ISSN 2579-2784 (Print)
ISSN 2538-2788 (Online)

MATHEMATICAL PROBLEMS OF COMPUTER SCIENCE

LIX

Yerevan
2023

Հայաստանի Հանրապետության Գիտությունների ազգային ակադեմիայի
Ինֆորմատիկայի և ավտոմատացման պրոբլեմների ինստիտուտ

Институт проблем информатики и автоматизации Национальной академии наук Республики Армения

Institute for Informatics and Automation Problems of the National Academy of Sciences of the Republic of Armenia

Կոմպյուտերային գիտության մաթեմատիկական խնդիրներ

Математические проблемы компьютерных наук

Mathematical Problems of Computer Science

LIX

ՀՐԱՏԱՐԱԿՎԱԾ Է ՀՀ ԳԱԱ ԻՆՖՈՐՄԱՏԻԿԱՅԻ ԵՎ ԱՎՏՈՄԱՏԱՑՄԱՆ ՊՐՈԲԼԵՄՆԵՐԻ ԻՆՍՏԻՏՈՒՏԻ ԿՈՂՄԻՑ
ОПУБЛИКОВАНО ИНСТИТУТОМ ПРОБЛЕМ ИНФОРМАТИКИ И АВТОМАТИЗАЦИИ НАН РА
PUBLISHED BY THE INSTITUTE FOR INFORMATICS AND AUTOMATION PROBLEMS OF NAS RA

ԵՐԵՎԱՆ 2023 YEREVAN

Կոմպյուտերային գիտության մաթեմատիկական խնդիրներ պարբերականը հրատարակվում է տարեկան երկու անգամ ՀՀ ԳԱԱ Ինֆորմատիկայի և ավտոմատացման պրոբլեմների ինստիտուտի (ԻԱՊԻ) կողմից: Այն ընդգրկում է տեսական և կիրառական մաթեմատիկայի, ինֆորմատիկայի և հաշվողական տեխնիկայի ժամանակակից ուղղությունները:

Այն ընդգրկված է Բարձրագույն որակավորման հանձնաժողովի ընդունելի ամսագրերի ցանկում:

Տպագրվում է Խմբագրական խորհրդի 2023թ. մայիսի 25-ի N 23-05/1 նիստի որոշման հիման վրա

ISSN 2579-2784 (Print)
ISSN 2738-2788 (Online)

Журнал **Математические проблемы компьютерных наук** издается два раза в год Институтом проблем информатики и автоматизации НАН РА. Он охватывает современные направления теоретической и прикладной математики, информатики и вычислительной техники.

Он включен в список допустимых журналов Высшей квалификационной комиссии.

Печатается на основании решения N 25-05/1 заседания
Редакционного совета от 25 мая 2023г.

ISSN 2579-2784 (Print)
ISSN 2738-2788 (Online)

Mathematical Problems of Computer Science, LIX

The periodical **Mathematical Problems of Computer Science** is published twice per year by the Institute for Informatics and Automation Problems of NAS RA. It covers modern directions of theoretical and applied mathematics, informatics and computer science.

It is included in the list of acceptable journals of the Higher Qualification Committee.

Printed on the basis of decision N 25-05/1 of the session of the Editorial Council dated May 25, 2023.

ISSN 2579-2784 (Print)
ISSN 2738-2788 (Online)

CONTENTS

Mathematical Problems of Computer Science 59, 7–15, 2023.
doi:10.51408/1963-0097

UDC 519.1

A Note on Large Cycles in Graphs Around Conjectures of Bondy and Jung

Zhora G. Nikoghosyan

Institute for Informatics and Automation Problems of NAS RA, Yerevan, Armenia
e-mail: zhora@iiap.sci.am

Abstract

New sufficient conditions are derived for generalized cycles (including Hamilton and dominating cycles as special cases) in an arbitrary k-connected ($k = 1, 2, ...$) graph, which prove the truth of Bondy's (1980) famous conjecture for some variants significantly improving the result expected by the given hypothesis. Similarly, new lower bounds for the circumference (the length of a longest cycle) are established for the reverse hypothesis proposed by Jung (2001) combined inspiring new improved versions of the original conjectures of Bondy and Jung.
Keywords: Hamilton cycle, Dominating cycle, Longest cycle, Large cycle.
Article info: Received 27 January 2021; sent for review 14 February 2022; received in revised form 11 January 2023; accepted 7 March 2023.

1. Introduction

We consider only finite undirected graphs without loops or multiple edges. The set of vertices of a graph G is denoted by $V(G)$; the set of edges by $E(G)$. For a subset S of $V(G)$, we denote by $G-S$ the maximum subgraph of G with the vertex set $V(G)-S$. For a subgraph H of G, we use $G-H$, short for $G-V(H)$. A good reference for any undefined terms is [3].

Let α and δ be the independence number and the minimum degree of a graph G, respectively. We define σ_k by the minimum degree sum of any k independent vertices if $\alpha \geq k$; if $\alpha < k$, we set $\sigma_k = +\infty$. In particular, we have $\sigma_1 = \delta$.

A simple cycle (or just a cycle) Q of order t (the number of vertices) is a sequence $v_1v_2...v_tv_1$ of distinct vertices $v_1, ..., v_t$ with $v_iv_{i+1} \in E(G)$ for each $i \in \{1, ..., t\}$, where $v_{t+1} = v_1$. When $t = 1$, the cycle v_1 coincides with the vertex v_1. So, by this standard definition, all vertices and edges in a graph can be considered as cycles of orders 1 and 2, respectively. Such an extension of the cycle definition allows to avoid unnecessary repetition "let G be a graph of order $n \geq 3$" in a large number of results. Further, a simple path (or just a path) of order t is a sequence $v_1v_2...v_t$ of distinct vertices $v_1, ..., v_t$ with $v_iv_{i+1} \in E(G)$ for each $i \in \{1, ..., t-1\}$.

A graph G is Hamiltonian if G contains a Hamilton cycle, i.e., a cycle of order $|V(G)|$.

Now let Q be an arbitrary cycle in G. We say that Q is a dominating cycle in G if $V(G-Q)$ is an independent set of vertices.

The first type of generalized cycles, including Hamilton and dominating cycles as special cases, was introduced by Bondy [4]. For a positive integer λ, Q is said to be a D_λ-cycle if $|H| \leq \lambda - 1$ for every component H of $G - Q$. Alternatively, Q is a D_λ-cycle of G if and only if every connected subgraph of order λ of G has at least one vertex with Q in common. Thus, a D_λ-cycle dominates all connected subgraphs of order λ. By this definition, Q is a Hamilton cycle if and only if Q is a D_1-cycle. Analogously, Q is a dominating cycle if and only if Q is a D_2-cycle.

We now present another two types of more interesting generalized cycles that form the main topic of this paper. For a positive integer λ, the cycle Q is called a PD_λ-cycle (PD - Path Dominating) if each path of order at least λ in G has at least one vertex with Q in common. Similarly, we call the cycle Q a CD_λ-cycle (CD - Cycle Dominating; introduced in [13]) if each cycle of order at least λ has at least one vertex with Q in common. In fact, a PD_λ-cycle dominates all paths of order λ in G; and a CD_λ-cycle dominates all cycles of order λ in G. In terms of PD_λ and CD_λ-cycles, Q is a Hamilton cycle if and only if either Q is a PD_1-cycle or a CD_1-cycle. Further, Q is a dominating cycle if and only if either Q is a PD_2-cycle or a CD_2-cycle.

Throughout the paper, we consider a graph G on n vertices with minimum degree δ and connectivity κ. Further, let C be a longest cycle in G with $c = |C|$, and let $\overline{p}$ and $\overline{c}$ denote the orders of a longest path and a longest cycle in $G - C$, respectively. In particular, C is a Hamilton cycle if and only if $\overline{p} \leq 0$ or $\overline{c} \leq 0$. Similarly, C is a dominating cycle if and only if $\overline{p} \leq 1$ or $\overline{c} \leq 1$.

In 1980, Bondy [4] conjectured a common generalization of some well-known degree-sum conditions for PD_λ-cycles (called $(\sigma, \overline{p})$-version) including Hamilton cycles (PD_1-cycles) and dominating cycles (PD_2-cycles) as special cases.

Conjecture 1. *(Bondy [4],1980): $(\sigma, \overline{p})$-version.*
Let C be a longest cycle in a λ-connected $(1 \leq \lambda \leq \delta)$ graph G of order n. If $\sigma_{\lambda+1} \geq n + \lambda(\lambda-1)$, then $\overline{p} \leq \lambda - 1$.

Parts of Conjecture 1 were proved for $\lambda = 1, 2, 3$.

(a) $\kappa \geq 1,\ \sigma_2 \geq n \implies \overline{p} \leq 0$ $(Ore[15], 1960)$,
(b) $\kappa \geq 2,\ \sigma_3 \geq n+2 \implies \overline{p} \leq 1$ $(Bondy[4], 1980)$,
(c) $\kappa \geq 3,\ \sigma_4 \geq n+6 \implies \overline{p} \leq 2$ $(Zou[17], 1987)$.

For the general case, Conjecture 1 is still open.

The long cycles analogue (the so called reverse version) of Bondy's conjecture (Conjecture 1) can be formulated as follows.

Conjecture 2. *(reverse, σ, $\overline{p}$)-version.*
Let C be a longest cycle in a λ-connected $(1 \leq \lambda \leq \delta)$ graph G. If $\overline{p} \geq \lambda - 1$, then $c \geq \sigma_\lambda - \lambda(\lambda-2)$.

Parts of Conjecture 2 were proved for $\lambda = 1, 2, 3, 4$.

(d) $\kappa \geq 1,\ \overline{p} \geq 0 \implies c \geq \sigma_1 + 1$ $(Dirac[6], 1952)$,

$(e)\ \ \kappa \geq 2,\ \overline{p} \geq 1 \implies c \geq \sigma_2$ $(Bondy[2], 1971; Bermond[1], 1976; Linial[11], 1976)$,
$(f)\ \ \kappa \geq 3,\ \overline{p} \geq 2 \implies c \geq \sigma_3 - 3$ $(Fraisse, Jung[8], 1989)$,
$(g)\ \ \kappa \geq 4,\ \overline{p} \geq 3 \implies c \geq \sigma_4 - 8$ $(Chiba, Tsugaki, Yamashita[5], 2014)$.

Note that the initial motivations of Conjecture 1 and Conjecture 2 come from their minimal degree versions - the most popular and much studied versions, which also remain unsolved.

Conjecture 3. *(Bondy [4],1980): $(\delta, \overline{p})$-version.*
Let C be a longest cycle in a λ-connected $(1 \leq \lambda \leq \delta)$ graph G of order n. If $\delta \geq \frac{n+2}{\lambda+1} + \lambda - 2$, then $\overline{p} \leq \lambda - 1$.

Conjecture 4. *(Jung [10], 2001): (reverse, δ, $\overline{p}$)-version.*
Let C be a longest cycle in a λ-connected $(1 \leq \lambda \leq \delta)$ graph G. If $\overline{p} \geq \lambda - 1$, then $c \geq \lambda(\delta - \lambda + 2)$.

Parts of Conjecture 3 were proved for $\lambda = 1, 2, 3$.

$(h)\ \ \kappa \geq 1,\ \ \delta \geq \frac{n}{2} \implies \overline{p} \leq 0$ $(Dirac[6], 1952)$,
$(i)\ \ \kappa \geq 2,\ \ \delta \geq \frac{n+2}{3} \implies \overline{p} \leq 1$ $(Nash - Williams[12], 1971)$,
$(j)\ \ \kappa \geq 3,\ \ \delta \geq \frac{n+6}{4} \implies \overline{p} \leq 2$ $(Fan[7], 1987)$.

Parts of Conjecture 4 were proved for $\lambda = 1, 2, 3, 4$.

$(k)\ \ \kappa \geq 1,\ \ \overline{p} \geq 0 \implies c \geq \delta + 1$ $(Dirac[6], 1952)$,
$(l)\ \ \kappa \geq 2,\ \ \overline{p} \geq 1 \implies c \geq 2\delta$ $(Dirac[6], 1952)$,
$(m)\ \kappa \geq 3,\ \ \overline{p} \geq 2 \implies c \geq 3\delta - 3$ $(Voss, Zuluaga[16], 1977)$,
$(n)\ \ \kappa \geq 4,\ \ \overline{p} \geq 3 \implies c \geq 4\delta - 8$ $(Jung[9], 1990)$.

Note that CD_λ-cycles are more suitable for research than PD_λ-cycles since cycles in $G - C$ are more symmetrical than paths in view of the connections between $G - C$ and CD_λ-cycles. This is the main reason why some minimum degree versions of Conjectures 1 and 2 have been solved just for CD_λ-cycles.

According to the above arguments, it is natural to consider the exact analogues of Bondy's generalized conjecture (Conjecture 1) and its reverse version (Conjecture 2) for CD_λ-cycles, which we call $(\sigma, \overline{c})$ and $(reverse, \sigma, \overline{c})$-versions, respectively.

Conjecture 5. *$(\sigma, \overline{c})$-version.*
Let C be a longest cycle in a λ-connected $(1 \leq \lambda \leq \delta)$ graph G of order n. If $\sigma_{\lambda+1} \geq n + \lambda(\lambda - 1)$, then $\overline{c} \leq \lambda - 1$.

Conjecture 6. *(reverse, σ, $\overline{c}$)-version.*
Let C be a longest cycle in a λ-connected $(1 \leq \lambda \leq \delta)$ graph. If $\overline{c} \geq \lambda - 1$, then $c \geq \sigma_\lambda - \lambda(\lambda - 2)$.

In 2009, the author proved [14] the validity of minimum degree versions of Conjectures 5 and 6.

Theorem 1. *([14], 2009): $(\delta, \overline{c})$-version.*
Let C be a longest cycle in a λ-connected $(1 \leq \lambda \leq \delta$ graph G of order n. If $\delta \geq \frac{n+2}{\lambda+1} + \lambda - 2$, then $\overline{c} \leq \lambda - 1$.

Theorem 2. *([14], 2009): (reverse, δ, $\overline{c}$)-version.*
Let C be a longest cycle in a λ-connected $(1 \leq \lambda \leq \delta)$ graph. If $\overline{c} \geq \lambda - 1$, then $c \geq \lambda(\delta - \lambda + 2)$.

Actually, in [14], a significantly stronger result than Theorem 1 was proved showing that the conclusion $\overline{c} \leq \lambda - 1$ in Theorem 1 can be strengthened to $\overline{c} \leq \min\{\lambda - 1, \delta - \lambda\}$, called $\overline{c}$-improvement.

Theorem 3. *([14], 2009): $(\delta, \overline{c})$-version, $\overline{c}$-improvement.*
Let C be a longest cycle in a λ-connected $(1 \leq \lambda \leq \delta)$ graph G of order n. If $\delta \geq \frac{n+2}{\lambda+1} + \lambda - 2$, then $\overline{c} \leq \min\{\lambda - 1, \delta - \lambda\}$.

Analogously, the condition $\overline{c} \geq \lambda - 1$ in Theorem 2 was weakened [14] to $\overline{c} \geq \min\{\lambda - 1, \delta - \lambda + 1\}$.

Theorem 4. *([14], 2009): (reverse, δ, $\overline{c}$)-version, $\overline{c}$-improvement.*
Let C be a longest cycle in a λ-connected $(1 \leq \lambda \leq \delta)$ graph G. If $\overline{c} \geq \min\{\lambda - 1, \delta - \lambda + 1\}$, then $c \geq \lambda(\delta - \lambda + 2)$.

In this paper, we present new analogous further improvements of Theorems 1, 2, 3, 4 inspiring new conjectures in forms of improvements of the initial generalized conjectures of Bondy and Jung.

2. Results

First, we prove that the connectivity condition $\kappa \geq \lambda$ in Theorem 1 can be weakened to $\kappa \geq \min\{\lambda, \delta - \lambda + 1\}$.

Theorem 5. *$(\delta, \overline{c})$-version, κ-improvement.*
Let C be a longest cycle in a graph G of order n and λ a positive integer with $1 \leq \lambda \leq \delta$. If $\kappa \geq \min\{\lambda, \delta - \lambda + 1\}$ and $\delta \geq \frac{n+2}{\lambda+1} + \lambda - 2$, then $\overline{c} \leq \lambda - 1$.

Analogously, we prove that the connectivity condition $\kappa \geq \lambda$ in Theorem 2 can be weakened to $\kappa \geq \min\{\lambda, \delta - \lambda + 2\}$.

Theorem 6. *(reverse, δ, $\overline{c}$)-version, κ-improvement.*
Let C be a longest cycle in a graph G and λ a positive integer with $1 \leq \lambda \leq \delta$. If $\kappa \geq \min\{\lambda, \delta - \lambda + 2\}$ and $\overline{c} \geq \lambda - 1$, then $c \geq \lambda(\delta - \lambda + 2)$.

Next, we prove that the conclusion $\overline{c} \leq \lambda - 1$ in Theorem 5 can be strengthened to $\overline{c} \leq \min\{\lambda - 1, \delta - \lambda\}$.

Theorem 7. *$(\delta, \overline{c})$-version, $(\overline{c}, \kappa)$-improvement.*
Let C be a longest cycle in a graph G of order n and λ a positive integer with $1 \leq \lambda \leq \delta$. If $\kappa \geq \min\{\lambda, \delta - \lambda + 1\}$ and $\delta \geq \frac{n+2}{\lambda+1} + \lambda - 2$, then $\overline{c} \leq \min\{\lambda - 1, \delta - \lambda\}$.

Finally, we prove that the condition $\overline{c} \geq \lambda - 1$ in Theorem 6 can be weakened to $\overline{c} \geq \min\{\lambda - 1, \delta - \lambda + 1\}$.

Theorem 8. *(reverse, δ, $\overline{c}$)-version, $(\overline{c}, \kappa)$-improvement.*
Let C be a longest cycle in a graph G and λ a positive integer with $1 \leq \lambda \leq \delta$. If $\kappa \geq \min\{\lambda, \delta - \lambda + 2\}$ and $\overline{c} \geq \min\{\lambda - 1, \delta - \lambda + 1\}$, then $c \geq \lambda(\delta - \lambda + 2)$.

3. Generalized Improvements of Conjectures of Bondy and Jung

Motivated by Theorems 5, 6, 7, 8 (minimum degree versions) with Conjectures 1 and 2, in this section we propose their exact analogs in terms of degree sums as generalized improvements of Bondy and Jung Conjectures.

Conjecture 7. *$(\sigma, \overline{c})$-version, $(\overline{c}, \kappa)$-improvement.*
Let C be a longest cycle in a graph G of order n and λ a positive integer. If $\kappa \geq \min\{\lambda, \delta - \lambda + 1\}$ and $\sigma_{\lambda+1} \geq n + \lambda(\lambda - 1)$, then $\overline{c} \leq \min\{\lambda - 1, \delta - \lambda\}$.

Conjecture 8. *(reverse, σ, $\overline{c}$)-version, $(\overline{c}, \kappa)$-improvement.*
Let C be a longest cycle in a graph G and λ a positive integer. If $\kappa \geq \min\{\lambda, \delta - \lambda + 2\}$ and $\overline{c} \geq \min\{\lambda - 1, \delta - \lambda + 1\}$, then $c \geq \sigma_\lambda - \lambda(\lambda - 2)$.

Conjecture 9. *$(\sigma, \overline{p})$-version, $(\overline{p}, \kappa)$-improvement.*
Let C be a longest cycle in a graph G of order n and λ a positive integer. If $\kappa \geq \min\{\lambda, \delta - \lambda + 1\}$ and $\sigma_{\lambda+1} \geq n + \lambda(\lambda - 1)$, then $\overline{p} \leq \min\{\lambda - 1, \delta - \lambda\}$.

Conjecture 10. *(reverse, σ, $\overline{p}$)-version, $(\overline{p}, \kappa)$-improvement.*
Let C be a longest cycle in a graph G and λ a positive integer. If $\kappa \geq \min\{\lambda, \delta - \lambda + 2\}$ and $\overline{p} \geq \min\{\lambda - 1, \delta - \lambda + 1\}$, then $c \geq \sigma_\lambda - \lambda(\lambda - 2)$.

4. Proofs

Proof of Theorem 7. We shall prove that $\overline{c} \leq \min\{\lambda - 1, \delta - \lambda\}$ under the conditions

$$\kappa \geq \min\{\lambda, \delta - \lambda + 1\}, \quad \delta \geq \frac{n+2}{\lambda+1} + \lambda - 2$$

for each $1 \leq \lambda \leq \delta$. If $\min\{\lambda, \delta - \lambda + 1\} = \lambda$, that is $\lambda \leq \lfloor \frac{\delta+1}{2} \rfloor$, then we shall prove that $\overline{c} \leq \lambda - 1$ under the conditions

$$\kappa \geq \lambda, \quad \delta \geq \frac{n+2}{\lambda+1} + \lambda - 2.$$

But the latter follows from Theorem 1 for all $\lambda = 1, 2, ..., \lfloor \frac{\delta+1}{2} \rfloor$ immediately.

Now let $\min\{\lambda, \delta - \lambda + 1\} = \delta - \lambda + 1$, that is $\lambda \geq \lfloor \frac{\delta+2}{2} \rfloor$. To conclude the proof, it remains to show that

$$\kappa \geq \delta - \lambda + 1, \ \delta \geq \frac{n+2}{\lambda+1} + \lambda - 2 \quad \Rightarrow \quad \overline{c} \leq \delta - \lambda \quad \left(\lambda = \delta, \delta - 1, ..., \left\lfloor \frac{\delta+2}{2} \right\rfloor\right). \qquad (1)$$

Put $\delta - \lambda + 1 = \mu$. Acording to this notation, (1) is equivalent to

$$\kappa \geq \mu, \ \delta \geq \frac{n+2}{\delta - \mu + 2} + \delta - \mu - 1 \quad \Rightarrow \quad \overline{c} \leq \mu - 1 \quad \left(\mu = 1, 2, ..., \left\lfloor \frac{\delta+1}{2} \right\rfloor\right). \qquad (2)$$

In (2), the inequality

$$\delta \geq \frac{n+2}{\delta - \mu + 2} + \delta - \mu - 1$$

is equivalent to

$$\delta \geq \frac{n+2}{\mu+1} + \mu - 2,$$

implying that (2) is equivalent to

$$\kappa \geq \mu,\ \delta \geq \frac{n+2}{\mu+1} + \mu - 2 \quad \Rightarrow \quad \overline{c} \leq \mu - 1 \quad \left(\mu = 1, 2, ..., \left\lfloor \frac{\delta+1}{2} \right\rfloor\right). \tag{3}$$

Observing that (3) follows from Theorem 1 immediately, we obtain

$$(1) \equiv (2) \equiv (3) \ \Leftarrow \ \text{"Theorem 1"}.$$

Theorem 7 is proved. ■

Proof of Theorem 5. Let G be a graph with

$$\kappa \geq \min\{\lambda, \delta - \lambda + 1\}, \quad \delta \geq \frac{n+2}{\lambda+1} + \lambda - 2$$

for each $1 \leq \lambda \leq \delta$. We shall prove that $\overline{c} \leq \lambda - 1$. Observing that $\min\{\lambda - 1, \delta - \lambda\} \leq \lambda - 1$, we can weaken the conclusion $\overline{c} \leq \min\{\lambda - 1, \delta - \lambda\}$ in Theorem 7 to $\overline{c} \leq \lambda - 1$ and the result follows immediatly. ■

Proof of Theorem 8. Let G be a graph with

$$\kappa \geq \min\{\lambda, \delta - \lambda + 2\}, \quad \overline{c} \geq \min\{\lambda - 1, \delta - \lambda + 1\}$$

for each $1 \leq \lambda \leq \delta$. We shall prove that $c \geq \lambda(\delta - \lambda + 2)$. If $\lambda = 1$, then the result follows from the fact that each graph has a cycle of length at least $\delta + 1$ [6]. Let $\lambda \geq 2$. Further, if $\min\{\lambda, \delta - \lambda + 2\} = \lambda$, then we are done by Theorem 2. Now let $\min\{\lambda, \delta - \lambda + 2\} = \delta - \lambda + 2$, that is $\lambda \geq \lfloor \frac{\delta+3}{2} \rfloor$. Then it remains to prove that

$$\kappa \geq \delta - \lambda + 2, \ \ \overline{c} \geq \delta - \lambda + 1 \ \Rightarrow \ c \geq \lambda(\delta - \lambda + 2) \quad \left(\lambda = \delta, \delta - 1, ..., \left\lfloor \frac{\delta+3}{2} \right\rfloor\right). \tag{4}$$

Put $\delta - \lambda + 2 = \mu$. By this notation, the statement (4) is equivalent to

$$\kappa \geq \mu \ \ \overline{c} \geq \mu - 1 \ \Rightarrow \ c \geq \mu(\delta - \mu + 2) \quad \left(\mu = 2, 3, ..., \left\lfloor \frac{\delta+2}{2} \right\rfloor\right), \tag{5}$$

which follows from Theorem 2 immediately. So, (4) $\equiv$ (5) $\Leftarrow$ "Theorem 2". Theorem 8 is proved. ■

Proof of Theorem 6. Let G be a graph with

$$\kappa \geq \min\{\lambda, \delta - \lambda + 2\}, \quad \overline{c} \geq \lambda - 1$$

for each $1 \leq \lambda \leq \delta$. We shall prove that $c \geq \lambda(\delta - \lambda + 2)$. Observing that $\min\{\lambda - 1, \delta - \lambda + 1\} \leq \lambda - 1$, we can strengthen the condition $\overline{c} \geq \min\{\lambda - 1, \delta - \lambda + 1\}$ in Theorem 8 to $\overline{c} \geq \lambda - 1$ and the result follows immediately. Theorem 6 is proved. ■.

