $n_2$ then $M$ is a square matrix of an order $n = n_1 + n_2$ and both vectors $u$ and $v$ are of a size $n$.

$$M = \begin{array}{cc} M_1 & \emptyset \\ \emptyset & M_2 \end{array} \quad u = \begin{array}{c} u' \\ u'' \end{array} \quad v = \begin{array}{c} v' \\ v'' \end{array}$$

Following (17) we compute $M^*$:

$$M^* = \begin{array}{cc} (M_1 \cup \emptyset \circ M_2^* \circ \emptyset)^* & M_1^* \circ \emptyset \circ M_2^* \\ M_2^* \circ \emptyset \circ M_1^* & M_2^* \cup M_2^* \circ \emptyset \circ M_1^* \circ \emptyset \circ M_2^* \end{array} = \begin{array}{cc} M_1^* & \emptyset \\ \emptyset & M_2^* \end{array}$$

Also following (14) we have that $L(G) = u^T \circ M^* \circ v$. Thus, substituting this expression into $M^*$ we get

$$L(G) = u'^T \circ M_1^* \circ v' \cup u''^T \circ M_2^* \circ v''.$$

Finally, from the following equalities $L(G_1) = u'^T \circ M_1^* \circ v'$, $L(G_2) = u''^T \circ M_2^* \circ v''$, $L(G_1) = L(R')$, $L(G_2) = L(R'')$ we can derive that

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

**Case 2: $R = R' \circ R''$**

Let us consider the second case and show that for $G_1$ and $G_2$ satisfying to the following conditions $L(G_1) = L(R')$, $L(G_2) = L(R'')$ and $R = R' \circ R''$ it is possible to construct a graph $G$ such that $L(G) = L(R)$.

The graph $G$ has the following matrix representation:

$$M = \begin{array}{cc} M_1 & v' \circ u''^T \circ M_2 \\ \emptyset & M_2 \end{array} \quad u = \begin{array}{c} u' \\ \emptyset \end{array} \quad v = \begin{array}{c} v' \circ u''^T \circ v'' \end{array}$$

By (17) we have that

$$M^* = \begin{array}{cc} M_1^* & M_1^* \circ v' \circ u''^T \circ M_2 \circ M_2^* \\ \emptyset & M_2^* \end{array}$$

Now by inserting $M^*$ into the expression for $L(G)$ following (14) we get
\[ L(G) = u^T \circ M^* \circ v = u^T \circ M_1^* \circ v' \circ u'^T \circ v'' \circ u'^T \circ M_1^* \circ v' \circ u'^T \circ M_2 \circ M_2^* \circ v'' = \]
\[ = u^T \circ M_1^* \circ v' \circ (u'^T \circ v'' \cup u'^T \circ M_2 \circ M_2^* \circ v''). \]

Following the definitions of matrix operations we know that \( M^* = M' \cup M^+ \), and then
\[ M^* \circ M = (M' \cup M^+) \circ M = M' \circ M \cup M^+ \circ M = M' \circ M \cup M \cup M \circ M \cup M \circ M ... = M^+. \]

Note that the matrix \( M \) is constructed in such a way that every element \( M_{ij} \) in the form \( xy \), where \( x, y \in X \) and the matrix \( M' \) is diagonal where main diagonal elements \( M'_{ii} = x \) and values of other elements are equal to \( \emptyset \). So \( M' \circ M = M \) for any \( M \) and therefore
\[ M^* \circ M = M \cup M \circ M \cup M \circ M ... = M^+. \]

Finally, by inserting computed value into expression for \( L(G) \) we have that \( L(G) = u^T \circ M_1^* \circ v' \circ (u'^T \circ v'' \cup u'^T \circ M_2 \circ M_2^* \circ v'') \), and now by (14) we get
\[ L(G) = u^T \circ M_1^* \circ v' \circ u'^T \circ M_2^* \circ v'' = L(R') \circ L(R'') = L(R). \]

**Case 3:** \( R = R_1^* \)

Finally, we show that for a graph \( G_1 \) and a regular expression \( R' \) such that \( L(G_1) = L(R') \) it is possible to construct a graph \( G \) with \( L(G) = L(R) \), where \( R = R'^* \).

