*Journal of Universal Computer Science, vol. 22, no. 11 (2016), 1437-1455*
*submitted: 30/5/16, accepted: 28/10/16, appeared: 1/11/16* © *J.UCS*

# All-Pairs Shortest Paths Algorithm for Regular 2D Mesh Topologies

**Vladimir Ciric**
(University of Niš, Faculty of Electronic Engineering, Niš, Serbia
vladimir.ciric@elfak.ni.ac.rs)

**Aleksandar Cvetkovic**
(University of Belgrade, Faculty of Mechanical Engineering, Belgrade, Serbia
acvetkovic@mas.bg.ac.rs)

**Ivan Milentijevic**
(University of Niš, Faculty of Electronic Engineering, Niš, Serbia
ivan.milentijevic@elfak.ni.ac.rs)

**Oliver Vojinovic**
(University of Niš, Faculty of Electronic Engineering, Niš, Serbia
oliver.vojinovic@elfak.ni.ac.rs)

**Abstract:** Motivated by the large number of vertices that future technologies will put in the front of path-search algorithms, and inspired by highly regular 2D mesh structures that exist in the domain applications, in this paper we propose a new all-pairs shortest paths algorithm, for any given regular 2D mesh topology, with complexity $\mathcal{O}(|\mathbf{V}|^2)$, where $|\mathbf{V}|$ is the number of vertices in the graph. The proposed algorithm can achieve better runtime than other known algorithms at the cost of narrowing the scope of the graphs that it can process to the graphs with regular 2D topology. The algorithm is developed into formalism by algebraic transformations in tropical algebra of the well-known Floyd-Warshall's algorithm. First we prove the equivalency of the Floyd-Warshall's algorithm and its tropical algebraic representation, and put the transformations of the algorithm into the algebraic domain. Secondly, having in mind the structure of the target class of graphs, we transform the original algorithm in the algebraic domain and develop a simple, low-complexity iterative algorithm for all-pairs shortest paths calculation. Decreasing of computational complexity can contribute to better exploitation of the algorithm in the wide range of applications from hardware design in new emerging technologies to big data problems in information technologies.

**Key Words:** algorithm design, graph algorithms, tropical algebra
**Category:** G.2.2, I.1.1, I.1.2, F.1.3, B.7.2, E.1

## 1 Introduction

One of the most studied problems in algorithmic graph theory is the problem of finding distances and paths in graphs. Many algorithms exists [Warshall 1962, Zwick 2001, Cormen et al. 2001, Zwick 2002, Hougardy 2010, Han et al. 2012]. The algorithms that calculate the distances of all vertices from one starting point are called Single-Source Shortest Paths (SSSP) algorithms, while the algorithms that give the distances of all vertex pairs are known as All-Pairs Shortest Paths (APSP) algorithms [Zwick 2001]. Regardless to SSSP and APSP classification, the algorithms can be further categorized as data-structure oriented, vertex-relaxation algorithms, such as well known Dijkstra's, Bellman-Ford's, or Johnson's algorithm [Cormen et al. 2001], and matrix based algorithms, like Warshall's "theorem on Boolean matrices", or Zvick's "rectangular matrix multiplication" [Zwick 2002]. These algorithms have worst case runtime from $\mathcal{O}(|\mathbf{V}|^3)$ for graphs with $|\mathbf{V}|$ vertices in the case of Floyd-Warshall's algorithm [Warshall 1962], through $\mathcal{O}\left(|\mathbf{V}|\left(|\mathbf{E}|+|\mathbf{V}|\log|\mathbf{V}|\right)\right)$ for Johnson's algorithm [Cormen et al. 2001], to $\mathcal{O}(|\mathbf{V}|^3 \log\log|\mathbf{V}| / \log^2|\mathbf{V}|)$ reported by Han et al. [Han et al. 2012], and $\mathcal{O}\left(|\mathbf{V}|^{2.376}\right)$ obtained by Zwick [Zwick 2002]. The reduced complexity is usually achieved at the cost of more complicated data structures, compared with relatively simple Warshall's algorithm [Hougardy 2010].

The latest development of technology introduced high requirements related to the number of vertices in search algorithms. The computational demands are getting higher and higher, starting with the emergence of "big trajectory data", as a subset of the big data problem, raised by world-wide population of GPS-equipped mobile devices and the data that they produce on a daily basis [Luo et al. 2013], until the error propagation analysis in emerging nanotechnology, where the number of devices that should be analyzed count $10^{12}$ or more devices [Stan et al. 2003, Ciric et al. 2009, Verma et al. 2016]. However, regular structures of some domain applications can be exploited in the algorithm design. For example, systolic arrays are computational modules with highly regular structures [Kung 1988, Parhi 1999]. Likewise, nanotechnology components are built of the huge number of devices ordered in highly regular mesh structures [Ciric et al. 2009]. The applications of nanotechnology that are under development include increasing of the density of memory chips, with a projected density of one terabyte of memory per square inch [Verma et al. 2016]. It is likely that such components will have highly regular 2-D mesh structures [DeHon 2002, Stan et al. 2003, Ciric et al. 2009, Verma et al. 2016].

Motivated by the large number of devices that can be fabricated using nanotechnology [Verma et al. 2016], and inspired by their regular 2D structures of the processing elements and interconnections [DeHon 2002, Stan et al. 2003, Ciric et al. 2013, Peng et al. 2016], with particular application in the computer architectures and systolic processing arrays design [Parhi 1999, Ciric et al. 2009, Ciric et al. 2010], in this paper we propose new all-pairs shortest paths algorithm, for any given regular 2D mesh topology, with worst runtime $\mathcal{O}(|\mathbf{V}|^2)$. The reduced complexity, compared to known algorithms, will be achieved under the assumption of regularity of graph topology. The algorithm will be developed into formalism by algebraic transformations in tropical algebra of the well-known Floyd-Warshall's algorithm, and it will be given as a generalization of the algorithm for error-propagation analysis within the array of hexagonal systolic multiplier [Ciric et al. 2013]. Tropical algebra will be chosen as a good candidate for algebraic representation [Simon 1988], which can enable algebraic transformations of the algorithm [Lu et al. 2012, Ciric et al. 2013]. It is a relatively new area in mathematics, considering the efforts that have been made since the late 1990s to consolidate the basic definitions of the theory [Simon 1988, Draisma 2008, Izhakian 2009, Izhakian and Rowen 2009]. We start with the proof of the equivalence of Floyd-Warshall's algorithm and its tropical algebraic representation, and proceed by putting the transformations of the algorithm into the algebraic domain. Having in mind the structure of the target class of graphs, we will transform the original algorithm in the algebraic domain and we will develop a simple, low-complexity iterative algorithm for all-pairs shortest paths calculation.

The paper is organized as follows. Section 2 gives a brief background on Warshall's transitive closure of Boolean matrices, and Floyd's shortest paths algorithm. In Sections 3 and 4 the introduction to tropical algebra is given, along with the equivalency proof of the tropical representation of Floyd-Warshall's algorithm. Section 5 is the main section where we present the development of a new iterative algorithm for shortest paths calculation of any given 2D mesh structure. Concluding remarks are given in Section 6.