5. Conclusion

In 2009 [14], a minimum degree sufficient condition for large cycles in graphs is established showing that the famous conjecture of Bondy principally is improvable. In the same paper, a lower bound for the length of a longest cycle (the circumference) is derived showing that the conjecture of Jung (reverse version of Bondys conjecture) principally is improvable as well. In this note, two new analogous sufficient conditions for large cycles and two new lower bounds for the circumference are derived inspiring four new improved versions of Bondys and Jungs conjectures.

References

[1] J.C. Bermond, "On Hamiltonian walks", *Congressus Numerantium*, vol.15, pp. 41-50, 1976.

[2] J.A. Bondy, "Large cycles in graphs", *Discrete Mathematics*, vol. 1, pp. 121-131, 1971.

[3] J.A. Bondy and U.S.R. Murty, *Graph Theory with Applications*, Macmillan, London and Elsevier, New York, 1976.

[4] J.A. Bondy, *Longest paths and cycles in graphs of high degree*, Research Report CORR 80-16, Department of Combinatorics and Optimization, Faculty of Mathematics, University of Waterloo, Ontario, Canada, 14 pages, 1980.

[5] S. Chiba, M. Tsugaki and T. Yamashita, "Degree sum conditions for the circumference of 4-connected graphs", *Discrete Math.*, vol. 333, pp. 66-83, 2014.

[6] G.A. Dirac, "Some theorems on abstract graphs", *Proc. London Math. Soc.*, vol. 2, pp. 69-81, 1952.

[7] G. Fan, *Extremal theorems on paths and cycles in graphs and weighted graphs*, PhD thesis, University of Waterloo, 1987.

[8] P. Fraisse and H.A. Jung, "Longest cycles and independent sets in k-connected graphs", *Recent Studies in Graph Theory*, Vischwa Internat. Publ., Gulbarga, India, pp. 114-139, 1989.

[9] H.A. Jung and H.A. Jung, "Longest cycles in graphs with moderate connectivity", *Topics in Combinatorics and Graph Theory, Essays in Honour of Gerhard Ringel*, Physica-Verlag, Heidelberg, pp. 765778, 1990.

[10] H.A. Jung, "Degree bounds for long paths and cycles in k-connected graphs", *Computational Discrete Mathematics, Lecture Notes in Comput. Sci., Springer*, Berlin, vol. 2122, pp. 56-60, 2001.

[11] N. Linial, "A lower bound on the circumference of a graph", *Discrete Math.*, vol. 15, pp. 297-300, 1976.

[12] C.St.J.A. Nash-Williams, "Edge-disjoint hamiltonian cycles in graphs with vertices of large valency", *Studies in Pure Mathematics*, Academic Press, San Diego, London, pp. 157-183, 1971.

[13] Zh. G. Nikoghosyan, "Cycle-Extensions and long cycles in graphs", *Transactions of the Institute for Informatics and Automation Problems (IIAP) of NAS of RA, Mathematical Problems of Computer Science*, vol. 21, pp. 121-128, 2000.

[14] Zh.G. Nikoghosyan, "Dirac-type generalizations concerning large cycles in graphs", *Discrete Mathematics*, vol. 309, pp. 1925-1930, 2009.

[15] O. Ore, "A note on Hamiltonian circuits", *Amer. Math. Monthly*, vol. 67, p. 55, 1960.

[16] H.-J. Voss and C. Zuluaga, "Maximale gerade und ungerade Kreise in Graphen", I, Wiss. Z. Techn. Hochschule Ilmenau, vol. 4, pp. 57-70, 1977.

[17] Y. Zou, "A generalization of a theorem of Jung", *J. Nanjing Normal Univ. Nat. Sci.*, vol. 2, pp. 8-11, 1987.

Ակնարկ գրաֆներում մեծ ցիկլերի մասին Բոնդիի և Յունգի վարկածների շուրջ

Ժորա Գ. Նիկողոսյան

ՀՀ ԳԱԱ Ինֆորմատիկայի և ավտոմատացման պրոբլեմների ինստիտուտ, Երևան, Հայաստան
e-mail: zhora@iiap.sci.am

Ամփոփում

Ստացվել են նոր բավարար պայմաններ գրաֆի ընդհանրացված ցիկլերի համար (ընդգրկելով Համիլթոնյան և դոմինանտ ցիկլերը որպես մասնավոր դեպքեր) կամայական k-կապակցված $(k = 1, 2, ...)$ գրաֆում, որոնք ապացուցում են Բոնդիի (1980) հայտնի վարկածի ճշմարտացիությունը որոշ տարբերակների դեպքում, ինչի շնորհիվ զգալիորեն լավացվում է տվյալ վարկածով ակնկալվող արդյունքը: Համանմանորեն, ամենաերկար ցիկլի երկարության համար ստացվել են նոր ստորին գնահատականներ հակադարձ վարկածի համար, որն առաջ է քաշել Յունգը 2001-ին: Ստացված արդյունքները բավարար հիմքեր են տալիս առաջ քաշելու նոր լավացված տարբերակներ Բոնդիի և Յունգի նախնական վարկածների փոխարեն:

Բանալի բառեր՝ Համիլթոնի ցիկլ, դոմինանտ ցիկլ, ամենաերկար ցիկլ, մեծ ցիկլ:

Заметка о больших циклах в графах вокруг гипотез Бонди и Юнга

Жора Г. Никогосян

Институт проблем информатики и автоматизации НАН РА, Ереван, Армения
e-mail: zhora@iiap.sci.am

Аннотация

Получены новые достаточные условия для обобщенных циклов (включая гамильтоновые и доминантные циклы как частные случаи) в произвольном k-

связном графе ($k = 1, 2, ...$), доказывающие справедливость известной гипотезы Бонди (1980) для некоторых вариантов, значительно улучшив ожидаемый по данной гипотезе результат. Аналогично, получены новые нижние оценки для длины длиннейшего цикла графа для обратной гипотезы, предложенной Юнгом (2001). Полученные результаты в сочетании дают основания выдвижения новых улучшенных вариантов для исходных гипотез Бонди и Юнга.

Ключевые слова: Гамильтонов цикл, доминантный цикл, длиннейший цикл, большой цикл.

Mathematical Problems of Computer Science 59, 16–26, 2023.
doi: 10.51408/1963-0098

UDC 519.714

RDNF Oriented Analytics to Random Boolean Functions

Levon H. Aslanyan, Irina A. Arsenyan, Vilik M. Karakhanyan and Hasmik A. Sahakyan

Institute for Informatics and Automation Problems of NAS RA,Yerevan, Armenia
e-mail: kavilik@gmail.com

Abstract

Dominant areas of computer science and computation systems are intensively linked to the hypercube-related studies and interpretations. This article presents some transformations and analytics for some example algorithms and Boolean domain problems. Our focus is on the methodology of complexity evaluation and integration of several types of postulations concerning special hypercube structures. Our primary goal is to demonstrate the usual formulas and analytics in this area, giving the necessary set of common formulas often used for complexity estimations and approximations. The basic example under considered is the Boolean minimization problem, in terms of the average complexity of the so-called reduced disjunctive normal form (also referred to as complete, prime irredundant, or Blake canonical form). In fact, combinatorial counterparts of the disjunctive normal form complexities are investigated in terms of sets of their maximal intervals. The results obtained compose the basis of logical separation classification algorithmic technology of pattern recognition. In fact, these considerations are not only general tools of minimization investigations of Boolean functions, but they also prove useful structures, models, and analytics for constraint logic programming, machine learning, decision policy optimization and other domains of computer science.
Keywords: Boolean function, Hypercube, Complexity, Asymptotic, Reduced disjunctive normal form.
Article info: Received 14 February 2023; sent for review 10 March 2023; accepted 11 April 2023.

1. Hypercube and Related Structures

The metric theory of Boolean functions (BF) [1], [2] arose in the 70's, in parallel with the emergence of broader design and implementation ideas for mechanical and electronic computation devices. It was then that it turned out that the system of binary representation of numbers is the most optimal, both from the point of view of the algorithmic implementation of arithmetic calculations and also from the point of view of developing physical carriers of performing these calculations [3]. BF – functions with only binary variables, and also

with values in the domain $\{0,1\}$, although simple among the other similar mathematical concepts, they are quite complex in solving problems associated with their transformations and optimization. The metric theory of Boolean functions provides the necessary knowledge for coding, transforming and implementing binary functions. Although the way to minimal BF representations are and remains difficult, a rather complete picture of the main forms of function representation of functions has been obtained, and the basic role here takes the concept of disjunctive normal forms. Successive steps of several transformations of functions are found to achieve minimal forms as a chain from the table or formula representation to the reduced d.n.f., then to the deadlock forms and finally – the minimal structures. The accompanying structures and bottlenecks of achieving acceptable optimization are investigated intensively [1], [4]–[7]. Here we will not cover the whole theory but will pay attention to one fundamental construction, – to the concept of reduced disjunctive normal forms (r.d.n.f.) of Boolean functions. R.d.n.f. is the collection of all minimal conjunctions and geometrically - the system of all maximum intervals/sub-cubes of functions. These forms are a universal concept, and they also arise in problems such as circuit design from set of functional elements (logical part of chip design), in the theory of pattern recognition (logic separation algorithm, and generation of logical regularities) [8]–[11], in biological models of heredity and mutations (phylogeny, parsimony) [12, 13], etc. Turning to the complexity characterization of structures associated with the reduced disjunctive normal form, where two types are usually considered: the largest and most typical characteristics, we will focus on the second component. In a concise survey of the domain, the initial studies of [5], [14], and [15], should be mentioned, that give the formulas of average numbers of maximal intervals in Boolean functions. [16], [17] extended these results to the case of partially defined Boolean functions. An alternative track of papers in these topics includes the articles [18], [19], [20]. Current research on the topics of BF and complexities might be demonstrated through the papers [21]–[26]. Methodologically, in studies in the area of BF, it should be taken into account that the function determination domain, as well as the number of functions itself, are finite, depending on the number of the variables – the dimensionality. So, considering the parameter $\pi(f)$ over the functions, we get the split of these functions into finite classes by the values of this parameter. These are the rates and intensity of the accepted values of the parameter $\pi(f)$. In some cases, it is convenient to refer to these valuations as probabilistic distributions, which is not obligatorily but is convenient in some contexts. In this concern, there appears a link to the model of Random Boolean functions and the combinatorial theories initiated by A. Renyi and P. Erdos [27], [28].

1.1 Concepts and Definitions in the Binary Domain

Elementary conjunction, Direction. Let $\tilde{\alpha}$ and $\tilde{\beta}$ – be arbitrary vertices of the n-dimensional unite cube. And let $j_i, i = 1, 2, \cdots, r$ be all coordinates, those where $\alpha_{j_i} = \beta_{j_i}$. Consider the formula

$$\mathcal{K}(x_1, x_2, \cdots, x_n) = \bigwedge_{i=1}^{r} x_{j_i}^{\sigma_{j_i}},$$

with $\sigma_{j_i} = \alpha_{j_i}$, $i = 1, 2, \cdots, r$. We say that K is an elementary conjunction stretched on the pair of vertices $\tilde{\alpha}$ and $\tilde{\beta}$ of the n-dimensional unit cube E_n. The number of literals in $\mathcal{K}$ is the rank of $\mathcal{K}$. The geometrical counterpart of $\mathcal{K}$ is a sub-cube defined as follows. Assign 0 values to all but $j_1, j_2, \cdots, j_r$ coordinates and denote this vertex by v_0. Similarly, assign

these coordinates by the value 1, obtaining the vertex v_1. These are the minimal and maximal vertices that belong to $\mathcal{K}$, and they determine a unique sub-cube of all truth vertices of $\mathcal{K}$. $n-r$, the number of variable coordinates of $\mathcal{K}$ is the size of its sub-cube.

Let $\lambda = \{j_1, j_2, \cdots, j_r\}$ be a collection of r indices drawn up of variables $x_1, x_2, \cdots, x_n$, and let $\bar{\lambda}$ be the complementary to the λ set of indices. Conjunctions of the form $\bigwedge_{i=1}^{r} x_{j_i}^{\sigma_{j_i}}$ and the corresponding intervals will be called conjunctions and intervals of the direction λ. For a fixed r there are C_n^r different directions, and each of them is determined by the appropriate selection of an r subset $\{j_1, j_2, \cdots, j_r\}$ of the set $\{1, 2, ..., n\}$. The individual interval in the direction $\{j_1, j_2, \cdots, j_r\}$ appears in result of assigning the values $\sigma_1, \sigma_2, \cdots, \sigma_r$ to the variables $x_{j_1}, x_{j_2}, \cdots, x_{j_r}$.

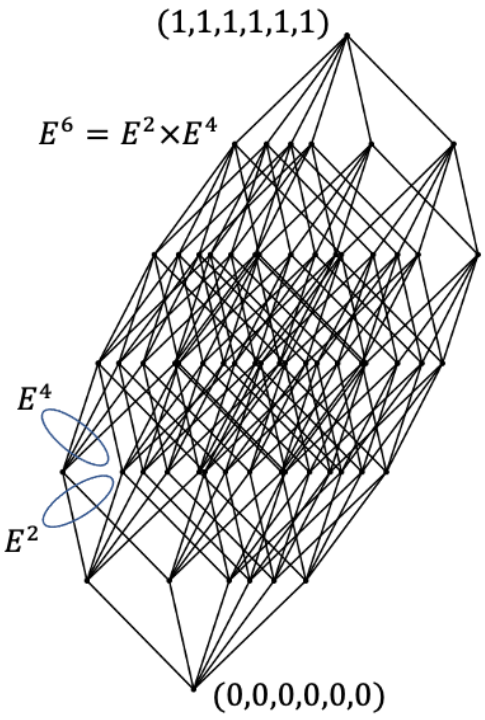

Fig. 1. Geometry of hypercube.

This also means that there are 2^{n-r} conjunctions and intervals in one of the r-directions. The collection $\bar{\lambda}$ of indices defines another set of directions.

Let $\mathcal{F}$ be an arbitrary logical formula and $\mathcal{M} \subseteq B^n$. We say that $\mathcal{F}$ absorbs or covers $\mathcal{M}$ if on each tuple $\tilde{\alpha} \in \mathcal{M}$ the formula $\mathcal{F}$ accepts the unite (true) value.

Let $\tilde{\alpha} \in E^n$ be an arbitrary vertex. Call the value $\mid \tilde{\alpha} \mid = \sum_{i=1}^{n} \alpha_i$ the module or the weight of $\tilde{\alpha}$. The set of all vertices $\tilde{\beta} \in E^n$, with $\rho(\tilde{\alpha}, \tilde{\beta}) = \mid \tilde{\alpha} \oplus \tilde{\beta} \mid = k$, call the k–the layer of E^n in relation to the vertex $\tilde{\alpha}$ ($\oplus$ – mentions $mod2$ summation).

Intervals $\mathcal{N}_{K^1}$ and $\mathcal{N}_{K^2}$,

$$K^1(x_1, x_2, \cdots, x_n) = \bigwedge_{i=1}^{r} x_{j_i}^{\sigma_{j_i}^1} \text{ and } K^2(x_1, x_2, \cdots, x_n) = \bigwedge_{i=1}^{r} x_{j_i}^{\sigma_{j_i}^2}$$

of the same size and the same direction we call neighbors if $\rho(\tilde{\sigma}^1, \tilde{\sigma}^2) = 1$, where ρ – be the Hamming distance, $\rho(\tilde{\sigma}^1, \tilde{\sigma}^2) = \sum_{i=1}^{r} \mid \sigma_{j_i}^1 - \sigma_{j_i}^2 \mid$. Let then j_{i_0} is the number of that unique coordinate for which $\sigma_{j_{i_0}}^1 \neq \sigma_{j_{i_0}}^2$. Then we say that the conjunctions K^1 and K^2 (or the pair of neighbor intervals corresponding to them) joined by the coordinate $x_{j_{i_0}}$, and, as a result,

a new conjunction (interval) appears:

$$\bigwedge_{i\neq i_0, i=1}^{r} x_{j_i}^{\sigma_{j_i}}.$$

Partition the variable set $x_1, x_2, \cdots, x_n$ in an arbitrary manner into two nonempty groups: $x_{i_1}, x_{i_2}, \cdots, x_k$ as the first group, and $x_{i_{k+1}}, x_{i_{k+2}}, \cdots, x_{i_n}$ as the second. Then, the n-dimensional unit cube E_n may be represented as the Cartesian multiplication $B^k \times B^{n-k}$ of two sub-cubes: B^k and B^{n-k} generated correspondingly by the sets of variables $x_{i_1}, x_{i_2}, \cdots, x_{i_k}$ and $x_{i_{k+1}}, x_{i_{k+2}}, \cdots, x_{i_n}$. Let us enumerate the vertices of B^{n-k} by the layers relative to the vertex $\tilde{0}$ of B^{n-k}. Enumeration among the vertices of a particular layer is arbitrary, but the first group that is enumerated by low numbers is layer zero, then the first layer, and so on. Additional ordering among layer vertices may use lexicographic order, binary value based order, etc.

Consider an arbitrary k-dimensional sub-cube B^k of E^n, the first k-dimensional interval B_1^k in the direction of B^k. List the neighbor intervals to the considered one, B_1^k, - $B_2^k, B_3^k, \cdots, B_{n-k+1}^k$. Let f be an arbitrary (partially defined) Boolean function that satisfies the following conditions:

$\alpha)$ B_1^k doesn't contain zero value vertices of f : $(\forall \tilde{\alpha} \in B_1^k,\ f(\tilde{\alpha}) \neq 0)$,

$\beta)$ Each of the neighbor with B_1^k interval contains at least one 'unit' value vertex f : $(\forall j,\ j = 2, 3, \cdots, n-k+1\ \exists \tilde{\alpha} \in B_j^k,\ f(\tilde{\alpha}) = 1)$,

$\gamma)$ B_1^k contains at least one 'unit' vertex of f : $(\exists \tilde{\alpha} \in B_1^k, f(\tilde{\alpha}) = 1)$.

In conditions $\alpha)$, $\beta)$, $\gamma)$, we say that B_1^k is a maximal interval of the function f. d.n.f., composed of all elementary conjunctions, corresponding to maximal intervals of function f is named the reduced disjunctive normal form of f. The number of disjunctive members of this formula is considered as its complexity. Denoting by $r_k(f)$ the number of all maximal k–intervals of the function f we get the formula of complexity of the reduced disjunctive normal form of f:

$$\sum_{k=0}^{n} r_k(f).$$

2. On the Maximum Number of k-Dimensional Maximal Intervals of RBF

Consider the class $P_2(n)$ of all Boolean functions of n variables $x_1, x_2, \cdots, x_n$. Let $p,\ 0 < p < 1$ be a fixed number, and F_p – the probability distribution on $P_2(n)$, generated in the following way. The function $f \in P_2(n)$ is induced as a result of a randomized experiment, where the values of the function on vertices of E^n are derived randomly. The value 1 appears with a probability p and the 0 value – with a complementary probability $1-p$. The vertices of E^n take part in this experiment independently of each other, and the probabilistic distribution F_p over the set of Boolean functions is generated in this way. The probability of an individual Boolean function f under the distribution F_p depends on the balance between the 0 and 1 values of the function f (the volumes of the sets $\mathcal{N}_f$ and $E^n - \mathcal{N}_f$). For $f \in P_2(n)$, this probability is equal to $p^{|\mathcal{N}_f|}(1-p)^{2^n - |\mathcal{N}_f|}$. When $p = 1/2$ this probability is simply $1/2^{2^n}$ and the corresponding distribution becomes the uniform distribution over

the $P_2(n)$. We introduce the notation $r_k(f)$ for the number of k-dimensional maximal intervals of the function $f \in P_2(n)$. And let $r_k(n,p)$ be the average value of the number of k-dimensional maximal intervals of functions $f \in P_2(n)$ under the distribution F_p. It is easy to make sure, that

$$r_k(n,p) = \sum_{f \in P_2(n)} F_p(f) * r_k(f) \tag{1}$$

The number $r_k(n,p)$ in the expression (1) is given by its definition as a sum over all functions of $f \in P_2(n)$, counting all their k-dimensional maximal intervals and taking into account the probabilities of f in the distribution F_p.

Further evidence of these constructions is provided by the following scheme:

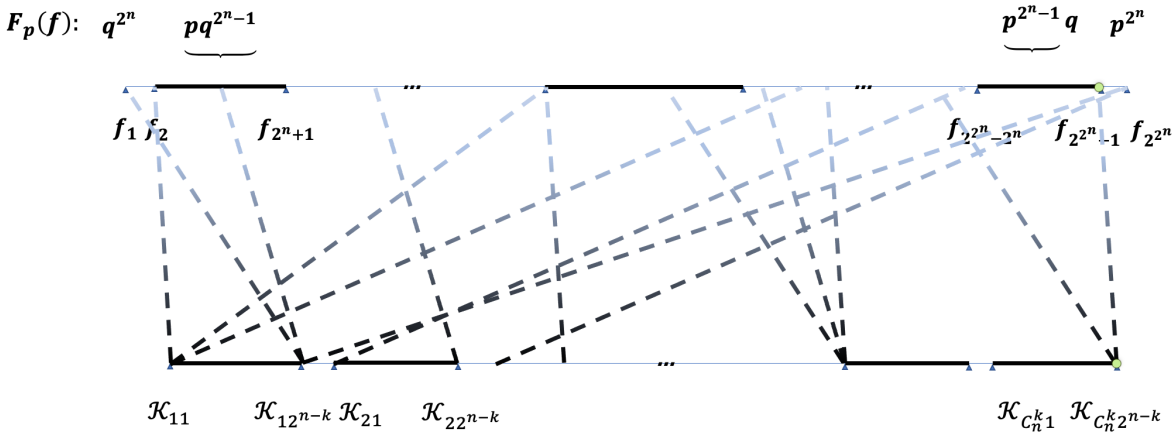

Fig. 2. This figure presents the bipartite graph of functions and k-dimensional maximal intervals. Upper line functions are placed in order of the number of their "true" values, from 0 to 2^k. Different functions include different numbers of k-dimensional maximal intervals and have different probabilities under the distribution F_p. Instead, each interval presented in the bottom line is connected to the same number of functions. This is because the sizes of intervals is the same. The order of intervals is by groups of intervals, that belong to the same direction. Numeration inside the functions with the same number of "ones" and inside the groups of intervals of the same direction is arbitrary.

Following [5], we change the order of counting in 1, first considering all k-dimensional intervals in E^n. We relay two events to these intervals: the one, about their maximality, and then the second, about the set of functions that accept the first event about maximality. In this regard, it is also convenient to split the E^n in parts: the current k-dimensional interval $\mathcal{K}$ and its all $n-k$ neighboring k-dimensional intervals $\mathcal{K}_1, \mathcal{K}_2, \cdots, \mathcal{K}_{n-k}$. This part, the current interval and its neighbors, covers an area $\mathcal{E}_1$ of $2^k(n-k+1)$ vertices of E^n. And the second part that we consider, consists of the complementary area $\mathcal{E}_2$ to $\mathcal{E}_1$ up to E^n. The probability of maximality of $\mathcal{K}$ for the function f becomes the product of probability of maximality of $\mathcal{K}$ together with the conditional probability of f when $\mathcal{K}$ is given to be maximal. The first probability equals $p^{2^k}(1-p^{2^k})^{n-k}$. The first and second parts consist of events, and their sums of probabilities are equal to 1 as a probabilistic distribution. Now, when we sum up the mentioned conditional probabilities with all f, we get the probability 1, and the final probability of maximality of $\mathcal{K}$, under the conditions of F_p, becomes $p^{2^k}(1-p^{2^k})^{n-k}$. It reminds us to take this probability for all k-dimensional intervals, obtaining the following equivalent form for (1),

$$r_k(n,p) = C_n^k 2^{n-k} p^{2^k} (1-p^{2^k})^{n-k}. \quad (2)$$

Theorem 1. *$r_k(n,p)$ is a concave function of the parameter k in the interval $[0,n]$.*

It is important to know the behavior of the function $r_k(n,p$ defined on the interval $[1,n]$. Initially, it is useful to calculate the values of the function at the boundary points of the domain of definition: $k = 0, 1, ..., n-1, n$. We give these values both for the arbitrary p and the value $1/2$.

Table 1: Values of $r_k(n,p)$ on boundary points, such as $k = 0, 1, ..., n-1, n$.