According to (13) and (14):
\[ L(G_1) = u^T \circ M_1^* \circ v' = u^T \circ M_1^+ \circ v' \cup u'^T \circ v' = u^T \circ M_1^* \circ v'. \]

Let us consider a graph \( G \) with a transition matrix \( M = M_1 \cup v' \cup u'^T \circ M_1 \), a vector of initial vertices \( u = u' \) and a vector of final vertices \( v = v' \). The language of this graph can be defined as: \( L(G) = u^T \circ M^* \circ v = u^T \circ (M \cup M')^* \circ v. \)

Since \( M' = M_1' \), we can redefine \( L(G) \) as:
\[ L(G) = u^T \circ ((M_1' \cup M_1 \cup v' \circ u'^T \circ M_1'))^* \circ v' = \]
\[ = u^T \circ ((M_1 \cup M_1') \cup v' \circ u'^T \circ (M_1 \cup M_1'))^* \circ v'. \]

Following the definition of \(*\)-operation we have that for any \( R \) and \( Q \) the equality \( (R \cup Q)^* = R^* \circ (Q \circ R^*)^* \) holds. A language which is represented
by the expression \((R \cup Q)^*\) consists of all words that can be constructed by merging words from \(Q\) and \(R\) and also from symbols of \(X\). On the other hand, any word from \(R^* \circ (Q \circ R)^*\) is in the form \((Q^i \circ R) \circ (Q^j \circ R) \circ \ldots \circ (Q^m \circ R) \circ Q^m\), i.e. if a word \(w\) belongs to \(R^* \circ (Q \circ R)^*\) then it also belongs to \((R \cup Q)^*\) and vice versa. Thus, \((R \cup Q)^* = R^* \circ (Q \circ R)^*\) holds for any \(R\) and \(Q\).

Using the above property we have that
\[
L(G) = u^T \circ ((M_1 \cup M_1') \cup v \circ u^T \circ (M_1 \cup M_1'))^* \circ v' = u^T \circ (M_1 \cup M_1')^* \circ (v' \circ u^T \circ (M_1 \cup M_1') \circ (M_1 \cup M_1')^* \circ v_1.
\]

Also from the definition of \(*\)-operation follows that \((R \circ Q)^* \circ R = R \circ (Q \circ R)^*\) for any \(R\) and \(Q\):
\[
(R \circ Q)^* \circ R = (X \cup R \circ Q \cup R \circ Q \circ R \circ Q \cup \ldots) \circ R = R \cup R \circ Q \circ R \cup R \circ Q \circ R \circ Q \circ R \cup \ldots = R \circ (X \cup Q \circ R \cup Q \circ R \circ Q \circ R \cup \ldots) = R \circ (Q \circ R)^*.
\]

Applying it we get
\[
L(G) = u^T \circ (M_1 \cup M_1') \circ v' \circ (u^T \circ (M_1 \cup M_1') \circ (M_1 \cup M_1')^* \circ v')^*.
\]

Following definitions of matrices \(M_1^*\) and \(M_1'\) we have that
\[
(M_1 \cup M_1') \circ (M_1 \cup M_1')^* = M_1^\oplus, \quad (M_1 \cup M_1')^* = M_1^*\quad \text{and therefore}
\]
\[
L(G) = u^T \circ M_1^* \circ v' \circ (u^T \circ M_1^\oplus \circ v')^* = L(G_1) \circ (L(G_1))^* = (L(G_1))^+.
\]

According to (8) and (9): \(R^\oplus = R_{beg} \circ R_{end} \cup R^+\). Therefore to get a graph with a language \((L(G_1))^\oplus\) from a graph \(G\) generating the language is \(L(G) = (L(G_1))^+\) we can do it as follows. For all vertices \(q_i \in I\) and \(q_j \in F\) such that \(\mu(q_i) = \mu(q_j)\) we need to add a new vertex with label \(\mu(q_i)\) 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 expression \(R\) it is possible to construct a graph \(G\) such that \(L(G) = L(R)\).

**Theorem 9** A language \(L \subseteq X^+\) is representable by an expression in the algebra \(<2^{X^+}, \circ, \cup, \oplus, \varnothing, X>\) if and only if it can be generated by a vertex-labeled graph.

**Proof.** The necessary condition is proved in Theorem 1 and the sufficient condition is proved in Theorem 2.
5 Conclusion

In this work we introduced the algebra $\langle 2^X^+, \circ, \cup, \oplus, \emptyset, X \rangle$ which is similar to Kleene algebra. It has been shown that the properties of proposed algebra give us an opportunity to use it for analysis of vertex-labeled graphs in the same way as Kleene algebra is used for 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