## 2 Transitive Closure of Boolean Matrices and Shortest Paths Algorithm

Given a directed graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V} = \{v_1, \ldots, v_n\}$ is a finite set of vertices and $\mathbf{E}$ is a finite set of edges, an edge $e \in \mathbf{E}$ is an ordered pair $(v_i, v_j)$, where $v_i, v_j \in \mathbf{V}$ and an edge $(v_i, v_j)$ means that vertices $v_i$ and $v_j$ are connected. Let $v_i$ and $v_j$ be vertices and let $e_{i,j}$ denote $(v_i, v_j) \in \mathbf{E}$. A path is ordered subset of edges $\mathbf{P} \subset \mathbf{E}, \mathbf{P} = \{e_{i,k_1}, e_{k_1,k_2}, \ldots, e_{k_n,j}\}$, which connects nodes $v_i$ and $v_j$ through nodes $v_{k_1}, v_{k_2}, \ldots, v_{k_n}$. The path length is equal to the cardinality $|\mathbf{P}|$.

A Boolean adjacency matrix $M$ with elements $(m_{i,j})$ of graph $\mathbf{G}$ is defined as

$$m_{i,j} = \begin{cases} 1, e_{i,j} \in \mathbf{E} \\ 0, otherwise \end{cases} . \tag{1}$$

The dimensions of the matrix $M$ are $d$x$d$, where $d$ is cardinality of $\mathbf{V}$, i.e. $d = |\mathbf{V}|$.

Given two Boolean matrices $A$ and $B$, the Boolean product $A \bigwedge B$ is matrix whose $(i, j)$-th entry is $\bigvee_k (a_{i,k} \wedge b_{k,j})$ [Warshall 1962]. The Boolean sum $A \bigvee B$ is matrix whose $(i, j)$-th entry is $a_{i,j} \vee b_{i,j}$.

The construction for transitive closure $M^*$, well-known as Warshall's transitive closure of Boolean matrix, is the following [Warshall 1962]:

0. Set $M^* = M$.
1. Set $i = 1$.
2. $(\forall_j)$ if $m^*_{j,i} = 1$ then $(\forall k)$ set $m^*_{j,k} = m^*_{j,k} \vee m^*_{i,k}$.
3. Increment $i$ by 1.
4. If $i \leq d$, go to step 2; otherwise stop.

The construction gives the Boolean transitive closure matrix such that $m^*_{i,j} = 1$ if and only if either $m_{i,j} = 1$ or there exist integers $k_1, \ldots, k_n$ such that $m_{i,k_1} = m_{k_1,k_2} = \cdots = m_{k_n,j} = 1$; $m^* = 0$ otherwise [Warshall 1962]. In other words, $m^*_{i,j} = 1$ if there is an edge in graph $\mathbf{G}$ from node $v_i$ to $v_j$ $(e_{i,j})$, or there is a set of connected edges $e_{i,k_1}, e_{k_1,k_2}, \ldots, e_{k_n,j}$ that creates a path from node $v_i$ to $v_j$ via nodes $v_{k_1}, v_{k_2}, \ldots, v_{k_n}$.

In words, Warshall's algorithm fixes one vertex $v_i$ in the step 1, and calculates all possible paths with length equal to 2 that pass through the vertex $v_i$ (step 2), i.e. $v_j \rightarrow v_i \rightarrow v_k$, while updating the values $m_{j,k}$. The main idea which reduced the complexity from $\mathcal{O}(|\mathbf{V}|^4)$ to $\mathcal{O}(|\mathbf{V}|^3)$ is the fact that there is a recursion in the algorithm, because the value $m_{j,k}$ is reused in the next iterations in the form of $m_{j,i}$, and new edges $m_{i,k}$ are added to it one by one.

Let us consider the formulation of the algorithm proposed by Robert Floyd in [Floyd 1962]. Let graph $\mathbf{G_w}$ be a weighted graph, and let $w_{i,j} \in (\mathbb{R} \cup +\{\infty\})$ denote the weight of edge $e_{i,j}$. The elements of adjacency matrix $M_w$ of graph $\mathbf{G_w}$ are not Boolean values, but rather $m^w_{i,j} \in (\mathbb{R} \cup +\{\infty\})$, such that

$$m^w_{i,j} = \begin{cases} w_{i,j}, \, e_{i,j} \in \mathbf{E} \\ \infty, \quad otherwise \end{cases} . \tag{2}$$

In such a setting the Warshall's construction of Boolean transitive closure gets the form of Floyd's shortest paths algorithm [Floyd 1962]:

**Algorithm 1** R.Floyd's shortest paths algorithm "Algorithm 97" [Floyd 1962]

```
procedure ShortestPath(M)
    for i := 1 to d step 1 do
        for j := 1 to d step 1 do
            if m^w_{j,i} ≤ ∞ then
                for k := 1 to d step 1 do
                    if m^w_{i,k} ≤ ∞ then
                        s ← m^w_{j,i} + m^w_{i,k};
                        if s ≤ m^w_{j,k} then
                            m^w_{j,k} ← s;
```

Algorithm 1 transforms matrix $M_w$ into the shortest paths matrix, whose $(i, j)$-th entry is a sum of all weights

$$w_{i,k_1} + w_{k_1,k_2} + \cdots + w_{k_n,j}$$

of edges $e_{i,k_1}, e_{k_1,k_2}, \ldots, e_{k_n,j}$ on the shortest among all paths that exists between nodes $v_i$ and $v_j$. In the other words, the algorithm 1 computes shortest paths for all pairs of vertices in the given graph.

Both Warshall's and Floyd's algorithms [Floyd 1962, Warshall 1962] have the complexity $\mathcal{O}(|\mathbf{V}|^3)$, for the total of $|\mathbf{V}|^2$ possible vertex pairs.

## 3 The Tropical Representation of the Shortest Paths Algorithm

In order to introduce the algebraic representation of the algorithm, and enable the transformations of the algorithm in algebraic domain, we'll give a brief introduction to tropical semirings, and tropical representations of Floyd's and Warshall's algorithms.

The Min-Plus semiring is $\mathcal{M} = (\mathbb{N}_0 \cup \{\infty\}, \mathtt{min}, +)$. The sum in $\mathcal{M}$ is defined as minimum, and the product is usual addition. Note that $\infty$ is the zero of this semiring and 0 is its unit. This semiring was introduced by Simon in [Simon 1978], in the context of automata theory. Similar semiring was introduced by Mascale in [Mascle 1986]. He extended the set, and replaced the minimum with the maximum as $\mathcal{P} = (\mathbb{N}_0 \cup \{-\infty\}, \mathtt{max}, +)$. Leung [Leung 1988] proposed the semiring $\bar{\mathcal{M}} = (\mathbb{N}_0 \cup \{\omega, \infty\}, \mathtt{min}, +)$, where the minimum is defined with the respect to the order $0 \leq 1 \leq 2 \leq \cdots \leq \omega \leq \infty$, and addition of Min-Plus is completed by setting $x + \omega = \omega + x = \mathtt{max}\{x, \omega\}$ for all $x$. All these semirings are called tropical semirings. Other extensions include the tropical integers $\mathcal{Z} = (\mathbb{Z} \cup \{\infty\}, \mathtt{min}, +)$, the tropical rationals $\mathcal{Q} = (\mathbb{Q} \cup \{\infty\}, \mathtt{min}, +)$, and the tropical reals $\mathcal{R} = (\mathbb{R} \cup \{\infty\}, \mathtt{min}, +)$ [Pin 1998].