Boundary point values of $r_k(n,p)$		
Dimension k of maximal interval	$r_k(n,p)$	$r_k(n,1/2)$
$k=0$	$2^n p(1-p)^n$	$1/2$
$k=1$	$n2^{n-1}p^2(1-p^2)^{n-1}$	$(n/4)(3/2)^{n-1}$
...	...	...
$k=n-1$	$n2^{n-1}p^{2^{n-1}}(1-p^{2^{n-1}})$	$n2^{n-1}(1-1/2^{2^{n-1}})/2^{2^{n-1}})$
$k=n$	p^{2^n}	$1/2^{2^n}$

As we can see, both the left and right boundary point values of the interval $(0,n)$ are small, but there is a noticeable increase from left to right at the left end, and a decrease from left to right at the right end. To get a complete picture of the behavior, consider a number of special intermediate point values of the function at:

$$k_1 = log\frac{1}{-logp},\ k_0 = log\frac{logn}{-logp},\ and\ k_1 = log\frac{n}{-logp}.$$

The technical element of choosing of these values is in simple evaluation of sub-formula $E_k = 2^{2^k}$, which is an important functional part of the 1. Substituting k_1, k_0, and k_2 into E_k we get:

$$E_{k_1} = 1/2,\ E_{k_0} = 1/n,\ E_{k_2} = 1/2^n. \quad (3)$$

Let us start the proof of postulations 1-3. For this, conduct a preliminary analysis of the expression (2) for $r_k(n,p)$. Consider an arbitrary integer value function $k(n)$ that obeys the restriction $0 \le k(n) \le n$, and substitute it into the expression 2. We are interested in the behaviour of the received function $r_{k(n)}(n,p)$ depending on the parameter $k(n)$ as $n \to \infty$.

First let's make sure that with the increase of k the expression $r_k(n,p)$ increases monotonically by the $k \le [k_0]$, and then it decreases, when $]k_0[\le k$. By doing this we compose the relation

$$R_k = \frac{r_{k+1}(n,p)}{r_k(n,p)} = \frac{(n-k)p^{2^k}(1+p^{2^k})^{n-k}}{2(k+1)(1-p^{2^{k+1}})}. \quad (4)$$

This expression can be considered for an arbitrary (not only for the integer) assignment to the parameter k. We will follow by checking if this function is concave in the interval $0 < k < n$ for large n. The direct way of this is to derive the expression of the fraction

R_k and treat it for a possible constant/zero value of it. In such consideration, the most important role takes the part $A_k = (n-k)p^{2^k}$ of the base expression 4. Substituting k_0 into A_k we obtain that $(n-k_0)p^{2^{k_0}} = (n-k_0)p^{(\frac{logn}{-logp})} = (n-k_0)2^{logp(\frac{-logn}{logp})} = \frac{n-k_0}{n}$, which is converging to 1 as $n \to \infty$. With the help of formulas in Section 3. we see that the part $B_k = (1+p^{2^k})^{n-k}$ of (4) is limited at the point k_0: (6) gives $(1+p^{2^{k_0}})^n \to e$ as $n \to \infty$, so that $(1+p^{2^{k_0}})^{n-k_0}$ also tends to e. Compose the fraction B_{k+1}/B_k in the following form:

$$B_{k+1}/B_k = \frac{\left(1+p^{2^k}p^{2^k}\right)^{n-k-1}}{\left(1+p^{2^k}\right)^{n-k}} = \frac{\left(\frac{1+p^{2^k}p^{2^k}}{1+p^{2^k}}\right)^{n-k}}{1+p^{2^k}p^{2^k}} \tag{5}$$

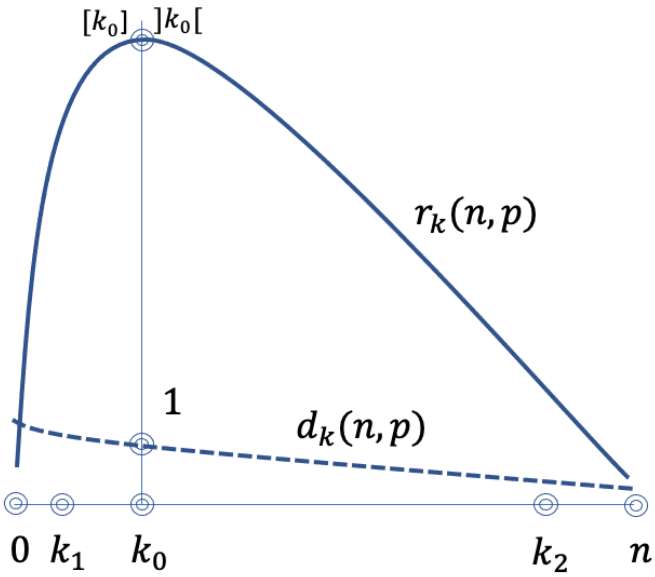

Fig. 3. Differential of growing $r_k(n,p)$.

Note that the fraction $\frac{1+p^{2^k}p^{2^k}}{1+p^{2^k}}$ is less than 1, so its $n-k$ degree is also less than 1. And the denominator of (5) is greater than 1 so that, finally, the expression (5) is less than 1 for all k, which means a monotonic decrease of the expression R_k in (5). In general, as k increases, all the factors of (4), other than B_k, decrease monotonically and, besides this, as $n \to \infty$, this expression tends to zero at the point k_0 and grows infinitely when $k = k_0 - 1$. Finally, we receive that with increasing k, for the beginning, $i_k(n,p)$ increases, achieving its maximal value at the point $[k_0]$ or $]k_0[$, and, then, it decreases.

3. On the Dependency of Number of k-Dimensional Maximal Intervals on k

Consider the parameter $k_2 = log\frac{n}{-logp}$. Since $0 < p < 1$, we have $k_2 = logn + c$, where c represents an absolute constant determined by the fixed value of p. We intend to obtain an asymptotic formula for $i_k(n,p)$ by the $n \to \infty$ for the values of k of the form $k_2 + const$. We make use of the following expressions $C_n^k \sim \frac{n^k}{k!}$, $(1-p^{2^k}) \sim 1$, and $n! \sim n^n e^{-n}\sqrt{2\pi n}$ as $n \to \infty$, which are based on the formulas

1. If $0 \le x \le 1$ and $0 \le y$, then

$$exp(x(1-\frac{x}{2})y) \le (1+x)^y \le exp(xy). \tag{6}$$

2. If $0 \leq x \leq 1$ and $0 \leq y$, then

$$(1-x)^y \leq exp(-xy); \text{ and} \tag{7}$$

$exp(-x(1-x)y) \leq (1-x)^y$, when additionally $0 \leq x \leq 1/2$.

3. If x and y be natural numbers, and $x \leq y$, then

$$(1-\frac{x}{y})^{\frac{x-1}{2}} \leq \prod_{i=1}^{x-1}(1-\frac{i}{y}) \leq (1-\frac{x}{2y})^{x-1}. \tag{8}$$

and are valid for the mentioned values of the parameter k, and for this reason

$$i_k(n,p) \sim \frac{n^k e^k 2^{n-k} p^{2^k}}{k^k\sqrt{2\pi k}} = \tilde{i}_k(n,p). \tag{9}$$

Theorem 2. *The probability, that functions of the class $P_2(n)$ under the distribution F_p have maximal intervals of sizes k, $k < [k_1]$ or $k > [k_2]$, where $k_1 = log\frac{1}{-logp}$ and $k_2 = log\frac{n}{-logp}$ tends to zero with $n \to \infty$.*

On the right side of (9) we have expression, that depends on the continuous argument k, and which is equivalent to the expression $i_k(n,p)$ for the integer values of the parameter k, of the form $k_2 + const$. In the mentioned area, $\tilde{i}_k(n,p)$ decreases monotonically with the increase of k, $\tilde{i}_{k_2}(n,p)$ tends to infinity, and $\tilde{i}_{k_2+1}(n,p)$ tends to zero, when $n \to \infty$, so that $i_k(n,p) \to 0$, for values $k >]k_2[$ and $i_k(n,p) \to \infty$ for values $k_0 \leq k \leq [k_2]$, by $n \to \infty$. Let us also denote, that we do not insist that $i_{]k_2[}(n,p)$ as $n \to \infty$ converges to any appropriate value.

In what follows, we will use the first Chebyshev inequality (1). The first inequality lets formulate an extension of a postulation from [29] for the case of the probability distribution F_p. Actually, if to consider the expression $i_k(f)$, as a parameter of $\pi(f)$ then for the values $k >]k_2[$ $i_k(n,p) \to 0$ by $n \to \infty$, and taking into the force the first inequality for the arbitrary $\epsilon(n) \geq 0$ $P(i_k(f) \geq \epsilon(n)) \to 0$ when $n \to \infty$.
A similar situation takes place in the region of small values of the parameter k. For the value $k = k_1$ and $p = 1/2$ by the (3) $p^{2^{k_1}} = 1/2$ and $r_{k_1}(n,p) \to \infty$ as $n \to \infty$. For $p > 1/2$, already for the value $k_1 - 1$, we observe that $r_{k_1-1}(n,p) \to 0$ as $n \to \infty$. This is just because $\frac{2^{n-k_1+1}}{1-p^{2^{k_1-1}}}$is a decreasing exponent, which together with C_n^k tends to 0.

4. Conclusion

This article has two goals: first, it considers the set of formulas needed to analyze the complexity of structures associated with a multidimensional unit cube, providing the necessary transformations and approximations for these formulas. Further, the paper considers a typical study for this field using these formulas. The problem under consideration estimates the complexity of the reduced disjunctive normal form of Boolean functions on average, or, what is the same, for almost the entire class of functions.

References

[1] Yu. I. Zhuravlev, "Set-Theoretical methods of algebra of logic, *Problemi Kibernetiki*, vol. 8, pp. 544, 1962.

[2] O. Lupanov and S. Yablonsky, *Discrete Mathematics and Mathematical Problems of Cybernetics*, Moscow, Nauka, 1974.

[3] A. I. Kitov and N. A. Krinitsky, *Electronic Computers*, Moscow: USSR Academy of Sciences, 1958.

[4] Yu. L. Vasiliev, "Difficulties of minimization of Boolean functions based on universal approaches", *Soviet Math. Dokl.*, vol. 171, no. 1, pp. 1316, 1966.

[5] V. Glagolev, "Some estimates of disjunctive normal forms in the algebra of logic, *Problems of Cybernetics*, Nauka, Moscow, vol. 19 pp. 7594, 1967.

[6] A. A. Sapozhenko, "Mathematical properties of almost all functions of algeΘbra of logic", *Discrete analysis*, vol. 10, pp. 91119, 1967.

[7] O. B. Lupanov, "Ob odnom metode sinteza skhem, *In: Izv. VUZ (Radiofizika)*, vol. 1, no.1, pp. 120140, 1958.

[8] L. H. Aslanyan, "On a recognition method, based on partitioning of classes by the disjunctive normal forms", *Kibernetika*, vol. 5 pp. 103110, 1975.

[9] L. H. Aslanyan, "Recognition algorithm with logical separators", *Collection of Works on Mathematical Cybernetics*, Computer Center, AS USSR, Moscow, pp. 116131, 1976.

[10] L. Aslanyan and J. Castellanos, "Logic based Pattern Recognition - Ontology content (1)", *Inf. Tech. and Applicat. (IJ ITA)*, vol. 1, pp. 206210, 2007.

[11] L. Aslanyan and V. Ryazanov, "Logic based Pattern Recognition - Ontology content (2)", *Inf. Theories and Applicat*, vol. 15, no. 4, pp. 314318, 2008.

[12] L. Aslanyan, H.Sahakyan, H.-D. Gronau and P. Wagner, Constraint satisfaction problems on specific subsets of the n-dimensional unit cube", *Proc. IEEE 10th Int. Comp. Sci. and Infor. Technol. (CSIT)*, Yerevan, Armenia, pp. 4752, 2015.

[13] L. Aslanyan and H. Sahakyan, "The splitting technique in monotone recognition", *Discrete Applied Mathematics*, vol. 216, pp. 502512, 2017.

[14] G. Putzolu and F. Mileto,"Average values of quantaties appearing in Boolean function minimization", *IEEE EC-13*, vol. 2, pp. 8792, 1964.

[15] G. Putzolu and F. Mileto, "Average values of quantaties appearing in multiple output Boolean minimization", *IEEE EC-14*, vol. 4, pp. 542552, 1965.

[16] L. H. Aslanyan, "On complexity of reduced disjunctive normal form of partial Boolean functions. I.", *Proceedings, Natural Sciences*, Yerevan State University, vol. 1, pp. 1118, 1974.

[17] L. H. Aslanyan, "On complexity of reduced disjunctive normal form of partial Boolean functions. II", *Proceedings, Natural Sciences*, Yerevan State University, vol. 3, pp. 1623, 1974.

[18] M. Skoviera. "Average values of quantities appearing in multiple output Boolean minimization", *Computers & Artificial Intelligence*, vol. 5, pp. 321334, 1986.

[19] E. Toman, "An upper bound for the average length of a dizjunktive normal form of a random Boolean function", *Computers & Artificial Intelligence*, vol. 2, pp. 1317, 1983.

[20] K. Weber, "Prime Implicants of Random Boolean Functions", *Journal of Information Processes and Cybernetics*, vol. 19, pp. 449458, 1983.

[21] D. Gardy, "Random Boolean expressions", *Computational Logic and Applications*, vol. 5, pp. 136, 2005.

[22] J. Boyar, R. Peralta and D. Pochuev, "On the multiplicative complexity of Boolean functions over the basis (and,mod2,1)", *Theoretical Computer Science*, vol. 235, no. 1, pp. 43–57, 2000.

[23] X. Gong and J. Socolar, "Quantifying the complexity of random Boolean networks", In: arXiv:1202.1540v3 [nlin.CG] 26 May 2012.

[24] P. Hrubes", On the complexity of computing a random Boolean funtion over the reals", *Electronic Colloquium on Computational Complexity Report*, no. 36, pp. 111, 2000.

[25] G. Sosa-Gomez, O. Paez-Osuna, O. Rojas, P. Lui del Angel Rodriguez, H. Kanarek and E. J. Madarro-Capo, "Con-struction of Boolean Functions from Hermitian Codes", *Mathematics*, MDPI 10.899, pp. 116, 2022.

[26] Chaubal Siddhesh Prashant, *Complexity Measures of Boolean Functions and their Applications*, Faculty of the Graduate School of The University of Texas at Austin 2020.

[27] P. Erdos, "Graph theory and probability", *Canad. J. Math*, vol. 11, pp. 3438, 1959.

[28] J. Spencer and P. Erdos, *Probabilistic Methods in Combinatorics*, Moscow: Mir, 1963.

[29] L. H. Aslanyan, "On implementation of reduced disjunctive normal form in the problem of extension of partial Boolean functions", *Junior Researcher, Natural Sciences*, Yerevan State University, vol. 20, no. 2, pp. 6575, 1974.

Պատահական բուլյան ֆունկցիաների ԿԴՆՁ կողմնորոշված վերլուծություն

Լևոն Հ. Ասլանյան, Իրինա Ա. Արսենյան, Վիլիկ Մ. Կարախանյան,
Հասմիկ Ա. Սահակյան

ՀՀ ԳԱԱ Ինֆորմատիկայի և ավտոմատացման պրոբլեմների ինստիտուտ, Երևան, Հայաստան
e-mail: kavilik@gmail.com

Ամփոփում

Այս հոդվածն ունի երկու նպատակ, նախ այն քննարկում է բանաձևերի մի շարք, որոնք անհրաժեշտ են բազմաչափ միավոր խորանարդի հետ կապված կառուցվածքների բարդությունը վերլուծելու համար՝ ապահովելով անհրաժեշտ փոխակերպումներ և մոտարկումներ այս բանաձևերի համար: Ավելին, հոդվածը քննարկում է այս ոլորտի համար որոշ բնորոշ ուսումնասիրություն՝ օգտագործելով այս բանաձևերը: Քննարկվող ընթացակարգը գնահատում է բուլյան ֆունկցիաների կրճատված դիզյունկտիվ նորմալ ձևի բարդությունը միջինում կամ, որ նույնն է, դասի գրեթե բոլոր ֆունկցիաների համար:

Բանալի բառեր՝ Բուլյան ֆունկցիա, բազմաչափ միավոր խորանարդ, բարդություն, ասիմպտոտիկա, կրճատված դիզյունկտիվ նորմալ ձև:

Аналитика ориентированная на СДНФ случайных булевых функций

Левон А. Асланян, Ирина А. Арсенян, Вилик М. Караханян, Асмик А. Саакян

Институт проблем информатики и автоматизации НАН РА, Ереван, Армения
e-mail: kavilik@gmail.com

Аннотация

Данная статья преследует две цели: во-первых, в ней рассматривается набор формул, необходимых для анализа сложности структур, связанных с многомерным единичным кубом, предоставляя необходимые преобразования и аппроксимации для этих формул. Далее, в статье рассматривается типичное исследование для данной области с использованием этих формул. Рассматриваемая проблема оценивает сложность сокращенной дизъюнктивной нормальной формы булевых функций в среднем, или, что то же самое, почти для всего класса функций.

Ключевые слова: булева функция, многомерный единичный куб, сложность, асимптотика, сокращенная дизъюнктивная нормальная форма.

Mathematical Problems of Computer Science 59, 27–34, 2023.
doi: 10.51408/1963-0099

UDC 510.64

The Relationship Between the Proof Complexities of Linear Proofs in Quantified Sequent Calculus and Substitution Frege Systems

Hakob A. Tamazyan

Yerevan State University, Yerevan, Armenia
e-mail: hakob.tamazyan@ysu.am

Abstract

It has formerly been proved that there is an exponential speed-up in the number of lines of the quantified propositional sequent calculus over substitution Frege systems when considering proofs as trees. This paper shows that a linear proof of any quantifier-free tautology in quantified propositional sequent calculus can be transformed into a linear proof of the same tautology in a substitution Frege systems with no more than polynomially increasing proof lines and size.
Keywords: Sequent systems, Frege systems, Proof size, Number of proof lines, Exponential speed-up.
Article info: Received 23 March 2023; sent for review 2 April 2023; accepted 19 May 2023.

1. Introduction

The existence of a propositional proof system that has proofs of polynomial size for all tautologies is equivalent to the equation $NP = co\text{-}NP$ [1]. This observation has gained attention in recent years, leading to the examination of new proof systems. Through the discovery of new systems, the computational power of existing ones is gaining a greater understanding. A hierarchy of proof systems has been established based on two complexity measures (size and lines), and the relationships between these systems are being explored. Alessandra Carbone in [2] compared the number of derivation lines in the form of a tree in some propositional calculus systems and revealed a distinctive property of the quantified propositional sequent calculus (QPK system). Namely, for some sequences of formulas, the QPK system has an exponential speed-up by lines with respect to the substitution sequent calculus (SPK system) and substitution Frege systems (SF systems) when proofs are considered as trees. It was shown in [3] that the lines of linear proofs of the same formulae families in all three systems are the same by order. Later, in [4], the same result was achieved if one considers the sizes of linear proofs of the same formulae families for comparison.

In this paper, the relationship between the proof complexities of linear proofs in QPK and SF has been investigated for all quantifier-free tautologies: it turns out that QPK system has no significant advantage over SF when only linear proofs are considered. Specifically, after the transformation of linear QPK-proof of a quantifier-free tautology into a linear SF-proof of the same tautology by some algorithm, both complexities (the number of lines and sizes) of linear proofs in SF can increase polynomially at most.

2. Preliminaries

First and foremost, lets define several proof systems according to [1, 5, 6].

The Frege system F uses a denumerable set of propositional variables, a finite, complete set of propositional connectives. It has a finite set of inference rules defined by a figure of the form $\frac{A_1A_2...A_m}{B}$(the rules of inference with zero hypotheses are the schemes of axioms). F must be sound and complete, i.e., for each rule of inference $\frac{A_1A_2...A_m}{B}$ every truth-value assignment, satisfying $A_1A_2...A_m$, also satisfies B, and F must prove every tautology.

The Substitution Frege system SF is defined by adding to F the substitution rule

$$\frac{A(p)}{A(B)}$$

where simultaneous substitution of the formula B is allowed for the variable p.

The LK Sequent calculus was introduced by Gentzen [7] for first-order logic. Each line in LK-proof is a sequent: a sequent is written in the form:

$$A_1, \ldots, A_n \to B_1, \ldots, B_m$$

where $A_1, \ldots, A_n$ and $B_1, \ldots, B_m$ are formulas. We denote these sequences of formulas by capital Greek letters Γ, Δ, etc. As a quantifier symbol in LK, we will include only the universal quantification $\forall$. The existential quantification symbol $\exists$ will be added by the following definition:

$$(\exists x)A(x) \equiv \neg(\forall x)\neg A(x).$$

The inference rules of the sequent calculus LK are as follows:

- Initial sequents are sequents of the following form:

  $$A \to A$$

  where A is any formula.

- Structural rules:

$$Weakening : left \quad \frac{\Gamma \to \Delta}{A, \Gamma \to \Delta} \qquad Weakening : right \quad \frac{\Gamma \to \Delta}{\Gamma \to \Delta, A}$$

$$Exchange : left \quad \frac{\Gamma_1, A, B, \Gamma_2 \to \Delta}{\Gamma_1, B, A, \Gamma_2 \to \Delta} \qquad Exchange : right \quad \frac{\Gamma \to \Delta_1, A, B, \Delta_2}{\Gamma \to \Delta_1, B, A, \Delta_2}$$

$$Contraction : left \quad \frac{\Gamma_1, A, A, \Gamma_2 \to \Delta}{\Gamma_1, A, \Gamma_2 \to \Delta} \qquad Contraction : right \quad \frac{\Gamma \to \Delta_1, A, A, \Delta_2}{\Gamma \to \Delta_1, A, \Delta_2}$$

- Logical rules:

$$\neg : left \quad \frac{\Gamma \to \Delta, A}{\neg A, \Gamma \to \Delta} \qquad \neg : right \quad \frac{A, \Gamma \to \Delta}{\Gamma \to \Delta, \neg A}$$

$$\wedge : left \quad \frac{A, B, \Gamma \to \Delta}{A \wedge B, \Gamma \to \Delta} \qquad \wedge : right \quad \frac{\Gamma \to \Delta, A \quad \Gamma \to \Delta, B}{\Gamma \to \Delta, A \wedge B}$$

$$\vee : left \quad \frac{A, \Gamma \to \Delta \quad B, \Gamma \to \Delta}{A \vee B, \Gamma \to \Delta} \qquad \vee : right \quad \frac{\Gamma \to \Delta, A, B}{\Gamma \to \Delta, A \vee B}$$

$$\supset : left \quad \frac{\Gamma \to \Delta, A \quad B, \Gamma \to \Delta}{A \supset B, \Gamma \to \Delta} \qquad \supset : right \quad \frac{A, \Gamma \to \Delta, B}{\Gamma \to \Delta, A \supset B}$$

- The cut rule:

$$\frac{\Gamma \to \Delta, A \quad A, \Gamma \to \Delta}{\Gamma \to \Delta}$$

Let us denote by PK the sequent calculus LK, where the rules are restricted to propositional logic.

The substitution system SPK is defined as the propositional sequent calculus PK with an additional substitution rule:

$$S_p^B \quad \frac{\Gamma \to \Delta, A(p)}{\Gamma \to \Delta, A(B)},$$

where simultaneous substitution of the formula B is allowed for the variable p, and p does not appear in Γ, Δ.

The quantifier system QPK is defined as the propositional sequent calculus PK, where new quantification rules are added:

$$\forall : left \quad \frac{A(B), \Gamma \to \Delta}{(\forall q)A(q), \Gamma \to \Delta} \qquad \forall : right \quad \frac{\Gamma \to \Delta, A(p)}{\Gamma \to \Delta, (\forall q)A(q)}$$

where B is any formula such that no free variable occurrence in B becomes bounded in $A(B)$, and with the restriction that the atom p does not occur freely in the lower sequents of $\forall : right$.

Notice that the the following two inferences can be derived in QPK system using the definition of the quantifier $\exists$:

$$\exists : left \quad \frac{A(p), \Gamma \to \Delta}{(\exists q)A(q), \Gamma \to \Delta} \qquad \exists : right \quad \frac{\Gamma \to \Delta, A(B)}{\Gamma \to \Delta, (\exists q)A(q)}$$

$$\frac{\dfrac{\dfrac{A(p), \Gamma \to \Delta}{\Gamma \to \Delta, \neg A(p)}}{\Gamma \to \Delta, (\forall q)(q)}}{\neg(\forall q)\neg A(q), \Gamma \to \Delta} \qquad \frac{\dfrac{\dfrac{\Gamma \to \Delta, A(B)}{\neg A(B), \Gamma \to \Delta}}{(\forall q)(q), \Gamma \to \Delta}}{\Gamma \to \Delta, \neg(\forall q)\neg A(q)}$$

3. Main Results

For a given linear proof in QPK with n number of lines and proof size s, one can always find a linear proof in SPK of the same tautology having $O(n^2)$ lines and $O(s^5)$ proof size.

First of all, notice that for any linear proof in SPK, there exists a linear proof in QPK of the same tautology with the same number of lines. The sequent $(\forall p)A(p), \Gamma \to \Delta, A(B)$ is provable for all A, B, and the sequent $\Gamma \to \Delta, (\forall p)A(p)$ is derivable from $\Delta \to \Delta, A(p)$. Hence, after combining them through a cut rule, one derives $\Gamma \to \Delta, A(B)$. Here we examine the relationship between these systems in the opposite scenario.

Lemma. *For $n, m \geq 0$ and p not appeared in Γ, Δ, the following inference*

$$\frac{\Gamma, A_1(p), \ldots, A_n(p) \to \Delta, A_{n+1}(p), \ldots, A_{n+m}(p)}{\Gamma, A_1(B), \ldots, A_n(B) \to \Delta, A_{n+1}(B), \ldots, A_{n+m}(B)}$$

can be achieved in SPK system with $O(n+m)$ lines using the substitution rule only once.