In the rest of the paper we will use the following definition.

**Definition 1 Min-Plus tropical algebra.** Min-Plus tropical algebra is the algebraic structure $(\mathbb{N}_0 \cup \{\infty\}, \mathtt{min}, +)$ with basic arithmetic operations of addition $(\oplus)$ and multiplication $(\odot)$ defined by

$$x \oplus y := \min(x, y), \quad x \odot y := x + y.$$

In words, the sum of two numbers is their minimum, and the product of two numbers is their usual sum. For example, the tropical sum of 4 and 9 is $4 \oplus 9 = 4$. The tropical product of 4 and 9 equals $4 \odot 9 = 13$.

Many of the familiar axioms of arithmetic remain valid in tropical settings [Izhakian 2009].

**Lemma 2.** *$(\mathbb{N}_0 \cup \{\infty\}, \oplus)$ and $(\mathbb{N}_0 \cup \{\infty\}, \odot)$ are commutative semigroups with units, and operation of $\odot$ is distributive in respect to $\oplus$.*

For instance we have:

$$x \oplus y = y \oplus x, \quad x \odot y = y \odot x.$$

These two arithmetic operations are also associative, and the times operator $\odot$ takes precedence when plus $\oplus$ and times $\odot$ occur in the same expression. The distributive law holds for tropical addition and multiplication:

$$x \odot (y \oplus z) = x \odot y \oplus x \odot z. \tag{3}$$

Both arithmetic operations have a neutral element. Infinity is the neutral element for addition and zero is the neutral element for multiplication:

$$x \oplus \infty = x, \quad x \odot 0 = x.$$

In order to express the results, we need extensions of the operations $\oplus$ and $\odot$ on matrices. We consider matrices, $A = [a_{i,j}]$, with the elements in $\mathbb{N}_0 \cup \{\infty\}$ [Izhakian 2009, Izhakian and Rowen 2009].

**Definition 3 Matrix addition.** Tropical addition, denoted $\oplus$, of matrices $A$ and $B$ is matrix $C = A \oplus B$ such that $c_{i,j} = a_{i,j} \oplus b_{i,j}$.

**Definition 4 Matrix multiplication.** Tropical product of matrices $A$, of type $N$x$K$, and $B$, of type $K$x$M$, is matrix $C = A \odot B = AB$, of type $N$x$M$, such that

$$c_{i,j} = \bigoplus_{k=1}^{K} (a_{i,k} \odot b_{k,j}), \tag{4}$$

where $i = 1, \dots, N$, $j = 1, \dots, M$, and the tropical sum $\bigoplus_{k=1}^{K} \nu_k$ represents the sum $\nu_1 \oplus \nu_2 \oplus \cdots \oplus \nu_K$.

From definition 4, the element $m^2_{i,j}$ of the matrix $M^2 = M \odot M$, is equal to

$$m^2_{i,j} = (m_{i,1} \odot m_{1,j}) \oplus (m_{i,2} \odot m_{2,j}) \oplus \cdots \oplus (m_{i,K} \odot m_{K,j}),$$

or in usual $(\mathbf{R} \cup \{\infty\}, +, \cdot)$ algebra

$$m^2_{i,j} = \min (m_{i,1} + m_{1,j}, m_{i,2} + m_{2,j}, \dots m_{i,K} + m_{K,j}),$$

which represents shortest path between vertices $v_i$ and $v_j$ from the graph $\mathbf{G}$ with length 2, in the case when the matrix $M$ is adjacency matrix of the graph $\mathbf{G}$.

The properties of the operations $\oplus$ and $\odot$ can be summarized in the following lemma.

**Lemma 5.** *Let $\mathcal{M}_n$ be the set of all matrices of type $N$x$M$ with elements in $\mathcal{R}\cup\{\infty\}$. $(\mathcal{M}_n, \oplus)$ and $(\mathcal{M}_n, \odot)$ are commutative semigroups with identity. Operation $\odot$ is distributive in the respect to $\oplus$.*

For example, matrix multiplication is associative operation, i.e.

$$A \odot (B \odot C) = (A \odot B) \odot C.$$

A neutral elements for tropical matrix addition and multiplication are zero matrix $O = [\infty]_{N\text{x}N}$ and identity matrix $I = [\delta_{i,j}]$, with elements

$$\delta_{i,j} = \begin{cases} 0, & i = j \\ \infty, & i \neq j \end{cases}.$$

As usual, we have $A \oplus O = A$, $A \odot O = O$, and $A \odot I = A$.

Let a tropical $k$-th power of matrix $M$ be matrix $M^k$ with elements $m^k_{i,j}$, where $M^1 = M$ and $M^{i+1} = M^i \odot M$.

Now we are in the position to state the algebraic representation of the shortest paths algorithm in tropical algebra.

**Theorem 6.** *Let $M$ be adjacency matrix of the graph $\mathbf{G}$. The computation of the matrix*

$$S = \bigoplus_{i=1}^{\infty} M^i, \tag{5}$$

*with elements $S = (s_{i,j})$, and the Floyd's algorithm 1 on the matrix $M$ are equivalent.*

Proof In order to prove the equivalency, we will substitute the computations from algorithm 1 with tropical equivalents.

Let us consider the central computational part of the algorithm 1. As the matrix $M$ is adjacency matrix of the weighted graph $\mathbf{G_w}$, in the 7th line of algorithm 1 the weight of the path from the vertex $v_j$ to the vertex $v_k$ that pases through exactly one additional vertex $v_i$ is computed. The computation from the 7th line can be represented in tropical algebra as

$$s = m^w_{j,i} \odot m^w_{i,k}. \tag{6}$$

After the execution of lines 8 and 9 in algorithm 1, $m^w_{j,k}$ will hold the minimum of the value previously contained in $m^w_{j,k}$ and the minimum given in (6), i.e.

$$m^w_{j,k} = \mathtt{min}\left\{m^w_{j,k}, s\right\} = m^w_{j,k} \oplus \left(m^w_{j,i} \odot m^w_{i,k}\right). \tag{7}$$

The weight $m^w_{j,k}$ is computed only if there is a path from the the vertex $v_i$ to the vertex $v_k$ (line 6 in algorithm 1), as well as if there is a path from the the vertex $v_j$ to the vertex $v_i$ (line 4 in algorithm 1). Due to the fact that $m^w_{p,q} = \infty$ if the path between vertices $v_p$ and $v_q$ doesn't exist, and the fact that $\infty$ is neutral element for tropical addition, the tropical equation (7) holds whether the values of $m^w_{j,i}$ and $m^w_{i,k}$ are less-than or equal $\infty$. Thus, algorithm 1 then becomes:

**Algorithm 2** The tropical form of algorithm 1

```
for i := 1 to d step 1 do
    for j := 1 to d step 1 do
        for k := 1 to d step 1 do
            m^w_{j,k} = m^w_{j,k} ⊕ (m^w_{j,i} ⊙ m^w_{i,k}) ;
```

We will show that algorithm 2 is equivalent to (5).

The equation (5) can be rewritten in iterative manner as

$$S^{(p+1)} = M \odot \left(I \oplus S^{(p)}\right), \tag{8}$$

where $S^1 = M$, and $p = 1, 2, \ldots$ represents the current iteration.