Proof. First, let's prove these additional inferences:

1. $$\frac{\Gamma \to \Delta, \neg A}{A, \Gamma \to \Delta}$$

$$\dfrac{\dfrac{\Gamma \to \Delta, \neg A \quad A \to A}{\Gamma \to \Delta, \neg A \quad \neg A, A \to}}{A, \Gamma \to \Delta}$$

2. $$\frac{\Gamma \to \Delta, A \vee B}{\Gamma \to \Delta, A, B}$$

$$\dfrac{\dfrac{\dfrac{\Gamma \to \Delta, A \vee B \quad A \to A \quad B \to B}{\Gamma \to \Delta, A \vee B \quad A \to A, B \quad B \to B}}{\Gamma \to \Delta, A \vee B \quad A \to A, B \quad B \to A, B}}{\dfrac{\Gamma \to \Delta, A \vee B \quad A \vee B \to A, B}{\Gamma \to \Delta, A, B}}$$

3. $$\frac{\Gamma, A \wedge B \to \Delta}{\Gamma, A, B \to \Delta}$$

$$\dfrac{\dfrac{\dfrac{\Gamma, A \wedge B \to \Delta \quad A \to A \quad B \to B}{\Gamma, A \wedge B \to \Delta \quad A, B \to A \quad B \to B}}{\Gamma, A \wedge B \to \Delta \quad A, B \to A \quad A, B \to B}}{\dfrac{\Gamma, A \wedge B \to \Delta \quad A, B \to A \wedge B}{\Gamma, A, B \to \Delta}}$$

The final proof will look like this:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma, A_1(p), \ldots, A_n(p) \to \Delta, A_{n+1}(p), \ldots, A_{n+m}(p)}{\Gamma, A_1(p) \wedge A_2(p), \ldots, A_n(p) \to \Delta, A_{n+1}(p), \ldots, A_{n+m}(p)}}{\vdots}}{\Gamma, A_1(p) \wedge \ldots \wedge A_n(p) \to \Delta, A_{n+1}(p), \ldots, A_{n+m}(p)}}{\vdots}}{\Gamma, A_1(p) \wedge \ldots \wedge A_n(p) \to \Delta, A_{n+1}(p) \vee \ldots \vee A_{n+m}(p)}}{\Gamma \to \Delta, A_{n+1}(p) \vee \ldots \vee A_{n+m}(p), \neg(A_1(p) \wedge \ldots \wedge A_n(p))}}{\Gamma \to \Delta, A_{n+1}(p) \vee \ldots \vee A_{n+m}(p) \vee \neg(A_1(p) \wedge \ldots \wedge A_n(p))}}{\Gamma \to \Delta, A_{n+1}(B) \vee \ldots \vee A_{n+m}(B) \vee \neg(A_1(B) \wedge \ldots \wedge A_n(B))}}{\Gamma \to \Delta, A_{n+1}(B) \vee \ldots \vee A_{n+m}(B), \neg(A_1(B) \wedge \ldots \wedge A_n B)}}{\Gamma, A_1(B) \wedge \ldots \wedge A_n(B) \to \Delta, A_{n+1}(B) \vee \ldots \vee A_{n+m}(B)}}{\vdots}}{\Gamma, A_1(B), \ldots, A_n(B) \to \Delta, A_{n+1}(B), \ldots, A_{n+m}(B)}$$

Note that in this proof the substitution rule is applied only once. ■

Theorem 1. *For a given linear proof in QPK of some quantifier-free tautology with n number of lines, there exists a linear proof in SPK of the same tautology having $O(n^2)$ number of lines.*

Proof. Suppose P is a given linear proof in QPK. Since P is the proof of a quantifier-free tautology, if a formula with a quantifier appears in the proof, then it must disappear at some point in the next lines. These formulas can appear either by quantification rules or by weakening rules, and the cut rule is the only inference rule capable of removing a formula from the sequent. Notice that if we apply the cut rule to two sequents and some formula A with a quantifier is removed, then it is impossible that both of these sequents got this quantifier by the $\forall : left$ rule.

First of all, we will remove all applications of the $\forall : left$ rule in the proof of P. Let $(\forall q)A(q)$ be some formula or subformula in the proof. Suppose it appeared by $\forall : right$ rule that infers $\Gamma \to \Delta, (\forall q)A(q)$ from $\Gamma \to \Delta, A(p)$. Since p does not occur free in sequent $\Gamma \to \Delta, (\forall q)A(q)$, instead of the $\forall : right$ rule, we can apply the substitution rule to $\Gamma \to \Delta, A(p)$ and substitute p with some new variable k that did not appear throughout the proof. If $(\forall q)A(q)$ appeared by weakening rules, we will replace it with the formula $A(k)$, where k is again some new variable that did not appear throughout the proof. According to the previously mentioned claim, the formula $(\forall q)A(q)$ should have been removed at some point via the cut rule. Therefore, just before the application of cut rule, we will substitute the variable k with the corresponding matching formula to be able to apply the cut rule successfully. This substitution is allowed since k does not appear in the remaining formulas of the sequent.

This removal of formulas with quantifiers from the proof can have the following effects.

Firstly, since these formulas have been replaced with different ones, the contraction rule can not be applied to these replacements anymore, as they can differ from each other. Therefore, instead of applying the contraction rule to them, in the next lines we will apply the same inference rules to both of them. As these formulas should disappear in one of the next lines by the cut rule, we will apply the cut-elimination rule twice so that both of them

will be removed. There are $O(n)$ applications of the contraction rule, then after this change, the number of lines will become $O(n^2)$. However, according to the lemma, the number of applications of the substitution rule will not change and will remain $O(n)$.

Secondly, the $\forall : left$ rule that transformed some sequent $A(B), \Gamma \rightarrow \Delta$ into $(\forall q)A(q), \Gamma \rightarrow \Delta$, will not be applied to the proof, and the formula B will appear in the next lines. Hence, there might be an application of the substitution rule in these next lines that substitutes some variable x into some formula C so that x also appears in the formula B. This means that besides the formula C, there can also be other formulas with the variable x in the sequent. Therefore, to fix this, we will apply the substitution to these formulas too. Considering that the number of applications of the $\forall : left$ rule was $O(n)$ and removing each application of the contraction rule adds just one formula to the sequent, the number of such formulas in the sequent will be $O(n)$. Therefore, according to the lemma, each such substitution will require $O(n)$ additional lines. Since there are $O(n)$ applications of the substitution rule, this change will add $O(n^2)$ number of lines to our proof. This will conclude the transformation process, and the transformed SPK proof will have $O(n^2)$ lines. ■

Theorem 2. *For a given linear proof in QPK of some quantifier-free tautology with a proof size s, there exists a linear proof in SPK of the same tautology having $O(s^5)$ proof size.*

Proof. Suppose P is a given linear proof in QPK with n number of lines and proof size s. Let P' be the transformed SPK proof according to the process described above. To calculate its size, let's dive into the transformation process step by step.

We replaced each application of the $\forall : right$ rule with a substitution rule to substitute one variable with another. The formulas with quantifiers that appeared by weakening rules have been replaced by formulas with the same size. Afterwards, we added a substitution before the application of the cut rule to match the corresponding formula. All these steps change the number of proof lines and the proof size linearly. Let's denote them by n', s', respectively.

Moreover, we removed all applications of the $\forall : left$ rule. Therefore, if some application of the $\forall : left$ rule transformed the sequent $A(B), \Gamma \rightarrow \Delta$ into $(\forall q)A(q), \Gamma \rightarrow \Delta$, then after the removal, the formula B will appear in the next lines. This will increase the proof size by at most $n' \cdot |A(B)|$, where $|A(B)|$ is the size of the formula $A(B)$. Removing the i^th application of the $\forall : left$ rule increases the proof size by at most $n' \cdot |A_i(B_i)|$, then removing all of them will add no more than

$$\sum_i n' \cdot |A_i(B_i)| = n' \cdot \sum_i |A_i(B_i)| \leq n' \cdot s' \leq s'^2$$

to the proof size. As s' is $O(s)$, after this step, the proof size will be $O(s^2)$ and the number of lines will remain $O(n)$.

Removing applications of the contraction rule has the following two effects on the proof size.

First of all, it will keep the eliminated formula in a sequent, so it will appear in the next lines. The added proof size can be calculated completely like the previous method. Since the number of applications of the contraction rule is $O(n)$ and the proof size is $O(s^2)$, this change will make the proof size $O(s^3)$. The number of lines will remain $O(n)$.

The second effect of removing applications of the contraction rule will be applying the same inference rules to both formulas. Since the proof size is $O(s^3)$, then applying the same

inference rule to the previously eliminated formula can increase the proof size by $O(s^3)$. The number of applications of the contraction rule is $O(n)$, and since $n \leq s$, the overall proof size will become $O(s^4)$.

Finally, the removal of the $\forall : left$ rule causes some substitution steps to also substitute the same variable in several other formulas of the same sequent. Notice that all these substitution steps were $\forall : right$ rule replacements that substitute one variable with another, as otherwise we won't face such a problem. Each such substitution that simultaneously substitutes the same variable in these sequent formulas required $O(n)$ lines. If the i^th such substitution is applied to the sequent S_i, then this change will overall add no more than

$$\sum_i c \cdot n \cdot |S_i| = c \cdot n \cdot \sum_i |S_i| \leq c \cdot s \cdot \sum_i |S_i|$$

to the proof size, where $|S_i|$ is the size of the sequent S_i and c is some constant. $\sum_i |S_i|$ is smaller than the current proof size, therefore the transformed SPK proof will have $O(s^5)$ size. ■

Corollary. *Since the system SPK is polynomially equivalent to the system SF, there is a transformation of a linear proof of any quantifier-free tautology in QPK into a linear proof in the system SF that increases the proof lines and size at most polynomially.*

4. Conclusion

This work described an algorithm according to which any QPK linear proof can be transformed into a SF linear proof by increasing its lines and size to at most a polynomial extent. The obtained results show that the QPK system does not have a substantial advantage over the system SF in terms of linear proofs.

References

[1] S. A. Cook and A. R. Reckhow, "The relative efficiency of propositional proof systems", *Symbolic Logic*, vol. 44, pp. 36–50, 1979.

[2] A. Carbone, "Quantified propositional logic and the number of lines of tree-like proofs", *Studia Logica*, vol. 64, pp. 315–321, 2000.

[3] H. A. Tamazyan and A. A. Chubaryan, "On proof complexities relations in some systems of propositional calculus, *Mathematical Problems of Computer Science*, vol. 54, pp. 138–146, 2020.

[4] L. A. Apinyan and A. A Chubaryan, "On sizes of linear and tree-like proofs for any formulae families in some systems of propositional calculus", *Mathematical Problems of Computer Science*, vol. 57, pp. 47–55, 2022.

[5] P. Pudlák, *The Lengths of Proofs*, in S. Buss (ed.), Handbook of Proof Theory, Elsevier, vol. 137, pp. 547-637, 1998.

[6] J. Krajíček, *Proof Complexity, Encyclopedia of Mathematics and Its Applications*, Cambridge University Press, vol. 170, 2019.

[7] G. Gentzen, "Die Widerspruchsfreiheit der reinen Zahlentheorie", *Mathematische Annalen*, vol. 112, pp. 493–565, 1936.

Գծային արտածումների բարդությունների կապը ծավալիչներով սեկվենցիալ համակարգում և տեղադրման կանոնով Ֆրեգեի համակարգերում

Հակոբ Ա. Թամազյան

Երևանի պետական համալսարան, Երևան, Հայաստան
e-mail: hakob.tamazyan@ysu.am

Ամփոփում

Նախկինում ապացուցվել է, որ ծավալիչներով սեկվենցիալ համակարգում առկա է քայլերի քանակի էքսպոնենցիալ արագացում տեղադրման կանոնով Ֆրեգեի համակարգերի նկատմամբ, երբ դիտարկում ենք ծառային արտածումները: Այս հոդվածը ցույց է տալիս, որ առանց ծավալիչների, ցանկացած նույնաբանության գծային արտածումը ծավալիչներով սեկվենցիալ համակարգում հնարավոր է վերածել նույն նույնաբանության գծային արտածման տեղադրման կանոնով Ֆրեգեի համակարգերում՝ ունենալով արտածման քայլերի քանակի և երկարության առավելագույն բազմանդամային աճ:

Բանալի բառեր՝ սեկվենցիալ համակարգեր, Ֆրեգեի համակարգեր, արտածման երկարություն, արտածման քայլերի քանակ, էքսպոնենցիալ արագացում:

Связь между сложностями доказательств линейных выводов в системе секвенциального исчисления с кванторами и системах Фреге с правилом подстановки

Акоб А. Тамазян

Ереванский государственный университет, Ереван, Армения
e-mail: hakob.tamazyan@ysu.am

Аннотация

Ранее было доказано, что существует экспоненциальное ускорение количества шагов в системе секвенциального исчисления высказываний с кванторами по сравнению с системами Фреге с правилом подстановки, когда мы рассматриваем выводы в виде деревьев. Эта статья показывает, что линейный вывод любой бесквантворной тавтологии в системе секвенциального исчисления высказываний с кванторами можно превратить в линейный вывод той же тавтологии в системах Фреге с правилом подстановки с не более чем полиномиально возрастающим количеством шагов и длиной вывода.

Ключевые слова: секвенциальные системы, системы Фреге, длина вывода, количество шагов вывода, экспоненциальное ускорение.

Mathematical Problems of Computer Science 59, 35–44, 2023.
doi: 10.51408/1963-0100

UDC 004.75

Data Compression-Aware Performance Analysis of Dask and Spark for Earth Observation Data Processing

Arthur G. Lalayan

Institute for Informatics and Automation Problems of NAS RA, Yerevan, Armenia
National Polytechnic University of Armenia, Yerevan, Armenia
e-mail: arthurlalayan97@gmail.com

Abstract

High-performance computing is a good choice for handling Big Earth Observation data, allowing the processing of the data in a distributed and performance-efficient way using in-memory computing frameworks. The data compression technique reduces the amount of storage and network transfer time and improves processing performance. The article aims to investigate the effectiveness of widely used distributed data processing frameworks in conjunction with lossless data compression techniques, to find the optimal compression method and processing framework for specific earth observation workflows. Normalized Difference Vegetation Index has been evaluated for the territory of Armenia, obtaining data from the Sentinel satellite and considering the supported compression methods to compare the performance of in-memory Dask and Spark frameworks. Experiments show that the Zstandard compression method and the Dask framework are the best choices for such workflows.
Keywords: Earth observation, HPC, Spark, Dask, Distributed computing, Data compression.
Article info: Received 29 January 2022; sent for review 7 February 2023; received in revised form 15 March 2023; accepted 17 April 2023.
Acknowledgement: The research was supported by the Science Committee of the Republic of Armenia and the University of Geneva Leading House by the projects entitled Self-organized Swarm of UAVs Smart Cloud Platform Equipped with Multi-agent Algorithms and Systems (Nr. 21AG-1B052), Remote sensing data processing methods using neural networks and deep learning to predict changes in weather phenomena (Nr. 21SC- BRFFR-1B009), and ADC4SD: Armenian Data Cube for Sustainable Development.

1. Background and Motivation

Earth Observation (EO) satellite data are necessary for environmental monitoring and gathering vital information about various Earth layers [1]. Specifically, EO data are widely used

to monitor the atmosphere including air pollution [2] and temperature [3], the oceans considering sea pollution and ocean acidity [4], and ground, such as deforestation [5] and forest fire [6], as well as to detect climatic changes [7].

To facilitate work with EO data, Australian researchers [8] have provided an open-source Open Data Cube (ODC) [9], which is deployed and widely used by several communities from different countries, including Armenia [10]. Nevertheless, the ODC communities still encounter the Big EO data processing challenge requiring high-performance computational (HPC) resources. For instance, the Sentinel-2 satellite [11] provides approximately 200-300 GB, 3 TB, and 36 TB of daily, monthly, and annual data for the territory of Armenia. Handling this amount of data is a complex task. Therefore, HPC is the right choice to improve data processing performance using distributed computing techniques. Thus, the Big EO data processing obstacle is coping with using open-source Apache Spark [12] and Dask [13] frameworks, which can process data in parallel by dividing them into chunks, processing them in a distributed way using computational clusters, and aggregating the result. Both frameworks have master-slave architecture, where slave nodes are worker nodes executing functions in parallel, and the master node is the driver or scheduler to manage them. Spark ecosystem supports many projects in data streaming, SQL analytics, and machine learning. Spark is a multi-language engine that processes and analyzes data, while Dask is a Python library. Therefore, Spark has its ecosystem APIs and memory models, while Dask uses them from the Python ecosystem. However, these frameworks have some differences and limitations in finding an optimal solution for EO data processing workflows.

Besides using HPC, the format of EO satellite images also has a crucial influence on performance. The data compression techniques can reduce storage usage and the number of I/O operations, improving processing performance. Recent studies [14, 15] show that compression methods combined with HPC can significantly enhance the performance of Big data workflows. One of the optimal satellite image formats is Cloud Optimized GeoTIFF (COG) [16], which provides essential advantages compared to traditional formats, such as NetCDF [17]. COG format provides an HTTP range request to extract a part of the data. Hence, when extracting EO data using COG, there is no need to download the entire image and then extract the area of interest as in the NetCDF format. Besides the mentioned benefit, both COG and NetCDF formats support data compression methods.

Several studies [18, 19, 20] evaluate and compare the performance of the frameworks for particular cases, such as data-intensive neuroimaging pipelines [18], different applications of molecular dynamics [20], and scientific image analytics [19]. Nevertheless, they did not consider performance-tuning techniques, such as data compression.

The main objective of the article is to investigate the efficacy of widely used distributed data processing frameworks, such as Dask and Spark, in combination with lossless data compression methods, to enhance the performance of EO data processing. The methodology involved evaluating the approach on the Armenian hybrid research computing platform, and the results obtained from the evaluation could be used by EO communities to make informed decisions about improving their data processing performance.

2. Methodology

A test-bed platform for EO data processing has been deployed to execute EO data processing functions and compare the performances in Spark and Dask. The platform is a container-based solution within the Kubernetes system, enabling evaluating and comparing

the environments' performance. It relies on the computational resources of the Armenian hybrid research computing platform [21]. Fig. 1 shows the architecture of the experimental platform.

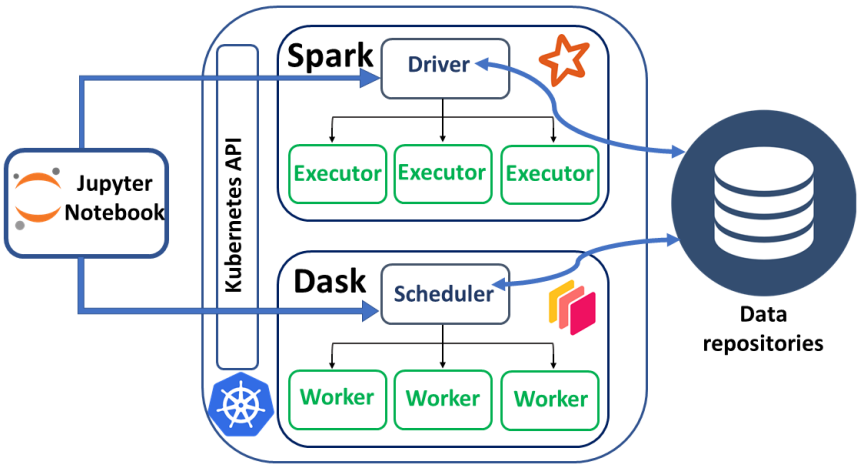

Fig. 1. Test-bed platform based on Spark and Dask.

As the figure shows, each node scheduler/driver or worker/executor corresponds to a pod in Kubernetes with some fixed computational resources. It is possible to configure the computational resource characteristics of nodes with Kubernetes API. The Jupyter Notebook [22] corresponds to the FrontEnd of the Spark and Dask cluster BackEnd. It connects to Dask and Spark of master nodes, configures environments by providing the number of worker nodes and computational resources for each node, requests to process EO data using Dask and Spark clusters, and visualizes the output. Dask and Spark clusters fetch data from repositories of either local Armenian DataCube [23] or global EO data providers. Armenian DataCube [10] provides data from Landsat 5, 7, 8 [24], and Sentinel-2 satellites, and one of the global EO data providers is Sentinel-2 Cloud-Optimized GeoTIFFs [25].

The functionality evaluation of the Dask and Spark frameworks is quite interesting. Dask is a flexible Python library, which makes it easy to migrate and execute the old-written Python code in a distributed manner. Moreover, Python is widely used in EO data workflows, and various useful libraries provide vital tools to make the work with EO data easier. However, working with EO data in Spark is a little tricky because the execution of the old-written codes in the Spark environment is impossible, as it supports APIs of its ecosystem, therefore, the code adjustment is inevitable. The GeoPySpark library [26] makes working with EO data somewhat easier in Spark. So the data processing function can be easily parallelized only in Dask, considering the limitations and complexity of using Spark.

As EO data processing applications, the Normalized Difference Vegetation Index (NDVI) [27] was evaluated during the experiments, which provides information for monitoring the health of the vegetation. The formula of the index is presented in (1).

$$NDVI = \frac{NIR - RED}{NIR + RED}, \tag{1}$$

where RED is the red band, and NIR is the near-infrared band. All bands and the calculation result are matrices or images and the NDVI index is calculated from Sentinel-2 satellite images.

Several experiments were conducted with different parameters to evaluate the performances of Dask and Spark using the developed experimental platform. Table 1 presents all parameters and their values.

Table 1: Experimental parameters and their values.

Parameter name	**Possible values**
Environment	Dask and Spark
Input Data sizes	16, 32, 64 GBs
Number of workers	4, 8, 16, 32
Applications	NDVI
Compression methods	None, Deflate, LZW, Packbits, and Zstandard

3. Experimental Results

Data compression techniques reduce the actual size of data, resulting in savings in storage space, providing faster network transmission times, and improving the performance of processing. EO data repositories, which provide satellite images in COG format, such as Sentinel-2 COGs, by default, use Deflate compression method to reduce the downloading time of satellite images and save some storage space. Besides the Deflate method, several compression methods, either lossy or lossless, could be applied with COGs. The accuracy of the satellite image is essential, as the spatial resolution of the Sentinel-2 image is 10m [10], which corresponds to the surface area measured on the ground represented by each pixel. Therefore, the compression methods used for optimization should be lossless to ensure accurate results. The COG format supports several lossless compression methods, such as Deflate [28], LZW [29], Packbits [30], and Zstandard [31].

EO band tiles come in three different sizes (light, medium, and heavy) by which the compression factor is estimated to understand the average compression ratio of the method. The light band tiles (coastal, water vapor, etc.) usually have up to 5-10 MB size, medium 50-70 MB (Short-wave infrared ($SWIR$), vegetation red edge, etc.), and heavy 200-250 MB (RED, NIR, etc.). They consider all types of possible lossless compression methods. The compression ratio is calculated for each method by dividing the compressed data size by the original uncompressed data size. The compression ratios for various compression methods are presented in Fig. 2.

The figure shows that the best compression factor is provided by the Zstandard method, whereas the worst one is provided by the Packbits method. Zstandard codec compresses the band image more than the Deflate does, which is by default used by the Sentinel-2 COGs repository. Therefore, using Zstandard instead of Deflate will lead to more storage savings, and less network transfer time and I/O operations. The storage reduction, in this case, is 34 % compared with the uncompressed data and 16 % compared with Deflate. The compression ratio of the Packbits method for the heavy tiles is close to 1, which means that the method is useless for data size reduction since the actual size and compressed data size will be the same. Besides the storage saving, further data processing is also essential, as

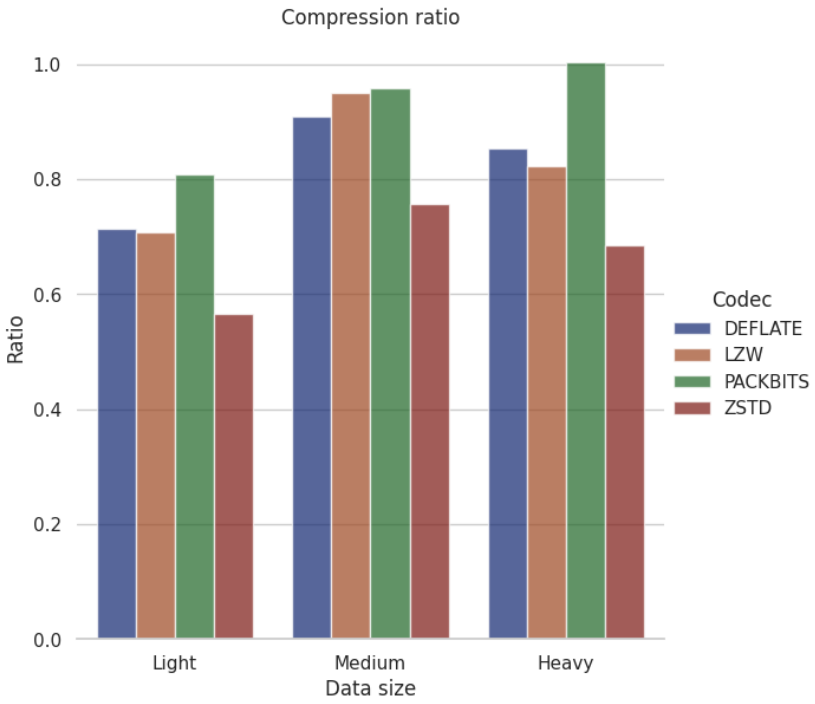

Fig. 2. Compression ratio of Deflate, LZW, Packbits, and Zstandard methods for light, medium, and heavy tiles.

high compression needs more CPU time to decompress into memory before processing. The majority of the time spent in computing NDVI is devoted to transferring satellite images over the network and loading them into memory, rather than performing calculations using the CPU. The comparison of the performances of Dask and Spark, considering different sizes of input data, compression methods, and 32 worker nodes is shown in Fig. 3.