By direct application of definitions 3 and 4 to the tropical product (8) of matrices $M$ and $\left(I \oplus S^{(p)}\right)$, for the elements of the matrix $S^{(p+1)} = \left(s^{(p+1)}_{i,j}\right)$ we have:

$$\begin{aligned} s^{(p+1)}_{i,j} &= \bigoplus_{k=1}^{d} \left(m_{i,k} \odot \left(\delta_{k,j} \oplus s^{(p)}_{k,j}\right)\right) = \\ &= \bigoplus_{k=1}^{d} \left((m_{i,k} \odot \delta_{k,j}) \oplus \left(m_{i,k} \odot s^{(p)}_{k,j}\right)\right) = \\ &= \bigoplus_{k=1}^{d} \left(m_{i,j} \oplus \left(m_{i,k} \odot s^{(p)}_{k,j}\right)\right) = \\ &= m_{i,j} \oplus \bigoplus_{k=1}^{d} \left(m_{i,k} \odot s^{(p)}_{k,j}\right). \end{aligned} \tag{9}$$

If we assume that the elements $s^{(1)}_{i,j} = m_{i,j}$, the elements $s^{(p+1)}_{i,j}$ can be computed, according to (9), by the following algorithm:

**Algorithm 3** Computation of the elements $s_{i,j}^{(p)}$ according to (9)

1: **for** $p := 1$ `to` $\infty$ `step` 1 **do**
2:     **for** $i := 1$ `to` $d$ `step` 1 **do**
3:         **for** $j := 1$ `to` $d$ `step` 1 **do**
4:             **for** $k := 1$ `to` $d$ `step` 1 **do**
5:                 $s_{i,j}^{(p+1)} = m_{i,j} \oplus \left(m_{i,k} \odot s_{k,j}^{(p)}\right);$

If we accept the premise given in the construction of the Floyd's algorithm in [Floyd 1962] that there are no negative cycles in the graph $\mathbf{G}$, then the shortest paths in the graph $\mathbf{G}$ can include at most $d = |\mathbf{V}|$ nodes. This is because of the fact that a path with more than $d$ vertices must contain a loop (according to a premise, a positive cycle), which can't be the shortest path. The corollary is that the $p$-loop in the first line from algorithm 3 can be given as $p := 1$ `to` $d$, instead of 1 `to` $\infty$. In this case the complexity of algorithm 3 is finite and it is equal to $\mathcal{O}(|\mathbf{V}|^4)$.

It is obvious that the complexity of algorithm 3 is greater than the complexity of algorithm 2 for the order of the magnitude ($\mathcal{O}(|\mathbf{V}|^4)$ vs. $\mathcal{O}(|\mathbf{V}|^3)$). In order to avoid the need to iterate through the matrices $S^{(p)}$ and consequently reduce the complexity of algorithm 3, we will use the key Warshall's idea from [Warshall 1962] to store each newly computed value into the same variable. Thus, every new value $s_{i,j}^{(p)}, p = 1, 2, \ldots, d$, will be stored in the same variable $m_{i,j}$, overriding the previous intermediate value, as given in Algorithm 4.

**Algorithm 4** Noniterative form of algorithm 3

**for** $i := 1$ `to` $d$ `step` 1 **do**
    **for** $j := 1$ `to` $d$ `step` 1 **do**
        **for** $k := 1$ `to` $d$ `step` 1 **do**
            $m_{i,j} = m_{i,j} \oplus (m_{i,k} \odot m_{k,j})$ ;

The tropical operations are commutative, thus the loops in algorithm 4 can be sequenced as the loops from algorithm 2, which proves the theorem. □

## 4 Tropical GF(2) algebra and Warshall's algorithm

In order to put Warshall's algorithm for transitive closure computation of Boolean matrices into the tropical framework, we need to define tropical algebra over GF(2) $= (\{0, 1\}, +, \cdot)$, where $+$ and $\cdot$ are addition and multiplication by module 2. Tropicalization of this algebraic structure is performed by the following definition.

**Definition 7.** For every $x, y \in \{0,1\}$ we define

$$x \oplus y = \max\{x, y\} = x \vee y$$

and

$$x \odot y = x \cdot y = x \wedge y.$$

Next lemma shows that algebraic structure $\mathcal{G} = (\{0,1\}, \oplus, \odot)$ is commutative algebra with units.

**Lemma 8.** *$(\{0,1\}, \oplus)$ and $(\{0,1\}, \odot)$ are commutative semigroups with units, and operation $\odot$ is distributive in respect to $\oplus$.*

Since $\oplus$ and $\odot$ are algebraic operations of conjunction and disjunction, we know that the operations are associative, commutative, and the unit for conjunction is 1, and for disjunction the unit is 0. Furthermore, distributive law holds for $\odot$ in respect to $\oplus$.

Now we are in the position to prove the equivalency of Warshall's algorithm over Boolean adjacency matrix and the tropical representation (5).

Using definitions 3 and 4 we introduce matrix operations over the tropical algebra $\mathcal{G}$. Also, we use the well know abbreviation for the matrix exponentiation.

**Theorem 9.** *Let $M$ be adjacency matrix of the graph* $\mathbf{G}$. *The computation of the matrix*

$$S = \bigoplus_{i=1}^{\infty} M^i, \tag{10}$$

*with elements $S = (s_{i,j})$, and the Warshall's algorithm for finding the transitive closure of $M$ are equivalent.*

PROOF We note that the step 2 in the Warshall's algorithm

$$(\forall_j) \text{ if } m^*_{j,i} = 1 \text{ then } (\forall k) \text{ set } m^*_{j,k} = m^*_{j,k} \vee m^*_{i,k} \tag{11}$$

can be rewritten as

$$(\forall_j)(\forall k) \text{ set } m^*_{j,k} = m^*_{j,k} \vee \left(m^*_{j,i} \wedge m^*_{i,k}\right), \tag{12}$$

having in mind that condition $\left(\text{if } m^*_{j,i} = 1 \text{ then}\right)$ represents neutral element for the operation $\wedge$.

We recognize that the algorithm has the same structure as the Floyd's algorithm, but over the tropical algebra $\mathcal{G}$. $\square$

By introducing the algebraic representation (5) we enabled transformations of the algorithm using familiar algebraic axioms, and we introduced the possibility to transform the algorithm and further reduce its complexity by exploiting the properties of the domain application. In the next section we will demonstrate how the complexity of the original algorithm can be reduced by algebraic manipulations of algorithm representation.

## 5 The development of All-Pairs Shortest Paths Algorithm in tropical domain

In [Ciric et al. 2013], the shortest paths algorithm that strongly relays on only one particular array structure, namely orthogonal array for integers multiplication, is developed with complexity $\mathcal{O}\left(\left|\mathbf{V}^2\right|\right)$. Here we will consider a general case of any given regular 2D mesh topology, using the proposed algebraic representation (5).

First, we define 2D mesh topology, and related terms of rows and columns of vertices.