The execution time of the COG tile compressed with the Packbits method and without compression is almost the same, as Packbits provides weak compression; thus, it uses little CPU time for decompression. The worst performance for both environments from the possible compression methods is Deflate, whereas the best one is Zstandard. Hence, the best compression method for satellite images in COG format is Zstandard, as it provides the highest compression ratio and optimal memory loading time. The performance improvement when using Zstandard compared to uncompressed mode is achieved by reducing network transfer time. Zstandard provides on average 2.15 and 1.82 times faster execution time compared with the uncompressed mode, approximately 4.72 and 3.99 times faster than the default selected Deflate method provided by global satellite image repositories correspondingly for Dask and Spark environments. Performance evaluation using Dask and Spark is quite interesting. For the default used Deflate compression method provided by EO repositories, Spark and Dask show similar execution times; however, Spark is a bit faster. The LZW compression method for the Dask environment is better than Deflate but worse than without compressing or compressing with Zstandard. Also, Spark does not support the compression method. With uncompressed data, Dask is faster than Spark for 16 GB input, whereas, in cases of 32 GB and 64 GB, Spark is faster. Performance in Dask using the Zstandard compression method is an optimal choice.

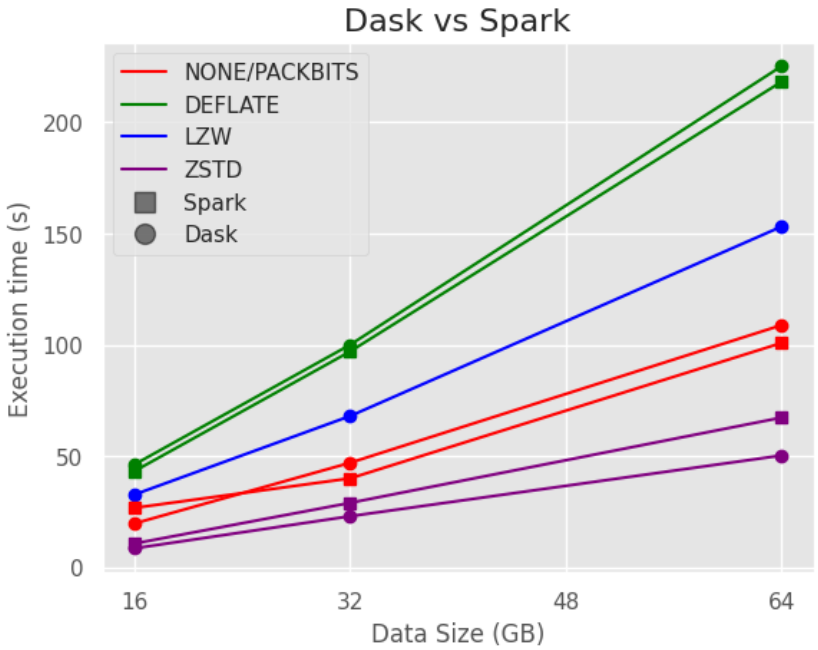

Fig. 3. Comparison of Dask and Spark considering 16, 32, 64 GBs of input data and compression methods.

4. Discussion

The study showed that various data compression methods could reduce storage requirements and network transfer time at different scales. Moreover, compressed data processing using multiple techniques in distributed environments such as Spark and Dask exhibited other execution times, with some compression methods outperforming uncompressed data processing time. The study aims to determine the optimal data compression method that balances performance and storage savings in the chosen distributed processing environments. The evaluation shows that the Dask and Zstandard combination is the best choice for the environment and compression method for EO satellite images. It provides the highest compression factor and performance compared to other supported compression methods.

The Armenian DataCube was initially set up with a 2-terabyte storage capacity, which is limited. To manage this, only the essential bands for specific EO applications that researchers are interested in during a particular period are downloaded and stored. If the storage capacity is exceeded, the options are to scale vertically or add external storage. The Zstandard compression technique was used in experiments to conserve 34 % of storage. This allows more data to be stored in the allocated DataCube space.

The Zstandard compression method combined with the Dask environment offers benefits such as improved data storage efficiency and EO data processing time. However, additional steps are required to achieve these benefits, such as converting analysis-ready data from the DataCube to Cloud Optimized GeoTIFF format and compressing them using the Zstandard method. Although this may increase the total execution time of downloading and preprocessing, it provides such benefits as enhanced processing time and storage savings. Moreover, this efficient method of storing compressed data can be applied to other types of EO data repositories and DataCubes.

In conclusion, data compression methods can effectively reduce the amount of EO data stored and improve processing performance. Zstandard exhibits the best performance and storage efficiency for EO data among the available compression methods. Additionally, the implementation of the Dask environment speeds up distributed processing.

5. Conclusion

The study evaluates the performance of EO data processing in Dask and Spark, considering compression methods. Experimental results show that Dask and Spark provide similar data processing performances. The mixture of the Dask and Zstandard compression methods is optimal, as the compression method provides the best compression factor of all possible lossless compression methods. It reduces the amount of used storage by 16 % and speeds up execution times by 4.72x and 3.99x in Dask and Spark, correspondingly compared with the Deflate method, which is used by default from the EO data repositories. In further work, it is planned to store the data in Armenian DataCube compressed with the Zstandard method and use the Dask environment for data processing.

References

[1] O. R. Young, M. Onoda. "Satellite Earth Observations in Environmental Problem-Solving", *In book: Satellite Earth Observations and Their Impact on Society and Policy*, pp. 3-27, 2017.

[2] D. A. Chu, Y. J. Kaufman, "Global monitoring of air pollution over land from the Earth Observing System-Terra Moderate Resolution Imaging Spectroradiometer (MODIS)", *Journal of Geophysical Research Atmospheres*, vol. 108, no. 21, November 2003.

[3] R.S. dos Santos, "Estimating spatio-temporal air temperature in London (UK) using machine learning and earth observation satellite data", *International Journal of Applied Earth Observation and Geoinformation*, vol. 88, June 2020.

[4] T. Krishnamurti and A. Chakraborty, "Impact of Arabian Sea pollution on the Bay of Bengal winter monsoon rains", *Journal of Geophysical Research*, vol. 114, March 2009.

[5] R. DeFries and F. Achard, "Earth observations for estimating greenhouse gas emissions from deforestation in developing countries", *Environmental Science & Policy*, vol. 10, no. 4, pp. 385–394, June 2007.

[6] Y. J. Kaufman and C. Ichoku, "Fire and smoke observed from the Earth Observing System MODIS instrument–products, validation, and operational use", *International Journal of Remote Sensing*, vol. 24, no. 8, pp. 1765–1781, November 2010.

[7] H. D. Guo and L. Zhang, "Earth observation big data for climate change research", *Advances in Climate Change Research*, vol. 6, no. 2, pp. 108–117, June 2015.

[8] A. Lewis, S. Oliver and L. Lymburner, "The Australian Geoscience Data Cube Foundations and lessons learned", *Remote Sensing of Environment*, vol. 202, pp. 276–292, 2017.

[9] Open data cube, [Online]. Available: https://www.opendatacube.org/

[10] S. Asmaryan and V. Muradyan, "Paving the Way towards an Armenian Data Cube", *Data*, vol. 4, no. 1, 2019.

[11] M. Drusch and U. D. Bello, "Sentinel-2: ESA's Optical High-Resolution Mission for GMES Operational Services", *Remote Sensing of Environment*, vol. 120, pp. 25–36, May 2012.

[12] M. Xiangrui, "Mllib: Machine learning in apache spark", *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1235–1241, 2016.

[13] R. Matthew, "Dask: Parallel computation with blocked algorithms and task scheduling", *Proceedings of the 14th python in science conference*, vol. 130, 2015.

[14] H. Astsatryan and A. Kocharyan, "Performance Optimization System for Hadoop and Spark Frameworks", *Cybernetics and Information Technologies*, vol. 20, no. 6, pp. 5–17, 2020.

[15] H. Astsatryan and A. Lalayan, "Performance-efficient Recommendation and Prediction Service for Big Data frameworks focusing on Data Compression and In-memory Data Storage Indicators", *Scalable Computing: Practice and Experience*, vol. 22, no. 4, pp. 401–412, 2021.

[16] Cloud Optimized GeoTIFF, [Online]. Available: https://www.cogeo.org/

[17] J. Li, "Parallel netCDF: A High-Performance Scientific I/O Interface", *Proceedings of the 2003 ACM/IEEE Conference on Supercomputing*, 2003.

[18] D. Mathieu and H. Sasson, "A Performance Comparison of Dask and Apache Spark for Data-Intensive Neuroimaging Pipelines", *2019 IEEE/ACM Workflows in Support of Large-Scale Science (WORKS)*, pp. 40–49, 2019.

[19] P. Mehta and S. Dorkenwald, "Comparative evaluation of big-data systems on scientific image analytics workloads", *Proceedings of the VLDB Endowment*, vol. 10, no. 11, pp. 1226-1237, 2017.

[20] I. Paraskevakos and A. Luckow, "Task-parallel Analysis of Molecular Dynamics Trajectories", *ICPP 2018: Proceedings of the 47th International Conference on Parallel Processing*, no. 49, pp. 1-10, 2018.

[21] Y. Shoukourian and V. Sahakyan, "E-Infrastructures in Armenia: Virtual research environments", *Ninth International Conference on Computer Science and Information Technologies Revised Selected Papers*, pp. 1-7, 2013.

[22] B. M. Randles and I. V. Pasquetto, "Using the Jupyter Notebook as a Tool for Open Science: An Empirical Study", *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, pp. 1-2, 2017.

[23] Armenian DataCube, [Online]. Available: http://datacube.sci.am/

[24] M. A. Wulder and T. R. Loveland, "Current status of Landsat program, science, and applications", *Remote Sensing of Environment*, vol. 225, pp. 127-147, 2019.

[25] Sentinel-2 Cloud-Optimized GeoTIFFs, [Online]. Available: https://registry.opendata.aws/sentinel-2-l2a-cogs

[26] G. Jifu and C. Huang, "A Scalable Computing Resources System for Remote Sensing Big Data Processing Using GeoPySpark Based on Spark on K8s", *Remote Sensing*, vol. 14, no. 3, 2022.

[27] N. Pettorelli, J. O. Vik, "Using the satellite-derived NDVI to assess ecological responses to environmental change", *Trends in Ecology & Evolution*, vol. 20, no. 9, pp. 503–510, 2005.

[28] S. Oswal, A. Singh, "Deflate compression algorithm", *International Journal of Engineering Research and General Science*, vol. 4, no. 1, 2016.

[29] M. J. Knieser, F. G. Wolff, "A technique for high ratio LZW compression [logic test vector compression", *Automation and Test in Europe Conference and Exhibition*, pp. 116–121, 2003.

[30] G. Feng, C. A. Bouman, "Efficient document rendering with enhanced run length encoding", *Color Imaging XI: Processing, Hardcopy, and Applications*, January 2006.

[31] Y. Collet, M. Kucherawy, "Zstandard Compression and the 'application/zstd' Media Type", *RFC Editor, USA*, February 2021.

Dask-ի և Spark-ի կատարողականի վերլուծություն՝ հաշվի առնելով տվյալների սեղմումը Երկրի դիտարկման տվյալների մշակման համար

Արթուր Գ. Լալայան

ՀՀ ԳԱԱ Ինֆորմատիկայի և ավտոմատացման պրոբլեմների ինստիտուտ, Երևան, Հայաստան
Հայաստանի ազգային պոլիտեխնիկական համալսարան, Երևան, Հայաստան
e-mail: arthurlalayan97@gmail.com

Ամփոփում

Բարձր կատարողական հաշվարկը լավ ընտրություն է երկրի դիտարկման մեծ տվյալների մշակման համար, ինչը թույլ է տալիս տվյալների մշակումը բաշխված և բարձր արդյունավետությամբ՝ օգտագործելով հիշողության մեջ հաշվողական հարթակներ: Տվյալների սեղմման տեխնոլոգիան նվազեցնում է պահանջվող պահեստավորման ծավալը և ցանցի փոխանցման ժամանակը, ինչպես նաև բարելավում է տվյալների մշակման ժամանակը: Հոդվածի նպատակն է ուսումնասիրել լայնորեն օգտագործվող տվյալների մշակման շրջանակների արդյունավետությունը՝ տվյալների անկորուստ սեղմման տեխնիկայի հետ համատեղ, Երկրի դիտարկման հատուկ աշխատանքային հոսքերի համար սեղմման օպտիմալ մեթոդ և մշակման շրջանակ գտնելու համար: Բուսականության նորմալացված տարբերության ինդեքսը գնահատվել է Հայաստանի տարածքի համար՝ օգտագործելով Sentinel արբանյակի տվյալները և հաշվի առնելով սեղմման աջակցվող մեթոդները հիշողության մեջ Dask և Spark շրջանակների աշխատանքի համեմատման համար: Փորձերը ցույց են տալիս, որ Zstandard սեղմման մեթոդը և Dask միջավայրը լավագույն ընտրություններն են նման աշխատանքային հոսքերի համար:

Բանալի բառեր՝ Երկրի դիտարկում, HPC, Spark, Dask, բաշխված հաշվարկ, տվյալների սեղմում:

Анализ производительности Dask и Spark для обработки данных наблюдения Земли с учетом сжатия данных

Артур Г. Лалаян

Институт проблем информатики и автоматизации НАН РА, Ереван, Армения
Национальный политехнический университет Армении, Ереван, Армения
e-mail: arthurlalayan97@gmail.com

Аннотация

Высокопроизводительные вычисления являются хорошим выбором для обработки больших данных наблюдения Земли, позволяя обрабатывать данные распределенным и высокопроизводительным способом с использованием вычислительных платформ в памяти. Технология сжатия данных сокращает объем хранилища и время передачи по сети и повышает производительность обработки. Целью статьи является исследование эффективности широко используемых систем распределенной обработки данных в сочетании с методами сжатия данных без потерь, чтобы найти оптимальный метод сжатия и структуру обработки для конкретных рабочих процессов наблюдения Земли. Нормализованный разностный индекс растительности был оценен для территории Армении с использованием данных со спутника Sentinel и с учетом поддерживаемых методов сжатия для сравнения производительности фреймворков Dask и Spark в памяти. Эксперименты показывают, что метод сжатия Zstandard и фреймворк Dask являются наилучшим выбором для таких рабочих процессов.

Ключевые слова: Наблюдение Земли, HPC, Spark, Dask, распределенные вычисления, сжатие данных.

Mathematical Problems of Computer Science 59, 45–56, 2023.
doi: 10.51408/1963-0101

UDC 004.891.3

Expert Knowledge-Based RGT Solvers for Software Testing

Mane P. Buniatyan[1], Sedrak V. Grigoryan[2] and Emma H. Danielyan[3]

[1]Synopsys Armenia, Yerevan, Armenia
[2]Institute for Informatics and Automation Problems of NAS RA,Yerevan, Armenia
[3]EPAM Systems Inc., Yerevan, Armenia
e-mail: buniatyanmane@gmail.com, addressforsd@gmail.com, emma_danielyan@yahoo.com

Abstract

Program testing is a way of assessing the quality of software and reducing the risk of software failure in operation [1]. Quality issues can cause as financial loss as well as harm to human lives (e.g., when the bug is in medical instruments, cars, etc.). So, it is very hard to underestimate the importance of testing.

There are multiple testing techniques, which are split into 3 major categories. One of them includes experience-based techniques. Test cases and scenarios used in experience-based testing are derived from the tester's knowledge and intuition, as well as their experience with similar applications and technologies. These techniques can be helpful in identifying tests that are not identified easily by other more systematic techniques. Depending on the tester's approach and experience, experience-based techniques may achieve widely varying degrees of coverage and effectiveness [1].

We propose a method for automation of experience-based testing via a class of combinatorial problems (RGT class). A Solver is developed for the class. It acquires expert knowledge and elaborates effective strategies for RGT problems [2]. The proposed method generates test cases dynamically based on the response of the program. The adequacy of the method is being experimented for "blender" open-source application, which has Python API allowing to experiment with testing and analyze test results.

Keywords: RGT class, RGT Solver, Software testing, Expert systems.
Article info: Received 25 September 2022; sent for review 11 October 2022; accepted 07 February 2023.
Acknowledgement: The authors express their deep gratitude to Dr. Edward Pogossian for his contribution and constructive comments to the work.

1. Introduction

Software Testing is an approach to assess the quality of software and to reduce the risk of its failure in operation [1].

In [1], testing techniques are divided into 3 groups: black-box, white-box and experience-based techniques. In the case of the last one, test cases are based on the testers' knowledge and intuition, on experience with similar applications and technologies. These techniques are efficient in identifying tests that are not identified easily by other more systematic techniques as well as when there is a limited testing time or incomplete specifications [1].

According to the World Quality Report 2021-2022 [3], one of the current trends in quality assurance and software testing is test automation. Test automation has the following benefits [1]:

- saves time by reducing repetitive manual work
- provides greater consistency and repeatability
- allows to evaluate the situation more objectively based on static measures, coverage reports, etc.
- provides more accurate information about the current state of testing based on gathered statistics, test progress, defect rates and performance.

There is a way to automate test case generation, known as the model-based testing (MBT). MBT is a technique for generating a test suite from requirements [4]. Instead of individual tests creation, testers create models that allow generating test cases automatically. These methods can be used in regression testing and are especially useful when the system changes frequently. In this case, the test suite can be regenerated easily by adjusting the model instead of readjusting each test case separately.

MBT has three important components [4]:

- a model (requirement, information, workflow, architectural, behavioral, configuration, deployment, performance, risk, environment, and usage models [5])
- a test-generation algorithm
- tools generating a supporting infrastructure (including the expected output).

MBT tools are meant to generate test suites by manipulating either with input data or behavior without handling both simultaneously. Generated test cases do not provide ways to test the system dynamically (the choice of modules to testing depends on the previous test results).

Software Testing can be considered as a combinatorial problem between a tester and states of a program. Hence, testing can be also considered as a representative of Reproducible Game Tree (RGT) class problems. RGT is a class of combinatorial problems, for which the space of solutions is a reproducible game tree. These problems meet the following requirements [6]:

- there are interacting actors (players, competitors, etc.) performing identified types of actions in specified moments of time and specified types of situations
- there are identified benefits for each actor
- there are descriptions of situations in which actors act in and are transformed after actions.

For such problems with a given arbitrary situation x and an actor A, who is going to act in x, we can generate a corresponding game tree GT(x, A) comprising all the games started from x. Games represent all possible sequences of legal actions for players and situations that they can create from the given initial, or the root situation x. In our consideration, the games are finite and end with one of the goal situations of the problem [6].

Assuming that A plays according to a deterministic program, a strategy, the GT(x, A) represents, in fact, all possible performance trees of the strategies from x. In that sense, the GT(x, A) determines the space of all possible solutions from the situation x. With the given criterion K to evaluate the quality of strategies, we can define the best strategy S*(x, A) and the corresponding best action of A from x [6].

RGT class includes important problems like computer networks intrusion protection, optimal management and marketing strategy elaboration in competitive environments, testing of programs, defense of military units from various types of attacks, communication problems, certain types of teaching, chess and chess-like games [2].

One of the advantages of RGT class is that these problems are reducible to the standard kernel problems K. K- methodology multiplies the achievements for particular problems of this class. Distributed development of this methodology is possible. K-methodology enhances the effectiveness of RGT Solvers providing answers to urgent RGT questions including the following ones [2]:

- measurement of the effectiveness of Solvers
- analysis and typification of combating knowledge
- construction of knowledge-based Solvers
- regular acquisition of RGT expert knowledge and enhancing the effectiveness of Solvers.

The validity of K-methodology was proved for certain RGT problems including Chess, Network Intrusion Protection, Navy Defense from Attacks, Management, Marketing etc. [2].

RGT Solver is a software that acquires expert knowledge and elaborates effective strategies for RGT problems [2]. It is a universal tool for solving RGT-class problems.

Strategy searching and game tree. As already mentioned, the space of solutions for RGT problems is a reproducible game tree, and with the given criteria, we can evaluate and choose the best possible actions in given situations for the given actor.

As the combinatorial complexity of the mentioned problems is huge, we need to reduce the game tree. Otherwise, the computer's computational resources (memory and storage) will not be enough to solve them. C. Shannon suggested reducing the tree by building it until the resources are expired. It is not an effective way because we waste our resources to compute steps that will not improve the current situation. Another approach, suggested by M. Botvinnik, is to consider only those steps that have potential benefit in the current case, i.e., we should not examine the steps that have no meaning. We can evaluate the possible usefulness of an action with the knowledge (without reviewing the opponent's answers) and choose the most profitable one. Then, by checking the opponent's potential actions, we can build the game tree and choose the best move in a given situation [7].

The Solver builds the game tree, evaluates situations with the knowledge, then chooses the best action using the minimax algorithm.

The purpose of this paper: Testing of programs can be considered as an RGT problem, and RGT Solver can be used for experience-based testing as an expert system when the corresponding knowledge is available.

In this work, we aim to provide a definition of testing problems as RGT problems, a way of formulating knowledge, and an approach for proper assessment of tested programs, which also covers the drawbacks of model-based testing approaches (in particular, combining different behaviors and input data, running both functional and non-functional tests at the same time, and generating tests dynamically). Thus, the following open questions are addressed:

1. What kind of knowledge are we going to use, who are the actors as well as what are their possible actions?

2. How to evaluate each situation, what kind of goals each actor has, etc.?

Overall, this leads to proposing a model for representing an experience-based testing as an RGT problem.

2. Reduction of Program Testing to RGT Class

In RGT problems, it is essential to define the situations, the actors, the actions, and benefits for each of them. Let's define these terms for program testing.

The actors in software testing are the system under test (i.e., the program) and the tester. Note, that unlike some other problems in the RGT class (e.g., like chess), where the opponent tries to make counteraction, in testing the program just responds to the tester's actions.

The actions are any valid elementary operations that can be performed with the program. While building the "game" tree, the Solver dynamically combines these actions, creates test cases and executes them depending on the response of the program. Note, that not all combinations of the elementary operations are meaningful from the perspective of the user (e.g., actions that have no effect or are not connected with each other). That is why we need to find a way to control these combinations. The actions of the program are actually only responses to the tester's actions.

The situations are the current states of the program. We can estimate the current situations with [0;1] numbers, where 0 means that no bugs are found, 1- that the program is in a critical state and is not usable. The numbers in-between 0 and 1 are intermediate values, and situations with values closer to 1 are worse than situations with values closer to 0. We suggest the following criteria for evaluating the current state of programs (these criteria can be expanded later):

- Existence of bugs (difference between expected and observed results): different bugs have different importance; when the main functionalities of the program do not work as expected, the program becomes useless (e.g., if the user is not able to log into a social network, save the result of the accomplished job, do a transfer in the banking system, etc.).

- Performance degradation: we all would like to have fast, high performing programs, but unfortunately it is not always possible. Performance degradation in a part of the program that is used frequently will cause to slowdown the work, but if it is in a part

that can be done without human interaction and/or is performing rarely, then it can be acceptable.

- Security: this is essential for some programs (e.g., banking system, strategic information storing, transfers, etc.).
- Crashes and hangovers: this is always bad, and in some cases, they can even cause to a fatal problem, like losing the whole work performed. In most situations, this is not acceptable.

We need to take into account the number of problems, as well as their severity and importance, the sequence of actions causing the problem (i.e., how frequently the problem occurs in "real life"). A bug in a very important functionality is worse than a crash that users might not even encounter, but, on the other hand, having lots of "minor" issues in the program is also not acceptable. When one of the main functionalities does not meet the requirements mentioned above, the program is in a critical state, and it cannot be delivered to customers. The importance of each functionality is considered as a multiplier for the appropriate criterion.

The current state of the program can be measured with the following evaluation function:

$$st = mc * c + mb * b + mp * p + ms * s, \tag{1}$$

where $mc, mb, mp, ms \in [0; 1]$, c, b, p, $s = \{0 \mid 1\}$. C, b, p and s are Boolean variables, that show the existence of crashes/hangovers, bugs, performance degradations or security problems respectively (1 if the mentioned problems occurred, otherwise - 0). Mc, mb, mp and ms are multipliers for the occurred problems (they show the importance of the broken functionality). Any occurred problem is counted only once, so if, for example, a crash occurs, even if it relates to a security problem or it is a bug (obviously, it is not an expected result) we will consider c = 1, b = 0, s = 0 and p = 0. If the current state of the program is bigger than 1, we consider it as 1.

3. RGT Expert Knowledge Formatting for Testing

Error guessing, exploratory testing and checklist-based testing are representatives of experience-based techniques [1].