Let $\mathbf{G}$ be an oriented graph, and let $\mathbf{G}_1 = (\mathbf{V}_1, \mathbf{E}_1)$ and $\mathbf{G}_2 = (\mathbf{V}_2, \mathbf{E}_2)$ be oriented subgraphs of the graph $\mathbf{G}$. We define graph $\widehat{\mathbf{G}_{1,2}} = \mathbf{G}_1 \nabla \mathbf{G}_2$ to be an oriented subgraph of the graph $\mathbf{G}$ with the set of vertices $\widehat{\mathbf{V}_{1,2}} = \mathbf{V}_1 \cup \mathbf{V}_2$, and the set of edges $\widehat{\mathbf{E}_{1,2}} = \{e_{i,j} | v_i, v_j \in \mathbf{V}_1 \cup \mathbf{V}_2\}$, and $\mathbf{G}_{1,2} = \mathbf{G}_1 \cup \mathbf{G}_2$ to be an oriented subgraph of the graph $\mathbf{G}$ with the set of vertices $\mathbf{V}_{1,2} = \mathbf{V}_1 \cup \mathbf{V}_2$, and the set of edges $\mathbf{E}_{1,2} = \mathbf{E}_1 \cup \mathbf{E}_2$.

In words, $\widehat{\mathbf{G}_{i,j}} = \mathbf{G}_i \nabla \mathbf{G}_j$ represents the union of the subgraphs $\mathbf{G}_i$ and $\mathbf{G}_j$, including edges from the graph $\mathbf{G}$ that connect them, and $\mathbf{G}_{i,j} = \mathbf{G}_i \cup \mathbf{G}_j$ represents the regular union of the graphs $\mathbf{G}_i$ and $\mathbf{G}_j$ without any additional edges between them.

Let the difference between graphs $\mathbf{G}_p = (\mathbf{V}_p, \mathbf{E}_p)$ and $\mathbf{G}_q = (\mathbf{V}_q, \mathbf{E}_q)$ be defined as a graph $\mathbf{G}_p \setminus \mathbf{G}_q = (\mathbf{V}_p \setminus \mathbf{V}_q, \mathbf{E}_p \setminus \mathbf{E}_q)$.

**Definition 10 The connectivity of subgraphs.** Two disjoint oriented subgraphs $\mathbf{G}_i$ and $\mathbf{G}_j$ are connected if and only if $\widehat{\mathbf{G}_{i,j}} \setminus \mathbf{G}_{i,j} \neq \emptyset$. The connectivity is oriented from $\mathbf{G}_i$ to $\mathbf{G}_j$ if all edges from $\widehat{\mathbf{G}_{i,j}} \setminus \mathbf{G}_{i,j} \neq \emptyset$ are oriented from $\mathbf{G}_i$ to $\mathbf{G}_j$.

Motivated by the extremely large number of devices that can be fabricated using nanotechnology [Verma et al. 2016], and inspired by their highly regular 2D mesh structures of the processing elements and interconnections [DeHon 2002, Stan et al. 2003, Ciric et al. 2013, Peng et al. 2016], we give the following definition.

**Definition 11 Graph with regular 2D mesh topology.** Let $\mathbf{G}$ be a directed graph. The graph $\mathbf{G}$ has a regular 2D mesh topology if and only if there exists disjoint partition $\mathbf{G}_i, i = 1, 2, \ldots, R$, each with $C$ vertices, where we call every subgraph $\mathbf{G}_i$ the $i$-th row of the graph $\mathbf{G}$, such that:

1. (isomorphism of rows) all rows are isomorphic,
2. (connectivity of neighboring rows) for every row $k$, except one row, there exists row $i$ such that the row $i$ has directed connectivity to row $k$; for every row $k$, except one,there exists row $j$ such that the row $k$ has directed connectivity to row $j$,
3. (isomorphism of edges that connect rows) there exists graph $\mathbf{H}$ such that for every $i$ and $j, i \neq j$ , $\widehat{\mathbf{G}_{i,j}}$ is isomorphic to $\mathbf{H}$ or $\mathbf{G}_{i,j}$.

The total number of vertices in $\mathbf{G}$ is $|\mathbf{V}| = R \cdot C$.

The example of graph $\mathbf{G}$ with arbitrarily 2D mesh structure and corresponding adjacency matrix $A$ is shown in Fig. 1. The graph $\mathbf{G}$ in the example from Fig. 1 has four subgraphs $\mathbf{G}_i, i = 1, 2, 3, 4$, i.e. rows, where each row has three vertices. We have $\mathbf{G}_1 = (\{v_1, v_2, v_3\}, \{e_{1,2}, e_{2,3}, e_{1,3}, e_{2,1}\})$, $\mathbf{G}_2 = (\{v_4, v_5, v_6\}, \{e_{4,5}, e_{5,6}, e_{4,6}, e_{5,4}\})$, $\mathbf{G}_3 = (\{v_7, v_8, v_9\}, \{e_{7,8}, e_{8,9}, , e_{7,9}, e_{8,7}\})$, $\mathbf{G}_4 = (\{v_{10}, v_{11}, v_{12}\}, \{e_{10,11}, e_{11,12}, , e_{10,12}, e_{11,10}\})$, $\mathbf{H} \approx \widehat{\mathbf{G}_{1,2}} \approx \widehat{\mathbf{G}_{2,3}} \approx \widehat{\mathbf{G}_{3,4}} = (\{v_1, \quad v_2, v_3, v_4, v_5, v_6\}, \{e_{1,2}, e_{2,3}, e_{1,3}, e_{2,1}, e_{4,5}, e_{5,6}, e_{4,6}, e_{5,4}, e_{1,4}, e_{2,6}, e_{3,6}\})$, $\mathbf{G}_{1,2} \approx \mathbf{G}_{2,3} \approx \mathbf{G}_{3,4} = (\{v_1, v_2, v_3, v_4, v_5, v_6\}, \{e_{1,2}, e_{2,3}, e_{1,3}, e_{2,1}, e_{4,5}, e_{5,6}, e_{4,6}, e_{5,4}\})$.

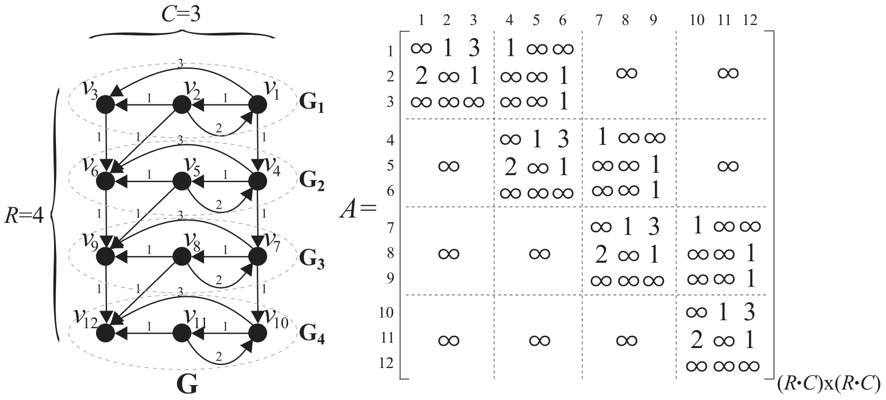


Figure 1: The example of the graph $\mathbf{G}$ (left) and corresponding adjacency matrix $A$ (right)

Let $\mathbf{G}_i$ be the $i$-th row of the graph $\mathbf{G}$, and let $A_R$ be an adjacency matrix of type $C$x$C$ of the row $\mathbf{G}_i$. Also, let $A_C$ represent an adjacency matrix containing the edges $e_{i,j}$, which connect rows $i$ and $i+1$.

**Lemma 12.** *The graph* $\mathbf{G}$ *with regular 2D mesh topology has an adjacency matrix* $A = [A_{p,q}]$*, where* $A_{p,q}$ *is a submatrix of type* $C\text{x}C$ *at the position* $(p,q)$*. The submatrices* $A_{p,q}$ *have the following form:*

$$A_{p,q} = \begin{cases} A_R, p = q \\ A_C, p = q-1 \\ O, \quad otherwise \end{cases}, \tag{13}$$

*i.e.*

$$A(\mathbf{G}) = \begin{bmatrix} A_R & A_C & O & \cdots & O \\ O & A_R & A_C & & O \\ O & O & A_R & & O \\ \vdots & & & \ddots & \\ O & O & O & & A_R \end{bmatrix}_{(R\cdot C)\text{x}(R\cdot C)}, \tag{14}$$

PROOF Directly from the isomorphism (1) of the definition 11 we have that all submatrices $A_{p,p}$ of the matrix $A$ are equal to $A_R$. From the properties (2) and (3) given in definition 11 we have that $A_{p,p+1}$ of the matrix $A$ are equal to $A_C$, and all other $A_{p,q}$, $q \neq p, p+1$, are $O$. □

The submatrices $A_{p,q}$ of the adjacency matrix $A$ are square matrices of type $C\text{x}C$. There are total of $R$ submatrices $A_{p,q}$ in each row, and $R$ submatrices $A_{p,q}$ in each column of the matrix $A$, of the type $R \cdot C$ x $R \cdot C$.

In order to obtain the shortest paths matrix according to Theorem 6, we can use the structure of the adjacency matrix (14) to obtain the matrix $A^k$. The matrix $A^k$ can be obtained in iterative manner, which we give in the following lemma.

**Lemma 13.** *Let* $A = [A_{p,q}]$ *given with (14) be the adjacency matrix of the graph* $\mathbf{G}$*, where* $A_{p,q}$*,* $p = 1, 2, \ldots, R$*,* $q = 1, 2, \ldots, R$ *is the submatrix at the position* $(p,q)$ *within the matrix* $A$*. The submatrix* $A^{k+1}_{p,q}$ *of the matrix* $A^{k+1} = \left[A^{k+1}_{p,q}\right]$ *can be obtain using the following iterative formula:*

$$A^{k+1}_{p,q} = \begin{cases} A^k_{p,q} \odot A_R, & p = q \\ \left(A^k_{p,q-1} \odot A_C\right) \oplus \left(A^k_{p,q} \odot A_R\right), & p \leq q-1 \\ O, & p > q \end{cases}. \tag{15}$$

PROOF For any given matrix $A^k = \left[A^k_{p,q}\right]$, from (13) and the definition and properties of matrix multiplication, the iterative formula (15) directly follows. □

From Theorem 6 and the properties of matrix multiplication we have

$$S_{p,q} = \bigoplus_{i=1}^{\infty} A^i_{p,q}. \tag{16}$$

Using (15) and (16) it is straightforward to show that the shortest paths matrix $S(\mathbf{G})$ has the following form:

$$S(\mathbf{G}) = \begin{bmatrix} S_0 & S_1 & S_2 & \cdots & S_{R-1} \\ O & S_0 & S_1 & & S_{R-2} \\ O & O & S_0 & & S_{R-3} \\ \vdots & & & \ddots & \\ O & O & O & & S_0 \end{bmatrix}_{(R\cdot C)\text{x}(R\cdot C)} . \tag{17}$$

The submatrices $S_m, m = 0, 1, \ldots, R-1$ from (17), where $m = q - p$, are of the same type $C$x$C$ as the submatrices $A_{p,q}$.

We give a general method for calculation of the submatrices $S_m$ of any given graph $\mathbf{G}$ that satisfies the terms of definition 11, in the form of the following theorem.

**Theorem 14.** *The submatrix $S_m, m = q-p$, of the shortest paths matrix $S(\mathbf{G}) = [S_{p,q}]$ of any given graph $\mathbf{G}$ with regular 2D mesh topology can be obtained as*

$$S_m = S_{m-1} A_C \left(I \oplus S_0\right), \tag{18}$$

*where $m = 2, 3, \ldots, R-1$,*

$$S_0 = \bigoplus_{k=1}^{C} A_R^k, \tag{19}$$

*and*

$$S_1 = A_C \oplus A_C S_0 \oplus S_0 A_C \oplus S_0 A_C S_0. \tag{20}$$

Proof  For $p = q$, i.e. $m = 0$, from (15) we have

$$A_0^{k+1} = A_0^k \odot A_R, \tag{21}$$

which, after substitution in (16), proves (19):

$$S_0 = \bigoplus_{k=1}^{\infty} A_0^k = \bigoplus_{k=1}^{\infty} A_R^k = \bigoplus_{k=1}^{C} A_R^k. \tag{22}$$

For $p \leq q-1$, i.e. $m \geq 1$, from (16) we have

$$S_m = \bigoplus_{k=1}^{\infty} A_m^k = A_m^1 \oplus \bigoplus_{k=1}^{\infty} A_m^{k+1}, \tag{23}$$

and from (15) we have

$$\begin{aligned} S_m &= A_m \oplus \bigoplus_{k=1}^{\infty} A_m^{k+1} = \\ &= A_m \oplus \bigoplus_{k=1}^{\infty} \left(\left(A_{m-1}^k A_C\right) \oplus \left(A_m^k A_R\right)\right) = \\ &= A_m \oplus \left(\bigoplus_{k=1}^{\infty} A_{m-1}^k\right) A_C \oplus \left(\bigoplus_{k=1}^{\infty} A_m^k\right) A_R = \\ &= A_m \oplus S_{m-1} A_C \oplus S_m A_R. \end{aligned} \tag{24}$$

If we recursively unfold the form (24), we get

$$\begin{aligned} S_m \;&=\; A_m \oplus S_{m-1} A_C \oplus \underbrace{\left(A_m \oplus S_{m-1} A_C \oplus S_m A_R\right)}_{S_m} A_R = \\ &=\; \left(A_m \oplus A_m A_R\right) \oplus \left(S_{m-1} A_C \oplus S_{m-1} A_C A_R\right) \oplus S_m A_R^2 = \\ &\dots \\ &=\; A_m \left(I \oplus \bigoplus_{k=1}^{\infty} A_R^k\right) \oplus S_{m-1} A_C \left(I \oplus \bigoplus_{k=1}^{\infty} A_R^k\right) \oplus S_m A_R^{\infty}. \end{aligned} \tag{25}$$

As matrix $A_R^{\infty}$ contains all paths with length equal to $\infty$ from any subgraph $\mathbf{G}_i, i = 1, 2, \dots, R$, we have $A_R^{\infty} = O$, thus (25) becomes

$$S_m = A_m \left(I \oplus S_0\right) \oplus S_{m-1} A_C \left(I \oplus S_0\right). \tag{26}$$

For $m = 1$, from (14) we have $A_1 = A_C$, thus

$$\begin{aligned} S_1 &= A_C \left(I \oplus S_0\right) \oplus S_0 A_C \left(I \oplus S_0\right) = \\ &= A_C \oplus A_C S_0 \oplus S_0 A_C \oplus S_0 A_C S_0, \end{aligned} \tag{27}$$

which proves (20). For $m = 2, 3, \dots, R-1$ we have $A_m = O$, thus (26) becomes (18), which proves the theorem for $m \geq 2$. □

Theorem 14 gives the calculation method for the shortest paths matrix $S(\mathbf{G})$ of any given graph $\mathbf{G}$ with regular 2D mesh topology. Algorithm 5 directly interprets the result of the Theorem 14.