Considering the characteristics of each of these techniques, we propose the following usage of the Solver: by reviewing issues occurred before, the usage of the program and its main functionalities, we create checklists. In the Solver, checklists are represented as plans, and the checklists' actions as goals. Based on the coverage reports, the source files responsible for each action can be defined. These connections help to prioritize the created checklists. The user can also define priorities depending on the module he/she is most interested in.

Checklists lead to the creation of a game tree. Each branch in the tree is a test case. It is important to mention that actions in the checklists are general, i.e., many elementary actions can correspond to one action in the checklist. It allows you to combine multiple actions and build a tree. Checklists define if it still needs to proceed to the next steps or not in case of a defect occurrences in the current step.

Multipliers in formula (1) are also given as knowledge for the Solver. They show the importance of user action. Note, that multipliers should be defined for both elementary

and checklist actions. The same elementary action in different situations can have different importance, e.g., if the user tries to save a text file it is more important to save the text than the style. We multiply both multipliers to get one for the action. Imagine that in the example below, mb = 0.8 for the elementary action "save" and for the following checklists of actions "open the program, add text, save", "open an existing text file, change the style, save". Let's say we have mb = 1 for the "save" in the checklist1 and mb = 0.6 for the "save" in the checklist2. In this case, if the program is not able to save the text, we will have mb=1*0.8=0.8 and for the second case: mb=0.6*0.8 = 0.48. Thus, the first case will be considered worse than the second one.

In the case of performance degradation, we need to multiply mp with the coefficient showing how many times the performance was slowed down or how much longer it takes to perform the same action. E.g., if the performance is 2x slower than expected, we need to multiply mp with 2.

The testing continues until a. the given time is expired, b. all/chosen checklists are checked or c. if the program gets into a critical state.

4. RGT Solver Experiments in Program Testing

We have chosen the Blender program as a system under test. It is an open-source 3D modeling program with a Python interface that can be used for testing. In order to understand how the program testing Solver works and how the knowledge and checklists can be represented, let's study an example.

To understand how the knowledge and checklists can be represented, let us review an example.

The checklist below checks some of the main functionalities of the program:

```
1 # basic_operations; x.cpp
2
3 Open the program; mc=1, mp/5s/=0.04;
4 Move 3d cursor; mb=0.7, mc=0.9, mp=0.06, nextStep=1;
5 Add object; mb=0.9, mc=0.9, mp=0.07;
6 Change geometry; mb=0.8, mc=0.9, mp=0.07;
7 Transform object; mb=0.8; mc=0.9, mp=0.07
```

Fig. 1. Checklist Example.

To keep it simple, we just added a few basic operations, but this list can be enlarged if needed. The operations in this checklist can be independent, like lines 6 and 7. But if this was a checklist based on the previous failures or a user story, then all steps would depend on each other. This checklist could be used if we had limited testing time and could only check the main operations to make sure that there were no critical issues (like a smoke test). The first line of the checklist (i.e., the comment) represents the name of the checklist and the source file which is associated with the checklist (here, as we don't know the corresponding source file, we put x.cpp just to show the structure of the checklist. The source file is not

mandatory). If some multipliers are absent in the checklist, we assign 0 to them (e.g., ms=0 for all actions in checklist below, because they could not lead to security problems). The variable nextStep is used to determine whether the next step should be performed or not in case of bug in the current step (e.g., if the user is not able to move the 3D cursor it is still somewhere in the scene and the user can add objects). In line 3 we open the program. If it crashes it is a critical state for the program, thus mc=1.

Next to mp there is the expected time the operation should take (mp/5s/). If it takes 25 seconds, we multiply mp by 5. As this operation is not repeatable and happens only once, when the work starts, its performance is not very important, but yet the user cannot wait for about 10 minutes to start working. As the performance depends on the users' computer, the performance parameters are defined for minimum system requirements of the program. In the example above, we just used values based on local resources.

In line 4, we need to move the 3D cursor. 3D cursor position defines where the object is being added. It can also be used as a 3D view orientation to define where to move objects, to move the pivot point to the 3D cursor, as the rotation point in the spin tool, etc. So, it is a quite important feature, but in case it does not work users can still find workarounds. Note that there is no expected time next to mp for this action. It is because this action should work simultaneously with the click (i.e., should not take noticeable time). Like other actions in the checklist, this is one of the basic operations, so crash is unacceptable here, thus mc = 0.9. Note that all the multipliers here are conditional and this is just an example. In real world example, probably, multipliers should be chosen more thoroughly. nextStep is 1 here, because even if the 3D cursor cannot be moved, we are still able to add an object. To perform this step using the Python API we do the following:

```
1  # Move 3d cursor, m=1
2
3  import random
4  import bpy # python module for blender
5  x = random.randint(0, 100)
6  y = random.randint(0, 100)
7  z = random.randint(0, 100)
8
9  bpy.context.scene.cursor.location = (x, y, z)
10
11 assertEqual(bpy.context.scene.cursor.location.x, x)
12 assertEqual(bpy.context.scene.cursor.location.y, y)
13 assertEqual(bpy.context.scene.cursor.location.z, z)
```

Fig. 2. Elementary Operation: Move 3D Cursor

This is an elementary operation for moving the 3D cursor. The first line comment shows the corresponding general operation (in the checklist) and the multiplier. As in this case

only 1 elementary operation corresponds to the checklist operation, its multiplier is 1. Note that the case is not always the same (the coordinates are randomly generated) and the test also checks if the operation was performed successfully or not.

In the 5th line of the checklist, we have the "Add object" operation. Many elementary operations correspond to this operation (see Fig. 3): there are lots of groups of objects, and each group itself contains various objects.

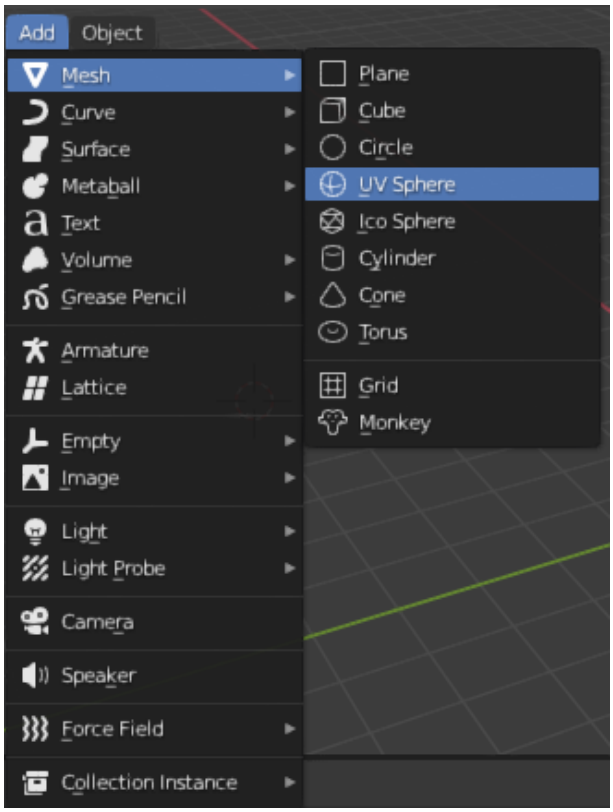

Fig. 3. Add Object.

The Python code below is an example of the "Add object" operation. It adds a cube in the current location of the cursor. As all objects can be used for creating different 3D models, and their importance is dependent on what exactly the user tries to create m=1 for all objects. Note that if the object is not added then we cannot perform the next action, i.e., we cannot change its geometry.

The last command in the checklist is "Change geometry". First of all, the user should switch to the edit mode in order to change the object's geometry, i.e., move the object's vertices, edges and faces, and then perform the corresponding operations. For this general action, there are 3 possible elementary actions (move vertices, edges, faces). All of them are important while creating a 3D model, but considering the fact that if a user is not able to move the edge, he/she can choose vertices of the edge and move them together (so that the edge will be moved), or choose all edges/vertices of a face and move it. The most important one in those operations is moving vertices, and then edges, then surfaces.

For the given example, the Solver moves the 3D cursor to different positions, adds different objects, changes their geometry, and makes sure that these operations work as expected for different objects (i.e., checks that the Python tests are passing). To check how the Solver

```
# Add object, cube, m=1
import bpy

initial_count = len(bpy.context.scene.objects)
c1 = bpy.ops.mesh.primitive_cube_add(enter_editmode=False,
align='WORLD',
location=bpy.context.scene.cursor.location, scale=(1, 1, 1))

count_after_add = len(bpy.context.scene.objects)
assertEqual(initial_count, (count_after_add - 1))
```

Fig. 4. Elementary Operation: Add Object.

Thus, in this case, the multiplier for each operation will be different:

```
# Change geometry, vertices, m=0.9
# . . .
```

```
# Change geometry, edges, m=0.8
# . . .
```

```
# Change geometry, faces, m=0.7
# . . .
```

Fig. 5. Elementary Operation: Change Geometry.

behaves if the operation does not work, we can simply use assertNotEqual function instead of assertEqual (e.g., instead of "assertEqual(bpy.context.scene.cursor.location.x, x)" we can write "assertNotEqual(bpy.context.scene.cursor.location.x, x)"). The Solver will combine different elementary tests together, create test cases and run them.

To run tests, we use the following command:

```
ctest -R <test_name> -C Release --output-on-failure
```

Fig. 6. Command For Running a Test.

In order to use the Solver for different programs, we use a configuration file, which defines how to run tests (e.g., paths to test cases, checklists and elementary operations).

5. Conclusion

We propose a new approach for test automation and test results evaluation considering the testing of programs as a RGT-class problem. In this work:

1. tools defining the types of knowledge for testing the target application are described. The described knowledge is being integrated into RGT Solver and being used to run test cases, test scenarios with later evaluation of test results.
2. An approach for evaluating the state of the program during the testing is proposed.
3. The adequacy of the proposed approach is being experimented with the open-source Blender application.
4. The proposed approach solves drawbacks of the model-based testing approach, namely, allows to generate test cases dynamically.

The described solution is generic for the RGT Solver and can be used for testing various applications.

References

[1] K. Olsen and M. Posthuma and S. Ulrich, " Certified Tester Foundation Level Syllalbus", *International Software Testing Qualifications Board*, pp. 56–62, 2019.

[2] E. Pogossian, *Constructing Models of Being by Cognizing*. Yerevan, pp. 150–159, 2020.

[3] *World Quality Report*, Capgemini, Sogeti, Micro Focus, pp 16–37, 2021

[4] D. Rakhi, J. Ashish, N. Karunanithi, J. Leaton, C. Lott, G. Patton and B. Horowitz, "Model-based testing in practice", *Proceedings of the 1999 International Conference on Software Engineering (IEEE Cat. No.99CB37002)*, Los Angeles, CA, USA, 1999, pp. 285-294, doi: 10.1145/302405.302640.

[5] I. Schieferdecker and A. Hoffmann, *Model-Based Testing*, IEEE Software 29.1, pp. 14–18, 2012.

[6] E. Pogossian, V. Vahradyan A. Grigoryan, On competing agents consistent with expert knowledge, *Proceedings of Second International Workshop, AIS-ADM 2007,Autonomous Intelligent Systems: Multi-Agents and Data Mining*, St. Petersburg, Russia, pp. 229–241, 2007.

[7] M. Botvinnik, *Computers in Chess: Solving Inexact Search Problems*, Springer-Verlag, New York, 1983.

Փորձագիտական գիտելիքների վրա հիմնված RGT SOLVER-ի կիրառումը ծրագրային ապահովման թեստավորման խնդրում

Մանե Պ. Բունիաթյան[1], Սեդրակ Վ. Գրիգորյան [2] և Էմմա Հ. Դանիելյան[3]

[1]Synopsys Հայաստան, Երևան
[2]ՀՀ ԳԱԱ Ինֆորմատիկայի և ավտոմատացման պրոբլեմների ինստիտուտ, Երևան, Հայաստան
[3] EPAM Հայաստան, Երևան
e-mail: buniatyanmane@gmail.com, addressforsd@gmail.com, emma_danielyan@yahoo.com

Ամփոփում

Թեստավորումը ծրագրի որակը գնահատելու և շահագործման մեջ ծրագրային ապահովման ձախողման ռիսկերը նվազեցնելու միջոց է: Ծրագրում սխալների առկայությունը կարող է բերել ինչպես ֆինանսական կորուստների, այնպես էլ մարդկային զոհերի (օրինակ, բժշկական սարքավորումների կամ մեքենաներում առկա սխալները): Այսպիսով, բարդ է թերագնահատել թեստավորման կարևորությունը: Թեստավորման մոտեցումները կարելի է բաժանել 3 հիմնական խմբերի, որոնցից մեկը փորձի վրա հիմնված (experience-based) թեստավորումն է: Այս պարագայում թեստերը ստեղծվում են՝ հիմնվելով թեստավորողի գիտելիքների և ինտուիցիայի, ինչպես նաև նախկինում նմանատիպ ծրագրերի հետ ունեցած փորձի վրա: Փորձի վրա հիմնված մոտեցումներն օգնում են բացահայտել այնպիսի սխալներ, որոնք շատ բարդ է հայտնաբերել ավելի համակարգված մոտեցումներով: Այս աշխատանքում մենք առաջարկում ենք փորձի վրա հիմնված թեստավորման ավտոմատացում՝ օգտագործելով կոմբինատոր խնդիրների RGT դասը: RGT դասի խնդիրների լուծման համար մշակվում էRGT Solver-ը՝ ծրագրային փաթեթ, որը կուտակում է փորձագիտական գիտելիքներ և ստեղծում է արդյունավետ ռազմավարություններ RGT դասի խնդիրների լուծման համար: Առաջարկում ենք RGT Solver-ն օգտագործել ծրագրերի թեստավորման խնդրում: Solver-ը ստեղծում է թեստային իրավիճակներ՝ կախված ծրագրի արձագանքից/պատասխանից և գնահատում է դրանք ըստ նախապես սահմանված չափանիշների: Այս մոտեցման աղեկվատությունը փորձարկվում է եռաչափ մոդելավորման “Blender” ծրագրի միջոցով:

Բանալի բառեր՝ RGT դաս, RGT Solver, ծրագրային ապահովման թեստավորում, փորձագիտական համակարգեր:

RGT SOLVER на основе экспертных знаний для тестирования программного обеспечения

Мане П. Буниатян[1], Седрак В. Григорян[2] и Емма Г. Даниелян[3]

[1]Synopsys Армения, Ереван
[2]Институт проблем информатики и автоматизации НАН РА, Ереван, Армения
[3]EPAM Армения, Ереван
e-mail: buniatyanmane@gmail.com, addressforsd@gmail.com, emma_danielyan@yahoo.com

Аннотация

Тестирование программ-это способ оценки качества программного обеспечения и снижения риска отказа программного обеспечения в работе. Очень трудно недооценить важность тестирования: проблемы с качеством программ могут привести как к финансовым потерям, так и нанести ущерб здоровью людей (например, когда ошибка находится в медицинских приборах, автомобилях и т. д.).

Методы тестирования можно подразделить на 3 основные группы. Одна из них - это методы, основанные на опыте. Здесь тестовые примеры создаются на основе знаний и интуиции тестировщика, а также на его опыте работы с аналогичными приложениями и технологиями. Эти методы могут быть полезны при определении тестов, которые не легко идентифицировать другими более систематическими подходами к тестированию. В зависимости от подхода и опыта тестировщика, эти методы могут обеспечивать широкую степень покрытия и эффективность тестирования. В данной статье мы предлагаем метод тестирования на основе опыта (автоматизация тестирования) через класс комбинаторных задач (RGT класс). RGT класс включает такие важные задачи, как защита от вторжений в компьютерные сети, разработка оптимальной стратегии управления и маркетинга в конкурентной среде, тестирование программ, защита воинских частей от различных типов атак, проблемы со связью, отдельные виды обучения, шахматы и шахматоподобные игры. RGT Solver - это программа, которая накапливает экспертные знания и разрабатывает эффективные стратегии для решения задач класса RGT. В качестве экспертной системы для тестирования, основанного на опыте, предлагается использовать RGT Solver. Solver генерирует тестовые ситуации на основе ответа/реакции программы и оценивает их по ряду заранее определенных критериев. Адекватность метода показана на примере приложения с открытым исходным кодом "Блендер".

Ключевые слова: RGT класс, RGT Solver, тестирование программного обеспечения, знания, экспертные системы.

Mathematical Problems of Computer Science 59, 57–68, 2023.
doi: 10.51408/1963-0102

UDC 004.934

Making Speaker Diarization System Noise Tolerant

Davit S. Karamyan[1,2], Grigor A. Kirakosyan[2,3] and Saten A. Harutyunyan[2]

[1]Russian-Armenian University, Yerevan, Armenia
[2]Krisp.ai, Yerevan
[3]Institute of Mathematics of NAS RA, Yerevan, Armenia
e-mail: {dkaramyan, gkirakosyan, sharutyunyan }@krisp.ai

Abstract

The goal of speaker diarization is to identify and separate different speakers in a multi-speaker audio recording. However, noise in the recording can interfere with the accuracy of these systems. In this paper, we explore methods such as multi-condition training, consistency regularization, and teacher-student techniques to improve the resilience of speaker embedding extractors to noise. We test the effectiveness of these methods on speaker verification and speaker diarization tasks and demonstrate that they lead to improved performance in the presence of noise and reverberation. To test the speaker verification and diarization system under noisy and reverberant conditions, we created augmented versions of the VoxCeleb1 cleaned test and Voxconverse dev datasets by adding noise and echo with different SNR values. Our results show that, on average, we can achieve a 19.1% relative improvement in speaker recognition using the teacher-student method and a 17% relative improvement in speaker diarization using consistency regularization compared to a multi-condition trained baseline.
Keywords: Speaker recognition, Speaker diarization, Noise robustness, Teacher-student, Consistency regularization.
Article info: Received 9 January 2023; send to review 30 January 2023, received in revised form 11 April 2023; accepted 17 April 2023.
Acknowledgement: This research was supported by Krisp.ai.

1. Introduction and Related Work

Speaker recognition (SR) is a broad field of study that addresses two major tasks: speaker identification and speaker verification. Speaker identification is the task of identifying a person, whereas speaker verification is the task of determining whether the speaker is who they claim to be. In this study, we focus on far-field, text-independent speaker recognition, where the speaker's identity is determined by the speaking style rather than the content of the speech. Typically, such speaker recognition systems operate on unconstrained speech utterances that are converted into a fixed-length vector known as speaker embedding. Many speech0-processing tasks use speaker embedding such as speaker diarization (SD) [1, 2], automatic speech recognition (ASR) [3], and speech synthesis [4, 5].

In recent years, deep neural networks have actively been employed for speaker embedding extractors since d-vector [6] was proposed. Subsequently, the x-vector [7] was widely used because of the superior performance achieved by employing statistical pooling and time delay neural network (TDNN). Other architectures such as ResNet-based convolutional neural networks and CNNs with cross-convolutional layers [8, 9] were employed for capturing the traits of speech. In addition, to deal with variable-length inputs, Transformer [10], CNN-LSTM [11] and a slew of variants of TDNN [12] were applied for DNN-based speaker embedding extractors. Finally, to reduce the computational complexity and make the models smaller, [13, 14] employed 1D depth-wise separable convolutions for the speaker recognition task.

Metric learning techniques have been successful in speaker recognition tasks. These methods aim to create speaker embeddings with small distances between embeddings of the same speaker and large distances between embeddings of different speakers since unsupervised clustering will be applied to embeddings later in the speaker diarization pipeline. The triplet loss was proposed in [15] which required a careful selection of a triplet because the effectiveness of the performance depended on the contrast between negative and query samples. The prototypical loss was proposed in [16], where many negative samples were used and the Euclidean distance between the centroid of all negative samples and the query embedding was maximized. In the generalized end-to-end loss [17], every utterance in the mini-batch functions as a query as opposed to just one in the prototypical loss. The angular prototypical (AP) loss [18] used only one utterance from each class as the query like the prototypical loss, but with a cosine similarity-based metric.

The primary use case for speaker embeddings is speaker diarization. Speaker diarization is the process of dividing an input audio stream into homogeneous segments according to the speaker's identity. A typical speaker diarization system usually consists of several steps: (1) Speech segmentation, where the input audio is segmented into short sections that are assumed to have a single speaker, and the non-speech sections are filtered out by Voice Activity Detection (VAD), (2) Speaker embedding extractor, where speaker embeddings are extracted from segmented sections, (3) Clustering, where the extracted audio embeddings are grouped [1] into clusters based on the number of speakers present in the audio recording, and optionally, (4) Resegmentation step is performed to further refine clustering results.

In real-world environment, noise causes significant degradations to the performance of speaker diarization systems, and is, hence, a major problem requiring special attention. The goal of noise-tolerant speaker diarization is to achieve improved performance in noisy environments. A recent work [19] tackles this problem using the auto-encoder architecture as a dimensionality reduction module. They extract two low-dimensional codes from speaker embeddings, representing the speaker identity and irrelevant noise information, then remove the noise factors. To our knowledge, there hasn't been a lot of research done in this particular area. ASR systems also suffer deterioration due to audio noise, and this has been the subject of extensive research [20, 21, 22], some of which inspired us.

In this paper, we explore several approaches, borrowed from unsupervised domain adaptation, to make the speaker recognition models noise tolerant. In particular, we apply teacher-student and consistency regularization techniques on speaker recognition and diarization tasks and compare them with multi-condition training when various noise augmentations are used.

We were inspired by the significant results of this work for teacher-student [22], where clean and noisy audios are fed to the teacher and the student, respectively, to enforce similarity between the output distributions. Consistency regularization is a commonly-used

technique amongst a variety of tasks in machine learning. This work [20] applies it in a manner similar to that mentioned previously, only here clean and noisy inputs are both fed to the student model. In the paragraphs that follow, we'll discuss in detail how we apply these concepts to obtain noise-robust speaker recognition and diarization.

2. Improving Noise Robustness of Speaker Diarization System

There are several ways to improve the performance of speaker diarization systems in noisy and reverberant environments. For instance, work in [1] proposed the sequence of refinement operations to smooth and denoise data in the similarity space. In this work, we will focus only on the speaker embedding extraction part, and we are going to use unsupervised domain adaptation techniques to make the model noise tolerant.

Given a training dataset consisting of pairs (x_i, y_i) where x_i represents an audio signal and y_i represents the speaker id. Our goal is to learn a parametrized function f_θ, which should be able to compress any given audio into a d-dimensional vector, also known as a speaker embedding. Moreover, if two audio signals are spoken by the same speaker, then the cosine similarity between their corresponding embeddings should be higher. Conversely, if the two audios are spoken by different speakers, the cosine similarity between their embeddings should be lower. The additive angular margin (AAM) loss, as proposed in [23], is a prevalent method for training speaker embedding extractors. The aim of the AAM loss is to minimize the angle between speaker embeddings belonging to the same speaker while simultaneously maximizing the angle between speaker embeddings belonging to different speakers.

2.1 Consistency Regularization

The core idea behind consistency regularization (CR) is to make sure that the network produces similar embeddings for the augmented versions of the same unlabeled utterance [20, 24, 25]. It is enforced by an additional regularization term in the loss function:

$$L_{CR} = \frac{1}{N}\sum_{i=1}^{N} |f_\theta(A(x_i)) - f_\theta(A(x_i))|_2^2,$$

where f_θ is an embedding extractor with parameters θ, N represents the total number of training examples within the dataset. By $A(x)$ we denote a stochastic operation that augments the audio in such a way that the speaker identity remains the same. So the difference is most likely non-zero. The final form of loss is a weighted combination of L_{AAM} and L_{CR} as shown below:

$$L = (1-\alpha)L_{AAM} + \alpha L_{CR},$$

where α is a hyperparameter taking values between 0 and 1.

2.2 Teacher-Student

One critical problem with L_{CR} loss is that it is not stable because of unstable target. To mitigate unstable target problem, the teacher-student model was proposed in [26], where two separate models were used: a Student network with θ parameters and a Teacher with

θ' parameters. On unlabeled examples, the Teacher network provides the learning target for the Student network:

$$L_{TS} = \frac{1}{N}\sum_{i=1}^{N} |f_{\theta}^{Student}(A(x_i)) - f_{\theta'}^{Teacher}(A(x_i))|_2^2.$$

Student is trained as usual. Teacher model is not trained via back-propagation. Instead, its weights are updated at each iteration using the weights from the Student network. Again, the final loss is a weighted combination of $\mathcal{L}_{\mathcal{AAM}}$ and L_{TS} as shown below:

$$L = (1-\alpha)L_{AAM} + \alpha L_{TS}.$$

2.3 Knowledge Distillation

If the teacher model is already trained, it is desirable that its weights remain constant. This training setup is known as "knowledge distillation", where the Student model is trained to mimic a pre-trained, larger model [27].

3. Experiments

3.1 Model Architecture

In all experiments, we will use the SpeakerNet [13] architecture as the backbone model. SpeakerNet models are made up of 1D Depth-wise separable convolutional layers. On top of the model, a statistical pooling layer is used to obtain a fixed-length vector. The proposed variation of SpeakerNet (SpeakerNet-M) has fewer parameters (5M) when compared to SOTA and shows very similar performance on VoxCeleb1 [28] trial files when compared to SOTA systems. The model provides embeddings of size 256 for a given audio sample.

In teacher-student experiments, both the teacher and the student have the same architecture.

3.2 Datasets

The VoxCeleb1 [28] and VoxCeleb2 [29] datasets are widely recognized benchmarks in the field of speaker recognition. These datasets have pre-defined development and test sets, which allow for an objective and consistent evaluation of speaker recognition models. We trained our speaker recognition models using only the development part, which consisted of 7205 distinct speakers.

For evaluation of speaker embeddings quality, we use VoxCeleb1 cleaned test trial file. The test trial file contains a list of audio pairs, and the model's performance is evaluated based on its ability to correctly determine whether the two recordings belong to the same speaker or not. To evaluate speaker diarization, we use the VoxConverse [30] development set. The dataset statistics are shown in Table 1.

3.3 Metrics

The equal error rate (EER) metric is used to evaluate the speaker verification. This is the rate used to determine the threshold value for a system when its false acceptance rate and

Table 1: Statistics of datasets used for training SpeakerNet.

Dataset	**# Speakers**	**Duration (h)**	**# Utterances**
VoxCeleb1	1211	340.4	148642
VoxCeleb2	5994	2359.77	1,092,009

false rejection rate are equal. We calculate EER on VoxCeleb1 cleaned test trial file under original, noisy and echo conditions.

For diarization evaluation purposes, we used diarization error rate (DER). This is the sum of three error terms: false alarm (FA), missed detection (MS) and speaker confusion error rate (CER). Similar to the previous works [12, 14], we use collar 0.25 sec and ignore overlap speech regions for confusion error rate calculation. We test the diarization system in original, noisy, and echo scenarios, just like we do for speaker verification.

Both EER and DER are calculated using the cosine similarity back-end.

3.4 Experiment Setup

3.4.1 Input Features

Our audio pre-processing procedure is identical to the one described in the SpeakerNet paper [13]. For each frame window of 20 ms, shifted by 10 ms, 64-dimensional acoustic features were calculated from the speech recordings. Each utterance fed to the encoder has a size $T \times 64$, where T is the number of frames in a given audio sample. We crop speech segments into random chunks from 3 to 8 seconds. With larger chunks, the model converges faster.

3.4.2 Clean Teacher

Our first baseline is a clean teacher trained on VoxCeleb1 and VoxCeleb2 datasets with additive angular margin loss. We set the AAM loss hyperparameters to $s = 30$ and $m = 0.2$, as it was shown in [13, 14], these values give the best results. To avoid overfitting, we added SpecAugment [31] to the training pipeline, which randomly masks blocks of frequency and time channels.

3.4.3 Noisy Teacher

Our second baseline is a noisy teacher trained with the same objective as a clean baseline, and with the additional augmentation steps described below:

- *No Augment*: Leave the utterance unchanged
- *RIR Augment*: Reverberate an input audio using an impulse response from RIRS dataset [32]
- *Noise Augment*: Add noise from MUSAN [33] dataset with signal-to-noise (SNR) values randomly chosen from 0-50DB
- *RIR-Noise Augment*: Apply noise and echo perturbations to the same audio at the same time

- *Speed Augment*: Speed perturbation with 0.95x and 1.05x speeds

RIR, *Noise*, and *RIR-Noise* augmentations all have a probability of 0.25 and are mutually exclusive. *Speed* augmentation is applied independently with a probability of 0.1.

3.4.4 Consistency Regularization

We add an extra mean squared loss between embeddings for the augmented and non-augmented versions of the same utterance to the AAM loss during training.

We set the α hyperparameter in the final loss to 0.1.

3.4.5 Teacher-Student

In order to supervise the student model, we choose our Clean-Teacher baseline as the teacher. We did not update teacher weights during the training and no perturbations were applied to the input of the teacher model. The flow chart of teacher-student training is presented in Fig. 1. During the training procedure, in addition to the AAM loss, the mean squared loss between the student and teacher-produced embeddings is minimized.

We set the α hyperparameter in the final loss to 0.1.

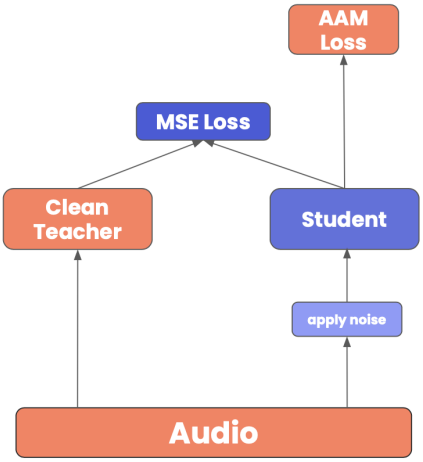

Fig.1. Flow chart of teacher-student learning for improving noise robustness of SR.

3.4.6 Optimization

All models are trained for 200 epochs with an SGD optimizer, with an initial learning rate (LR) of 0.08 using a cosine annealing LR scheduler on 4 A100 GPUs.

3.5 Evaluations

3.5.1 Speaker Verification

All the experiment findings are displayed in Table 2. The results of the original SpeakerNet and the pre-trained checkpoint[1] publicly released by Nvidia are also provided for comparison.

[1] `https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/speakerverification_speakernet`

The pre-trained checkpoint was trained solely with noise augmentation using the above-mentioned datasets. In order to examine the speaker verification system under noisy and reverberant conditions, we created augmented versions of VoxCeleb1 clean test trials by injecting noise and echo with different SNR values.

Table 2: Comparison of different speaker verification models under noise and reverb conditions. The results are reported in equal error rates. The more aggressively noise has been applied, the lower the SNR values were. A noise level of 0 db indicates that the sound and the noise have the same energy.

Model	**Orig**	**0db**	**5db**	**10db**	**Rir**
SpeakerNet [13]	2.14	-	-	-	-
SpeakerNet (NVIDIA)	1.92	9.75	5.43	3.61	16.5
Clean Teacher	1.87	12.9	6.94	4.21	16.5
Noisy Teacher	2.6	9.35	5.84	4.23	12.74
Consistency Reg.	1.76	**8.05**	**4.40**	**3.13**	12.26
Teacher-Student	**1.73**	9.16	4.79	3.26	**9.18**

Table 2 showcases the effectiveness of the methods applied. We can see that training the SpeakerNet model with data augmentation (Noisy Teacher) improves the results in the noisy/reverberant environment with a small deterioration of EER on the original (not perturbed) audios. The Teacher-Student method achieves the lowest EER scores in original and reverberant cases (RIR), whereas the consistency regularization method shows the best results for noisy audios. Using the teacher-student method, we were able to improve the EER by an average of 19.1% compared to the multi-condition trained model. With consistency regularization, we were able to improve the EER by an average of 14.8% compared to the multi-condition trained model.

3.5.2 Speaker Diarization

We employ our trained SpeakerNet models for speaker diarization task to see which model has the smallest performance degradation in noisy conditions. We found that the optimal sliding window size and shift for speech segmentation are 1.5 and 0.5 seconds, respectively. In addition, diarization experiments are based on oracle VAD to evaluate the VAD-independent performance. The affinity matrix A is constructed using the cosine similarity between segment embeddings. We further apply the following sequence of refinement operations to the affinity matrix A:

- *Row-wise Thresholding*: For each row, keep top-12 largest elements and set the rest to 0
- *Symmetrization*: $Y = \frac{1}{2}(A + A^T)$
- *Diffusion*: $Y = AA^T$

We use the spectral clustering method [34] to obtain speaker labels. To get a full picture, we present the diarization results for both known (oracle) and unknown numbers of speakers. In the latter case, we utilize the maximal eigen-gap approach to determine the number of speakers [1].

Table 3: Comparison of speaker diarization systems with various speaker embedding extractors under noise and reverberant conditions. The results are reported in diarization error rate (DER).

Model	**Known #Speakers**						**Unknown #Speakers**					
	0db	**5db**	**10db**	**Rir**	**Orig**	**Avg**	**0db**	**5db**	**10db**	**Rir**	**Orig**	**Avg**
Clean Teacher	12.13	4.48	**1.96**	2.44	**1.26**	4.45	15.44	7.59	2.74	4.48	1.78	6.40
Noisy Teacher	**9.20**	4.49	3.13	3.12	1.57	4.30	**13.09**	7.94	4.18	4.14	1.95	6.26
Consistency Reg.	9.50	3.46	2.0	2.50	1.45	**3.78**	13.40	**4.90**	**2.57**	**3.45**	1.67	**5.20**
Teacher-Student	9.84	**3.41**	2.11	**2.43**	1.36	3.83	13.99	6.17	3.09	3.52	**1.61**	5.67

In order to assess the performance of the speaker diarization system under noisy and reverberant conditions, we modified the Voxconverse dev dataset by adding noise and echo at various signal-to-noise ratios. The results, shown in Table 3, indicate that the teacher-student and consistency regularization methods generally outperform the multi-condition baseline model for both scenarios involving known and unknown numbers of speakers. In particular, when the number of speakers is unknown, we observed approximately 17% and 9.5% relative performance improvements for the consistency regularization and teacher-student methods, respectively, compared to the multi-condition baseline.

However, it is worth noting that in certain specific scenarios, the baseline models may outperform the models with the overall best average performance.

4. Conclusions

In this research, we explore ways to increase the accuracy of speaker recognition and speaker diarization in noisy and reverberant environments, such as multi-condition, teacher-student, and consistency regularization. The key component of the methods used is the additional regularization term between embeddings for augmented and non-augmented versions of the same utterance. Through the use of teacher-student and consistency regularization, we were able to improve the performance of SpeakerNet on speaker recognition and diarization tasks in noisy and reverberant situations.

References

[1] Q. Wang, C. Downey, L. Wan, P. Mansfield and I. Moreno, "Speaker diarization with LSTM", *2018 IEEE International Conference On Acoustics, Speech And Signal Processing (ICASSP)*. pp. 5239-5243, 2018.

[2] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker diarization: A review of recent research", *IEEE Transactions On Audio, Speech, And Language Processing*, vol. 20, pp. 356-370, 2012.

[3] Q. Wang, H. Muckenhirn, K. Wilson, P. Sridhar, Z. Wu, J. Hershey, R. Saurous, R. Weiss, Y. Jia, and I. Moreno, "Voicefilter: Targeted voice separation by speaker-conditioned spectrogram masking", *ArXiv Preprint ArXiv:1810.04826*, 2018.

[4] Y. Jia, Y. Zhang, R. Weiss, Q. Wang, J. Shen, F. Ren, P. Nguyen, R. Pang, I. Lopez Moreno, Y. Wu, and Others, "Transfer learning from speaker verification to multi-

speaker text-to-speech synthesis", *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[5] E. Cooper, C. Lai, Y. Yasuda, F. Fang, X. Wang, N. Chen, and J. Yamagishi, "Zero-shot multi-speaker text-to-speech with state-of-the-art neural speaker embeddings", *ICASSP 2020-2020 IEEE International Conference On Acoustics, Speech And Signal Processing (ICASSP)*, pp. 6184-6188, 2020.

[6] E. Variani, X. Lei, E. McDermott, I. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification", *2014 IEEE International Conference On Acoustics, Speech And Signal Processing (ICASSP)*, pp. 4052-4056, 2014.

[7] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition", *2018 IEEE International Conference On Acoustics, Speech And Signal Processing (ICASSP)*, pp. 5329-5333, 2018.

[8] Y. Yu, L. Fan, and W. Li, "Ensemble additive margin softmax for speaker verification", *ICASSP 2019-2019 IEEE International Conference On Acoustics, Speech And Signal Processing (ICASSP)*, pp. 6046-6050, (2019).

[9] Z. Gao, Y. Song, I. McLoughlin, W. Guo, and L. Dai, "An improved deep embedding learning method for short duration speaker verification", International Speech Communication Association, 2018.

[10] P. Safari, M. India, and J. Hernando, "Self-attention encoding and pooling for speaker recognition", *ArXiv Preprint ArXiv:2008.01077*, 2020.

[11] J. Jung, H. Heo, I. Yang, H. Shim, and H. Yu, "A complete end-to-end speaker verification system using deep neural networks: From raw signals to verification result", *IEEE International Conference On Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5349-5353, 2018.

[12] N. Dawalatabad, M. Ravanelli, F. Grondin, J.Thienpondt, B. Desplanques and H. Na, "ECAPA-TDNN embeddings for speaker diarization", *ArXiv Preprint ArXiv:2104.01466*, 2021.

[13] N. Koluguri, J. Li, V. Lavrukhin and B. Ginsburg, "SpeakerNet: 1D depth-wise separable convolutional network for text-independent speaker recognition and verification", *ArXiv Preprint ArXiv:2010.12653*, 2020.

[14] N. Koluguri, T. Park and B. Ginsburg, "TitaNet: Neural Model for speaker representation with 1D Depth-wise separable convolutions and global context", *Proceedings of the IEEE International Conference on Acoustics, Speech And Signal Processing (ICASSP)*, pp. 8102-8106, 2022.

[15] F. Schroff, D. Kalenichenko and J. Philbin, "Facenet: A unified embedding for face recognition and clustering", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 815-823, 2015.

[16] J. Snell, K. Swersky and R.Zemel, "Prototypical networks for few-shot learning", *Advances in Neural Information Processing Systems*, vol.30, 2017.

[17] L. Wan, Q. Wang, A. Papir and I. Moreno, "Generalized end-to-end loss for speaker verification", *Proceedings of the IEEE International Conference on Acoustics, Speech And Signal Processing (ICASSP)*, pp. 4879-4883, 2018.

[18] J. Chung, J. Huh, S. Mun, M. Lee, H. Heo, S. Choe, C. Ham, S. Jung, B. Lee and I. Han, "In defence of metric learning for speaker recognition", *ArXiv Preprint ArXiv:2003.11982*, 2020.

[19] Y. Kim, H. Heo, J. Jung, Y. Kwon, B. Lee and J. Chung, "Disentangled dimensionality reduction for noise-robust speaker diarization", *ArXiv Preprint ArXiv:2110.03380*, 2021.

[20] Y. Hu, N. Hou, C. Chen E. Chng, "Dual-path style learning for end-to-end noise-robust speech recognition", *ArXiv Preprint ArXiv:2203.14838*, 2022.

[21] Q. Zhu, J. Zhang, Z. Zhang, M. Wu, X. Fang and L. Dai, "A noise-robust self-supervised pre-training model based speech representation learning for automatic speech recognition", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3174-3178, 2022.

[22] L. Moner, M. Wu, A. Raju, S. Parthasarathi, K. Kumatani, S. Sundaram, R. Maas, and B. Hoffmeister, "Improving noise robustness of automatic speech recognition via parallel data and teacher-student learning", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6475-6479, 2019.

[23] J. Deng, J. Guo, N. Xue and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4690-4699, 2019.

[24] A. Vanyan and H. Khachatrian, "Deep semi-supervised image classification algorithms: a survey", *J. Univers. Comput. Sci.*, vol. 27, pp. 1390-1407, 2021.

[25] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning", *ArXiv Preprint ArXiv:1610.02242*, 2016.

[26] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results", *Advances in Neural Information Processing Systems*, vol.30, 2017.

[27] G. Hinton, O. Vinyals and J. Dean, "Distilling the knowledge in a neural network. *ArXiv Preprint ArXiv:1503.02531*, 2015.

[28] A. Nagrani, J. Chung and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset", *ArXiv Preprint ArXiv:1706.08612*, 2017.

[29] J. Chung, A. Nagrani and A. Zisserman, "Voxceleb2: Deep speaker recognition", *ArXiv Preprint ArXiv:1806.05622*, 2018.

[30] J. Chung, J. Huh, A. Nagrani, T. Afouras and A. Zisserman, "Spot the conversation: speaker diarization in the wild", *ArXiv Preprint ArXiv:2007.01216*, 2020.

[31] D. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. Cubuk and Q. Le, "Specaugment: A simple data augmentation method for automatic speech recognition", *ArXiv Preprint ArXiv:1904.08779*, 2019.

[32] T. Ko, V. Peddinti, D. Povey, M. Seltzer and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5220-5224, 2017.

[33] D. Snyder, G. Chen and D. Povey, "Musan: A music, speech, and noise corpus", *ArXiv Preprint ArXiv:1510.08484*, 2015.

[34] U.Von Luxburg, "A tutorial on spectral clustering", *Statistics and Computing*, vol. 17, pp. 395-416, 2007.

Աղմկաղիմացկունության ապահովումը խոսնակների դիարիզացիայի համակարգում

Դավիթ Ս. Քարամյան[1,2], Գրիգոր Ա. Կիրակոսյան[2,3] և Սաթեն Ա. Հարությունյան[2]

[1]Հայ-Ռուսական համալսարան, Երևան, Հայաստան
[2]Krisp.ai, Երևան, Հայաստան
[3]ՀՀ ԳԱԱ մաթեմատիկայի ինստիտուտ,Երևան, Հայաստան
e-mail: {dkaramyan, sharutyunyan, gkirakosyan }@krisp.ai

Ամփոփում

Խոսնակների դիարիզացիայի նպատակը աուդիո ձայնագրության մեջ տարբեր խոսնակների հայտնաբերումն ու առանձնացումն է: Այնուամենայնիվ, ֆոնային աղմուկը կարող է ազդել այս համակարգերի ճշգրտության վրա: Այս հոդվածում ուսումնասիրվել են այնպիսի մեթոդներ, ինչպիսիք են՝ տարբեր աուգմենտացիաներով ուսուցումը, կայունության կարգավորումը (consistency regularization) և ուսուցիչ-աշակերտ մեթոդը՝ խոսնակների ձայնային հատկանիշներ դուրս բերող մոդելի կայունությունը աղմուկի նկատմամբ բարձրացնելու համար: Նշված մեթոդների արդյունավետությունը ստուգվել է խոսնակների նույնականացման և դիարիզացիայի խնդիրներում և ցույց է տրվել, որ դրանք հանգեցնում են կայունության բարելավմանը՝ աղմուկի և արձագանքի առկայության դեպքում: Խոսնակների նույնականացման և դիարիզացիայի համակարգերը աղմուկի և արձագանքի պայմաններում փորձարկելու համար ստեղծվել են VoxCeleb1 և Voxconverse dev տվյալների հավաքածուների ընդլայնված տարբերակները՝ ավելացնելով տարբեր SNR արժեքներով ֆոնային աղմուկ և արձագանք: Ստացված արդյունքները ցույց են տալիս, որ միջին հաշվով կարելի է հասնել խոսնակների նույնականացման ճշգրտության հարաբերական բարելավմանը՝ 19,1%-ով՝ օգտագործելով ուսուցիչ-աշակերտ մեթոդը և խոսնակների դիարիզացիայի ճշգրտության հարաբերական բարելավմանը՝ 17%-ով՝ օգտագործելով կայունության կարգավորման մեթոդը՝ համեմատած տարբեր աուգմենտացիաներով վարժեցված մոդելի հետ:

Բանալի բառեր՝ խոսնակների նույնականացում, խոսնակների դիարիզացիա, աղմկա-դիմացկունություն, ուսուցիչ-աշակերտ, կայունության կարգավորում:

Обеспечение шумоустойчивости системы диаризации дикторов

Давид С. Карамян[1,2], Григор А. Киракосян[2,3] и Сатен А. Арутюнян[2]

[1]Российско-Армянский университет, Ереван, Армения
[2]Krisp.ai, Ереван, Армения
[3]Институт математики НАН РА, Ереван, Армения
e-mail: {dkaramyan, sharutyunyan, gkirakosyan}@krisp.ai

Аннотация

Целью системы диаризации дикторов является идентифицирование и разделениеразных дикторов в аудиозаписи. Однако шум в записи может повлиять на точность этих систем. В этой статье мы исследуем такие методы, как обучение с различными аугментациями, регуляризация согласованности (consistency regularization) и метод "учитель-ученик", чтобы повысить устойчивость экстракторов речевых характеристик к шуму. Мы проверяем эффективность этих методов в задачах распознавания дикторов по голосу и диаризации дикторов и демонстрируем, что они приводят к улучшению устойчивости при наличии шума и реверберации. Чтобы проверить систему распознавания и диаризации дикторов в условиях шума и реверберации, мы создали расширенные версии VoxCeleb1 и наборов данных Voxconverse dev, добавив шум и эхо с разными значениями SNR. Наши результаты показывают, что в среднем мы можем добиться относительного улучшения распознавания дикторов на $19,1\%$ с использованием метода "учитель-ученик" и относительного улучшения диаризации дикторов на 17% с использованием метода регуляризации согласованности по сравнению с базовой моделью, обученной с помощью различных аугментаций.

Ключевые слова:распознавание по голосу, диаризация дикторов, устойчивость к шуму, учитель-ученик, регуляризация согласованности.

Mathematical Problems of Computer Science 59, 69–81, 2023.
doi: 10.51408/1963-0103

UDC 004.725, 004.852

Research of Model Increasing Reliability Intrusion Detection Systems

Timur V. Jamgharyan

National Polytechnic University of Armenia, Yrevan, Armenia
e-mail: t.jamgharyan@yandex.ru

Abstract

The paper presents the results of the using, a recurrent neural network to detect malicious software as part of the *Snort* intrusion detection system.The *research* was conducted on datasets generated on the basis of *athena, dyre, engrat, grum, mimikatz, surtr* malware exploiting vulnerability *CVE-2022-20685* in the *Snort* intrusion detection system. Processing of input traffic data was carried out before the *frag-3* and *modbus* preprocessors. The method of *k nearest neighbors* was used as a mathematical apparatus. The simulation of the developed software at different iterations.

All research results are available at https://github.com/T-JN

Keywords: Machine learning, Dataset, Malware, Preprocessor, Metasploit, k nearest neighbors method, Intrusion detection system.

Article info: Received 8 January 2023; send to review 7 February 2023; accepted 7 March 2023.

1. Introduction

The intrusion detection systems (IDS) include many different software components designed to detect various types of traffic with an embedded malicious component. Detection is carried out according to a set of rules that are configured based on the threat model and security policies. The security architecture of the Network Infrastructure (NI) is built taking into account possible attacks according to various models: triad CIA (Confindentiality, Integrity, Availability, CIA), Parker's hexad [1]. Network IDS, unlike host IDS, detect attacks directed at the network segment and contain a set of complementary rules and security scripts that can neutralize an attack on the network. Unlike host-based IDS, network-based IDS require more computing resources due to the fact that a larger set of rules and detectors is activated during their operation [2]. When using host IDS in the Infrastructure for a fleet of computing systems running Linux OS, can disable

the rules for Windows (or another OS), but hardly possible for a network IDS, since different operating systems are used in the Infrastructure. Modern IDS are able to detect various types of attacks at different levels of the OSI (Open System Interconnection, OSI) model: bad traffic, system scanning, the use of known exploits to attack over various protocols, various backdoors, various known malware [3]. A significant limitation of systems for analyzing network traffic and the state of NI is the algorithmic and functional determinism inherent in them.

An important issue of Infrastructure security is the reliability of the processed data of the IDS itself (*data reliability – is, the property of the processed data not to have hidden errors* [4]). The processing of data streams in the IDS itself is determined by the functioning algorithms, data presentation formats, and the formalization of signature classifiers. Protecting the IDS signature database (both remote and local) is also one of the most important tasks. If the signatures database has been attacked for availability, then when a new vulnerability appears, the IDS will not receive the necessary signature and the Infrastructure perimeter will become vulnerable [5]. The development of M2M (Machine-to-Machine, M2M) and ML (Machine learning, ML) technologies has increased the capabilities of both attack and defense tools. Various researchers are conducting research on increasing (improving) various parameters of IDS with ML [6, 7, 8]. One of the parameters that improves when using ML modules as part of a standard IDS is its variability. Unlike deterministic IDS, IDS with ML are capable of forming a multi-criteria sample on the basis of which the detector operation scheme is formed within the given constraints. But IDS with ML have certain limitations when integrating them into the NI architecture. In particular, ML IDS are very sensitive to various implementations of *«noise attacks» («noise attack» is a variant of an availability attack in which a large number of random and meaningless fragmented packets are sent to the attacked system, some of which contain malware* [9]). A dangerous consequence of a «noise attack» on a ML network IDS is that attackers «attack» it for a long time with streams of datasets that cause false positives, «teach» the ML IDS discriminator to be immune to this type of traffic (creating a cyclic chain of operations: false positive--true negative--false negative--true positive, which overload both the IDS itself and the SIEM system (Security information and event management, SIEM).

Various manufacturers combine IDS modules into different classes, which allows you to quickly reconfigure the IDS itself for specific tasks. In particular, for Snort open source IDS, there are many different types of preprocessors (*frag-3, stream, performance monitor, SMTP, POP, IMAP, SSH, DNS, DCE/RPC, SIP preprocessors, reputation preprocessor, modbus preprocessor*) each of which is functionally is responsible for handling the given protocol and/or data type.