**Algorithm 5** The shortest paths algorithm for regular 2D mesh

1: $S_0 = A_R$;
2: **for** $i := 2$ `to` $C$ **do**
3: $\quad S_0 = S_0 \oplus \left(S_0 \odot A_R\right)$;
4: $S_1 = A_C \oplus A_C S_0 \oplus S_0 A_C \oplus S_0 A_C S_0$;
5: $T_0 = A_C \odot \left(I \oplus S_0\right)$;
6: **for** $i := 2$ `to` $R-1$ **do**
7: $\quad S_i = S_{i-1} \odot T_0$;

Using algorithm 5 for the graph $\mathbf{G}$ from Fig. 1, we obtain the shortest paths matrix as

$$S(\mathbf{G}) = \left[\begin{array}{ccc|ccc|ccc|ccc} 3 & 1 & 2 & 1 & 2 & 2 & 2 & 3 & 3 & 3 & 4 & 4 \\ 2 & 3 & 1 & 3 & 4 & 1 & 4 & 5 & 2 & 5 & 6 & 3 \\ \infty & \infty & \infty & \infty & \infty & 1 & \infty & \infty & 2 & \infty & \infty & 3 \\ \hline \infty & \infty & \infty & 3 & 1 & 2 & 1 & 2 & 2 & 2 & 3 & 3 \\ \infty & \infty & \infty & 2 & 3 & 1 & 3 & 4 & 1 & 4 & 5 & 2 \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 1 & \infty & \infty & 2 \\ \hline \infty & \infty & \infty & \infty & \infty & \infty & 3 & 1 & 2 & 1 & 2 & 2 \\ \infty & \infty & \infty & \infty & \infty & \infty & 2 & 3 & 1 & 3 & 4 & 1 \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 1 \\ \hline \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 3 & 1 & 2 \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 2 & 3 & 1 \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \end{array}\right] \tag{28}$$

We will evaluate the complexity of the algorithm 5 by considering the complexity of matrix multiplication and addition. Having in mind that the type of the submatrices $A_{p,q}$ is $C$x$C$, the complexity of matrix multiplication is $\mathcal{O}\left(C^3\right)$, while the complexity of matrix addition is $\mathcal{O}\left(C^2\right)$. Thus, the overall complexity of the Algorithm 5 is

$$\mathcal{O}\left(\underbrace{(C-1)\left(C^3+C^2\right)}_{\text{lines 2 and 3}}+\underbrace{4C^3+3C^2}_{\text{line 4}}+\underbrace{C^2+C^3}_{\text{line 5}}+\underbrace{(R-2)\left(C^3\right)}_{\text{lines 6 and 7}}\right) = \\ = \mathcal{O}\left(C^4+C^3R\right). \tag{29}$$

If we denote the aspect ratio of the rows and columns as $\alpha = C/R$, having in mind that $|\mathbf{V}| = R \cdot C$, from (29) we get for the complexity

$$\mathcal{O}\left(\left(\alpha^2+\alpha\right)|\mathbf{V}|^2\right). \tag{30}$$

Due to the fact that the algorithm complexity (30) depends on the graph $\mathbf{G}$ proportions, we will evaluate the ratio ($\alpha$) of the rows ($R$) and columns ($C$) for which the algorithm 5 has better runtime than widely used Johnson's algorithm [Cormen et al. 2001]. The Johnson's algorithm has the complexity $\mathcal{O}\left(|\mathbf{V}|\left(|\mathbf{E}|+|\mathbf{V}|\log|\mathbf{V}|\right)\right)$. The maximum number of edges within the regular graph from Def. 11 is

$$|\mathbf{E}| = \underbrace{C^2 \cdot R}_{\text{max.edgs.in a row x num.of.rows}} + \underbrace{C^2 \cdot (R-1)}_{\text{edgs. between the rows}} = \mathcal{O}\left(C^2 \cdot R\right).$$

Having in mind that $\alpha = C/R$ and $|\mathbf{V}| = R \cdot C$, the maximum number of edges can be expressed as $|\mathbf{E}| = \mathcal{O}\left(\sqrt{\alpha} \cdot |\mathbf{V}|^{3/2}\right)$ Thus, the complexity of the

Johnson's algorithm is $\mathcal{O}\left(|\sqrt{\alpha} \cdot \mathbf{V}|^{2.5}\right)$. We will set this as a boundary for the proposed algorithm as

$$\mathcal{O}\left(\left(\alpha^2 + \alpha\right) |\mathbf{V}|^2\right) < \mathcal{O}\left(\sqrt{\alpha} \cdot |\mathbf{V}|^{2.5}\right). \tag{31}$$

The inequality (31) is a cubic inequality of the following form:

$$\left(\sqrt{\alpha}\right)^3 + \sqrt{\alpha} - \sqrt{|\mathbf{V}|} \leq 0, \tag{32}$$

which has two imaginary and one real root. Fig. 2 shows the area in $|\mathbf{V}|-\alpha$ space, bounded by the real root of (32), i.e. the area where the proposed algorithm has better runtime then $\mathcal{O}\left(\sqrt{\alpha} \cdot |\mathbf{V}|^{2.5}\right)$.

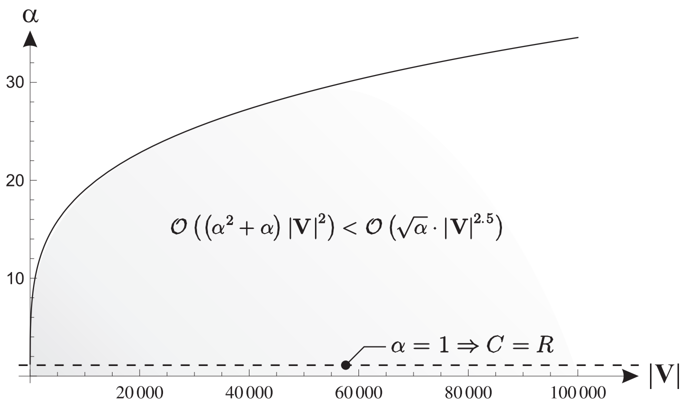


Figure 2: The ratio $\alpha$ for which the proposed algorithm has better runtime then Johnson's algorithm

For example, if $\alpha = 1$ the complexity of the proposed algorithm is equal to $\mathcal{O}\left(|\mathbf{V}|^2\right)$, and the complexity of the Johnson's algorithm is $\mathcal{O}\left(|\mathbf{V}|^{2.5}\right)$. As $\alpha$ represents the ratio between the number of rows ($R$) and columns ($C$) of the graph $\mathbf{G}$ with $|\mathbf{V}| = R \cdot C$ vertices, $\alpha = 1$ represents the special case of the "square" graph $\mathbf{G}$ with $C = R$, typical for nano-FPGA arrays (nFPGA) [Dong et al. 2007]. This case is denoted with dashed line in Fig. 2.

If we choose a graph $\mathbf{G}$ with $\alpha$ which is outside of the area illustrated in Fig. 2, i.e. $R \in \Theta(1)$, then we have $C \in \Theta(|\mathbf{V}|)$ and $\alpha \in \Theta(|\mathbf{V}|)$, thus (30) leads to a running time of $\mathcal{O}\left(|\mathbf{V}|^4\right)$. However, it is straightforward to show that Def. 11 holds when the graph is transposed, as well. This is due to the isomorphism of both rows and columns, introduced in Def. 11. Hence, if the ratio of the rows and columns ($\alpha$) of the graph gives worse runtime, the graph can be simply transposed, which reduces the complexity back to the values below $\mathcal{O}\left(\sqrt{\alpha}\cdot|\mathbf{V}|^{2.5}\right)$.

## 6 Conclusion

In this paper we proposed a new all-pairs shortest paths algorithm, for any given regular 2D mesh topology, with worst case complexity of $\mathcal{O}(|\mathbf{V}|^2)$. The algorithm is developed into formalism by algebraic transformations in tropical algebra of the well-known Floyd-Warshall's algorithm. We explored the possibility to transform the algorithm and provide further reduction of its complexity by exploiting the properties of the domain application. The proposed algorithm can achieve better runtime than other known algorithms at the cost of narrowing the scope of the graphs that it can process to the graphs with regular 2D topology. First, we proved the equivalency of the Floyd-Warshall's algorithm and its tropical algebraic representation, and put the transformations of the algorithm into the algebraic domain. Secondly, having in mind the structure of the target class of graphs, we transformed the original algorithm in the algebraic domain and developed a simple, low-complexity iterative algorithm for all-pairs shortest paths calculation.

## References

[Ciric et al. 2009] Ciric, Vladimir and Kolokotronis, Jelena and Milentijevic, Ivan: "Partial error tolerance for bit-plane FIR filter architecture"; Elsevier, AEU-International Journal of Electronics and Communications 63, 5 (2009), 398-405.

[Ciric et al. 2010] Ciric, Vladimri and Cvetkovic, Aleksandar and Milentijevic, Ivan: "Yield analysis of partial defect tolerant bit-plane array"; Elsevier, Computers & mathematics with applications 59, 1 (2010), 98-107.

[Ciric et al. 2013] Ciric, Vladimir and Cvetkovic, Aleksandar and Simic, Vladimir and Milentijevic, Ivan: "Tropical algebra based framework for error propagation analysis in systolic arrays"; Elsevier, Applied Mathematics and Computation 225 (2013), 512-525.

[Cormen et al. 2001] Cormen, Thomas H., et al. "Introduction to Algorithms, MIT Press and McGraw-Hill, 2001." ISBN 262032937 (2001): 636-640.

[DeHon 2002] DeHon, Andre. "Array-based architecture for molecular electronics." Proceedings of the First Workshop on Non-Silicon Computation (NSC-1). 2002.

[Dong et al. 2007] Dong, Chen, et al. "3-D nFPGA: A reconfigurable architecture for 3-D CMOS/nanomaterial hybrid digital circuits." IEEE Transactions on Circuits and Systems I: Regular Papers 54.11 (2007): 2489-2501.

[Draisma 2008] Draisma, Jan: "A tropical approach to secant dimensions"; Journal of Pure and Applied Algebra 212, 2 (2008), 349-363.

[Floyd 1962] Floyd, Robert W.: "Algorithm 97: Shortest Path"; ACM, Commun. ACM 5, 6 (June 1962), 345-345.

[Han et al. 2012] Han, Yijie, and Takaoka, Tadao. "An O (n 3 loglogn/log2 n) Time Algorithm for All Pairs Shortest Paths." Algorithm TheorySWAT 2012. Springer Berlin Heidelberg, 2012. 131-141.

[Hougardy 2010] Hougardy, Stefan: "The Floyd–Warshall algorithm on graphs with negative cycles"; Elsevier, Information Processing Letters 110, 8 (2010), 279-281.

[Izhakian 2009] Izhakian, Zur: "Tropical Arithmetic and Matrix Algebra"; Communications in Algebra 37, 4 (2009), 1445-1468.

[Izhakian and Rowen 2009] Izhakian, Zur and Rowen, Louis : "The Tropical Rank of a Tropical Matrix"; Communications in Algebra 37, 11 (2009), 3912-3927.

[Kung 1988] Kung, Sun Yuan. "VLSI array processors." Englewood Cliffs, NJ, Prentice Hall, 1988, 685 p. Research supported by the Semiconductor Research Corp., SDIO, NSF, and US Navy. 1 (1988).

[Leung 1988] Leung, Hing: "On the topological structure of a finitely generated semigroup of matrices"; Springer 37, 1 (1988), 273-287.

[Lu et al. 2012] Lu, Qi and Madsen, Michael and Milata, Martin and Ravn, Søren and Fahrenberg, Uli and Larsen, Kim G.: "The Floyd–Warshall algorithm on graphs with negative cycles"; Elsevier, The Journal of Logic and Algebraic Programming 81, 3 (2012), 298-313.

[Luo et al. 2013] Luo, Wuman and Tan, Haoyu and Chen, Lei and Ni, Lionel: "Finding time period-based most frequent path in big trajectory data"; ACM, Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (2013), 713-724.

[Mascle 1986] Mascle, Jean-Paul: "Torsion matrix semigroups and recognizable transductions"; Springer (1986).

[Parhi 1999] Parhi, Keshab K. "VLSI Digital Signal Processing Systems: Design and Implementation". John Wiley & Sons, 1999.

[Peng et al. 2016] Peng, Kang, et al. "Emerging Parallel Dual 2D Composites: Natural Clay Mineral Hybridizing MoS2 and Interfacial Structure." Advanced Functional Materials (2016).

[Pin 1998] Pin, Jean-Eric: "Tropical Semirings"; Cambridge Univ. Press, Publ. Newton Inst. 11, Idempotency (1998), 50-69.

[Simon 1978] Simon, Imre: "Limited Subsets of a Free Monoid"; Piscataway, N.J., Proc. 19th Annual Symposium on Foundations of Computer Science (1978), 143150.

[Simon 1988] Imre Simon: "Recognizable sets with multiplicities in the tropical semiring"; Springer Berlin Heidelberg, Mathematical Foundations of Computer Science 324 (1988), 107-120.

[Stan et al. 2003] Stan, Mircea R., et al. "Molecular electronics: From devices and interconnect to circuits and architecture." Proceedings of the IEEE 91.11 (2003): 1940-1957.

[Verma et al. 2016] Verma, Saurabh, Khangarat Vijaysingh, and Rampukar Kushwaha. "Nanotechnology: A Review." (2016).

[Warshall 1962] Warshall, Stephen: "A Theorem on Boolean Matrices"; ACM, J. ACM 9, 1 (Jan. 1962), 11-12.

[Zwick 2001] Zwick, Uri. "Exact and approximate distances in graphsa survey." AlgorithmsESA 2001. Springer Berlin Heidelberg, 2001. 33-48.

[Zwick 2002] Zwick, Uri. "All pairs shortest paths using bridging sets and rectangular matrix multiplication." Journal of the ACM (JACM) 49.3 (2002): 289-317.