- *IDS preprocessor is a software module that receives data from the network traffic decoding module and outputs them to the input of intrusion detection modules.*

As stated in the article «Attacks on Machine Learning Systems» [10], the most vulnerable part of the ML IDS is the traditional IDS component (the deterministic part of the IDS). ML systems, like any other, will be hacked using vulnerabilities in these traditional components. The use of ML at the preprocessor level is due to the fact that when developing an IDS with ML, it is not enough to create a functioning model that can detect a threat not described in a set of rules (signatures) or generate new ones based on «known» signatures, but it is also necessary to protect the IDS itself from probable infection with malware that can compromise the reliability of the results issued by IDS.The choice of using a neural network at the preprocessor level is also due to the fact that the IDS, which has a neural network in its component composition after the preprocessor, is able to protect the NI, since malware not detected by standard datasets (described in the signature/rule database) will be detected with varying probability neural network. But with a «noise attack», the target is the IDS itself, which, when taken out of the reliable functioning mode, will no longer detect malware. Undescribed at the preprocessor level,

malicious data embedded in IDS can be detected using performance preprocessors that evaluate various kinds of statistics. But the problem is that, having determined the type of network IDS, attackers can design an attack taking into account the work of preprocessors, and malware embedded in the IDS itself will not go beyond the allowable statistical deviations. A lot of research has been devoted to the task of applying machine learning as part of IDS, but only a small part of them explores the use of machine learning at the preprocessor level. This limitation, in particular, is due to the fact that the «response» of the neural network is probabilistic in nature and it is necessary to introduce clear boundaries for the neural network itself. Otherwise, the neural network will be an event generator, which will be classified as an attack by the IDS detection modules. Thus, there is a recursion to the problem of stability and integrity of both the IDS and the NI as a whole [11]. *This research explores the potential of a recurrent neural network (RNN) to detect malware at the preprocessor level*. The choice in the research of RNN from the entire set of neural networks is determined by the fact that RNN form a directed sequence between elements, which allows processing a series of events in time (this characteristic allows granular processing of fragmented datasets). The relevance of the work lies in the ever-increasing role of IDS with ML in the NI security architecture and the increasing security requirements of the IDS itself. The use of a neural network at the preprocessor level will increase the reliability of malware detection results without affecting the main IDS signature database, which will reduce the attack surface for the IDS itself. The novelty of the research lies in the application of the k nearest neighbors (k Nearest Neighbors, kNN) method to detect malware in IDS *before preprocessors*.

- *The k nearest neighbors method is a metric algorithm for classifying objects.*

Malicious software athena, dyre, engrat, grum, mimikatz, surtr obtained from publicly available sources was used as calibration data [12--15]. The choice of the kNN method is determined by the fact that it is necessary to minimize the value of the preprocessor error, and for this it is necessary to carry out a preliminary grouping and classification of unknown input datasets in normalized traffic.

- *Traffic normalization - modification of packets of protocols of the transport, and network levels for their subsequent processing by IDS detection modules.*

2. Formulation and Description the Problem

It is necessary to detect a malicious dataset in normalized traffic.

The mathematical model construction was carried out on the basis of the formulas obtained in the sources [16,17]. There are network traffic X inputs that contain malware fragments (1).

$$X^m = \{(x_1, y_1), \dots, (x_m, y_m)\}, \quad (1)$$

where,

x_m- network traffic datasets that do not contain malicious components,

y_m- network traffic datasets containing malicious components,

m- number of the analyzed packet of the input dataset.

On the set of input traffic data sets, the distance function $x\rho(y, y')$ is given. The greater the value of the distance function, the less similar the entities are y, y', where y'- the minimum size of a malware dataset that can be uniquely identified and classified with respect to y. For any entity v in the data package, arrange the objects x_i in ascending order (2).

$$\rho(v, x_{1;v}) \le \rho(v, x_{2;v}) \le \cdots \le \rho(v, x_{m;v}), \quad (2)$$

where $x_{i;v}$ the set of network traffic data that is the i-th neighbor of the entity v. Similarly for the i -th neighbor of the entity v in the dataset $y_{i;v}$. Using the formula (3 from the source [17], we determine the malicious kNN components for the traffic arriving in the NI.

$$\alpha(v) = \arg\max_{y \in Y} \sum_{i=1}^{m} [y(x_{i;v}) = y]\, \omega(i, v), \tag{3}$$

where, $\omega(i, v)$- a given weight function that evaluates the degree of importance of the i-th neighbor for the classification of the entity v. By changing the $\omega(i, v)$ value, you can get different versions of the k nearest neighbors method (4).

$$\omega(i, v) = [i \leq k]. \tag{4}$$

When $\omega(i, v) = [i = 1]$ malware is detected only in the given single value ω. That is, the RNN is only able to detect the malware datasets it was trained on. A graphical representation of a RNN is shown in Fig. 1.

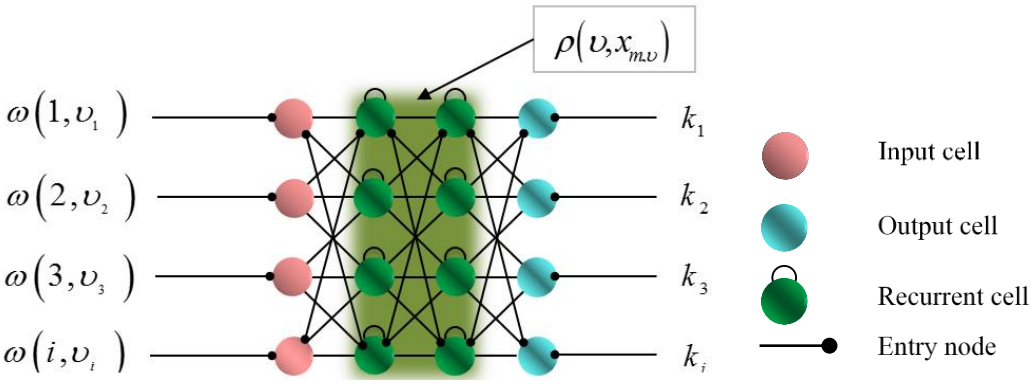

Fig. 1. Recurrent neural network.

Attackers can load malware into the IDS itself not in a single package, but in fragments (using the built-in *frag-3* preprocessor as an internal attack tool), then the research task of grouping and classifying malware fragments arises. Standard IDS do not cope with this task very effectively, but ML IDS, in the presence of a training set, are able to solve this problem. The disadvantage of ML IDS is that they can produce unreliable results if the preprocessor responsible for a particular type of traffic/protocol is «damaged» as a result of a *«noise attack»*. A particular danger lies in the fact that any traffic entering the IDS preprocessors (both ML and deterministic) is not checked for malicious components, since the task of the preprocessor is to «reformat» traffic for processing by detectors.

3. Task Statement

It is necessary to develop and programmatically implement an algorithm and, based on it, software that integrates a RNN capable of solving the problem of grouping and classification with the IDS preprocessor.

4. Boundary Conditions

1. The smallest fragment of the malware file (ξ) that can be classified $\xi = 20byte$ (detection was carried out using context-piecewise hashing (Context Triggered Piecewise Hashing, CTPH), which is discussed in detail in [18].
2. The delay in the processed module should not cause a «signal race». Traffic from the output of the preprocessor module to the input of the detection modules must be sent synchronously. As part of this condition, an additional restriction has been introduced - only UDP (User Datagram Protocol, UDP) traffic is processed.
3. The hardware must support the parallel computing mode.

The developed software connects the RNN to *frag-3* and *modbus* preprocessors (*frag- 3* preprocessor for defragmenting an IP packet, *modbus* - preprocessor for processing data from a variety of devices operating in SCADA networks (Supervisory Control And Data Acquisition, SCADA).Since the *frag-3* preprocessor is designed to build packages, using a trained RNN can neutralize the process of «assembling» malicious packages inside the IDS, increasing the level of reliability of its functioning. On Fig.2 shows a diagram of the *Snort* IDS with the proposed data processing software implemented on RNN.

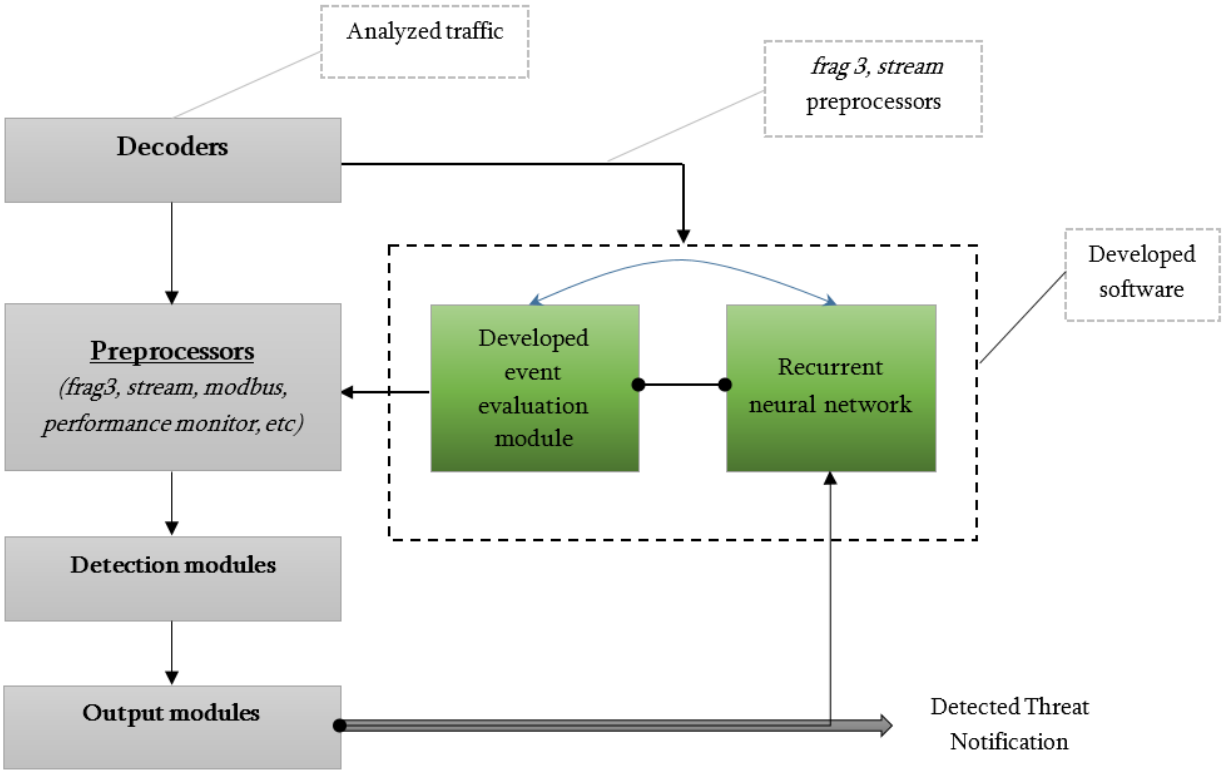

Fig. 2. Snort IDS with developed data processing software.

5. Description of the Module

The network traffic coming from the decoders is directed to the preprocessor processing module (standard operation of the *Snort* IDS). The traffic that should processed by the *frag - 3* and *modbus* preprocessors is sent to the developed module based on the RNN. After processing according to the developed algorithm, this traffic is again sent to the standard detection modules. The task of the module is to carry out the primary «cut-off» of possible malware and protect the IDS itself from being modified by malware.

The developed algorithm is shown in Fig. 3.

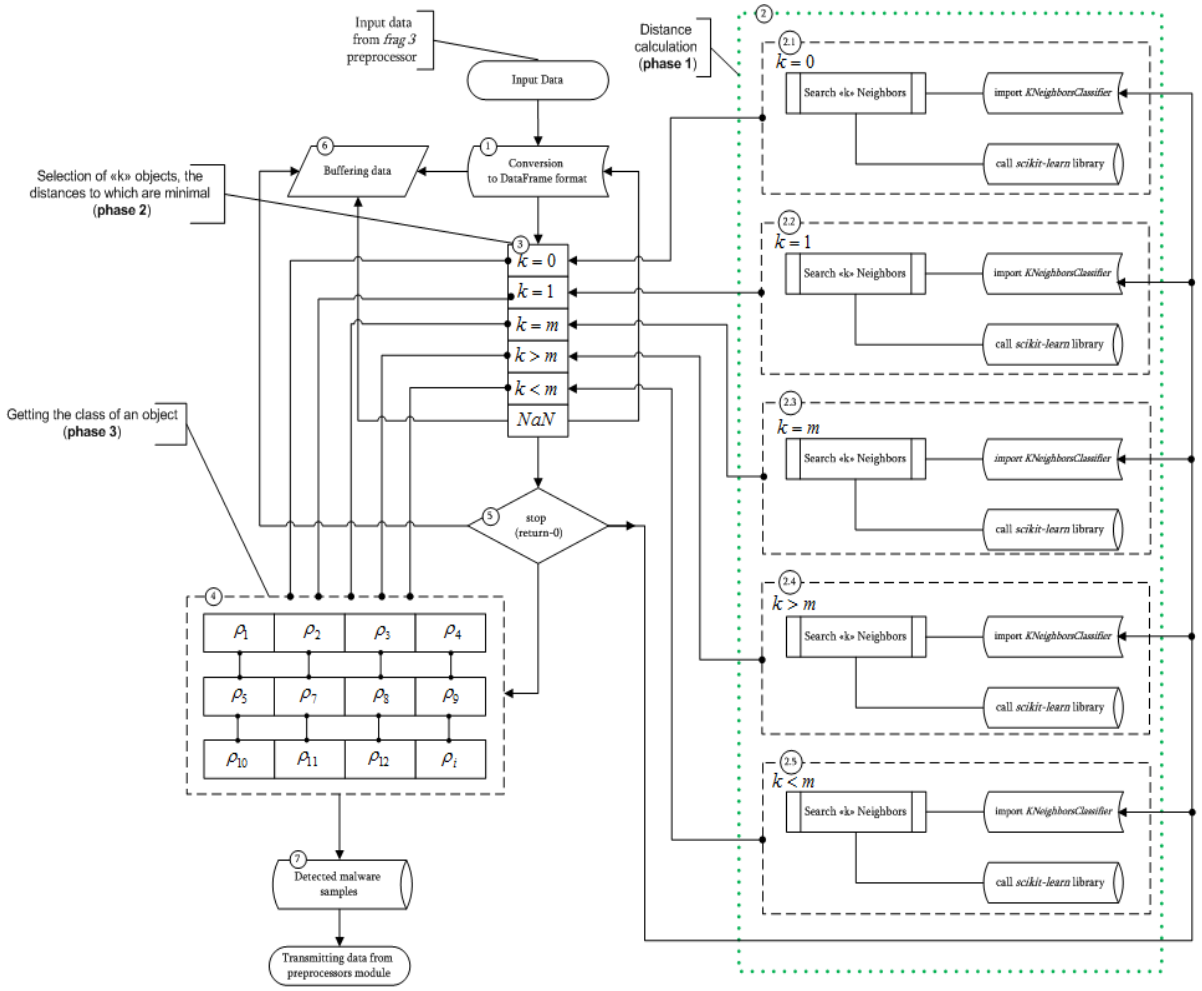

Fig. 3. Developed algorithm.

Algorithm operation

The software that searches for fragmented malware receives network traffic datasets from a decoder (*Snort* IDS a low-level interceptor) as input. Only traffic that must be processed by the *frag-3* and *modbus* preprocessors is subject to processing.

Step 1. Converting received datasets to «Data Frame». This conversion is necessary to speed up the work of the RNN, since the traffic not processed by the developed module goes directly to

the preprocessor module and the processing delay should not exceed the boundary conditions (boundary condition 2).

Step 2 phase 1. *Calculation of the distance from the target object, which must be classified to each of the sample objects (traffic).* Computing a distance metric between likely malware datasets. All calculations are performed in parallel mode (boundary condition 3),

- **2.1 k=0** calculation of the distance metric and detection of malicious datasets is not performed, since the classification of malicious and non-malicious datasets is impossible,
- **2.2 k=1** the distance between malicious and non-malicious datasets is constant (k=const). Only those malicious datasets that fall within the specified distance metric are detected,
- **2.3 k=m** continuous detection mode.Upper limit: the value of m that the hardware can handle,
- **2.4 k>m** malicious datasets are not detected,
- **2.5 k<m** malicious datasets are detected down to the minimum CTPH value. All calculations were based on the scikit-learn ML library (using instances of the *kNeighborsClassifier* class).

Step 3 phase 2. *Selection of k objects from the sample, the distances to which are minimal.*
The RNN to fed only datasets, where corresponding to paragraphs 2.2, 2.3, 2.5. When a number value with an undefined result NaN (Not-a-Number, NaN) appears in the handler, the execution of the entire program is «stopped», which resets all values to zero (step 5).

Step 4 phase 3. *Obtaining a class of sample objects based on the most frequently occurring k.*
Setting the «weights» of the RNN. The weight setting is determined by the number of malware hash values detected by the CTPH method. Increasing the value $\rho(\upsilon_i, x_{m;\upsilon})$ (increasing the number of hits) for a certain type of dataset increases the «weight» of this dataset in the RNN. The output is a class of malware datasets.

Step 5. Stop and reset all values when NaN values appear in the dataset.

Step 6. Buffering values one step before zeroing. The buffer always contains n-1 dataset values (the n-dataset currently being processed).

Step 7. Detected malware datasets.

Step 8. Transfer of traffic to the input of the preprocessor module.

All class instances are implemented based on the *StandardScaler* library. The training was carried out on the basis of the *fit* software library.

6. Description of the Experiment

In Windows Server 2016 Standard operating system environment installed the Hyper-V role (Based on the Dell Power Edge T-330 server). A software-defined network (SDN) has been deploy, in which Parrot OS is installed with the Metasploit framework and Ubuntu v20.04 OS in which are installed: IDS *Snort* version 2.9.18, *Clion* development environment and developed software. The introduction of traffic with malware that could lead to a denial of service for the *Snort* IDS and an attack on the Infrastructure was carried out using the Metasploit framework based on the Parrot OS pentest distribution kit. The malicious input was based on a *pcap* network traffic dump file. The choice of version 2.9.18.1 of the *Snort* IDS is due to the fact that in this version there is a vulnerability *CVE-2022-20685* (*CVE-2022-20685 Snort* IDS vulnerability leading to a denial of service, bypassing security restrictions and compromising the system [19]) when exploited, attackers can inject malware into the IDS itself and attack the Infrastructure. With the correct operation of the developed software, the attack should be detected, which will make it possible to further check the effectiveness of the software for possible and probable

unknown attacks. Through this vulnerability, *athena, dyre, engrat, grum, mimikatz, surtr* malware was introduced into the virtual Infrastructure. The Windows Server 2016 operating system, which is the *test.local* domain controller, and the Windows 10 client machine were used as the protected Infrastructure. To increase the reliability of the experiment results, all virtual machines are connected to each other by a *private* virtual adapter and connected to different VLAN (Virtual Local Area Network, VLAN, with vlan ID=100 and vlan ID=101). Network address translation (NAT) is configured between virtual networks 172.16.0.0/30 and 192.168.0.0/29.

The experiment was carried out in 2 stages.

Stage 1.

Injection of *mimikatz* malware through *CVE-2022-20685* with *kNN*-based detection software disabled. In the first case, the IDS did not detect the intrusion, and the *mimikatz* software implemented through the Snort IDS in the «*noise attack*» mode compromised the domain administrator's password and did not register the *Snort* network IDS in any way.

Stage 2.

Introduction of various types of malware (*athena, dyre, engrat, grum, mimikatz, surtr*) into the Infrastructure through a vulnerability in the *Snort* network IDS. The *mimikatz, surtr, engrat, and grum* malware were detected immediately, while the athena and dyre malware was detected after the second iteration.

The scheme of the experiment is shown in Fig. 4.

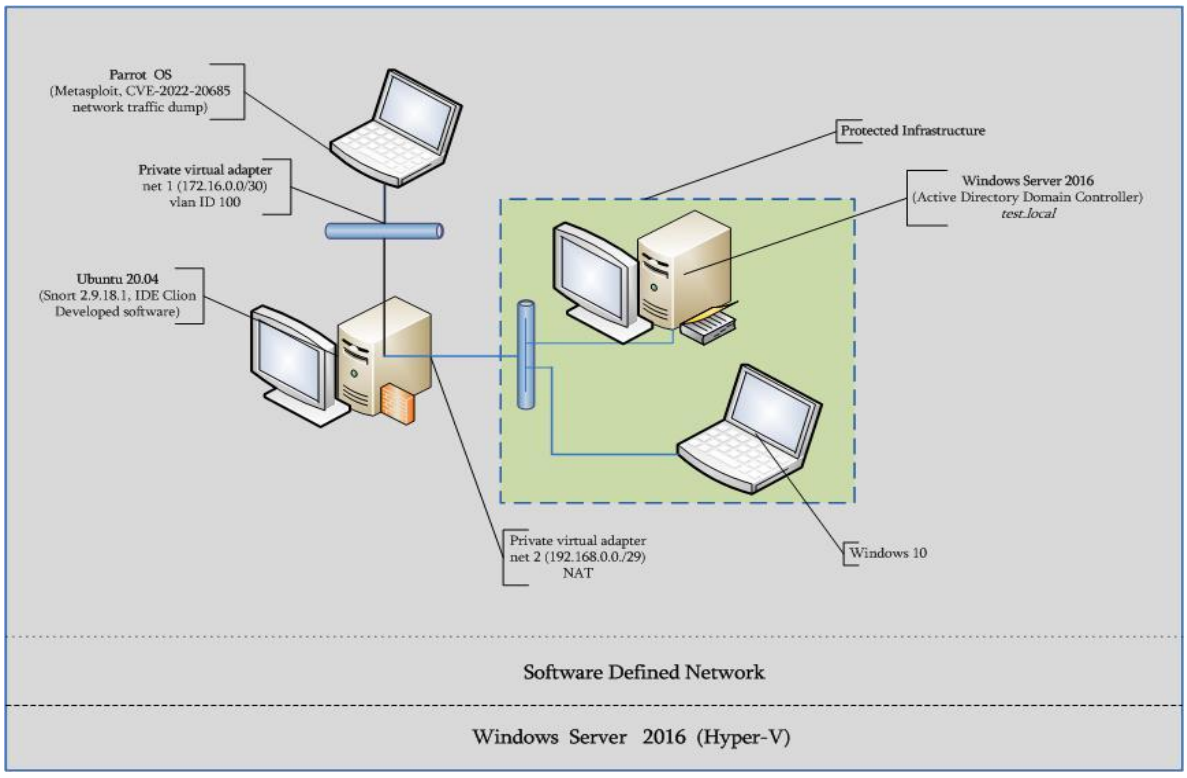

Fig. 4. Scheme of the experiment in SDN.

7. Results

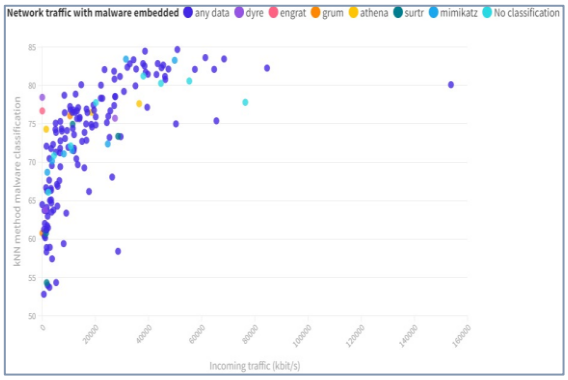

Fig. 5. Visualization of datasets classified by the kNN method of malware (I-iteration).

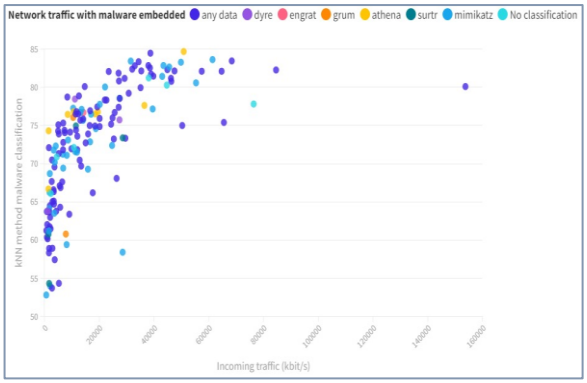

Fig. 6. Visualization of datasets classified by the kNN method of malware (II-iteration).

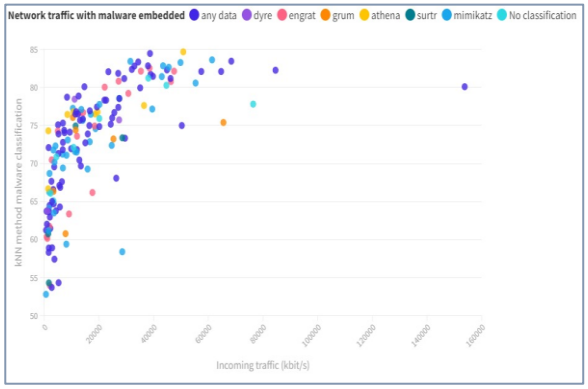

Fig. 7. Visualization of datasets classified by the kNN method of malware (III-iteration).

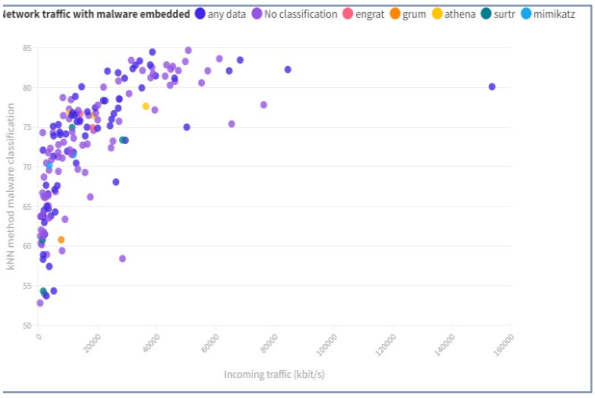

Fig. 8. Visualization of datasets classified by the kNN method of malware (IV-iteration).

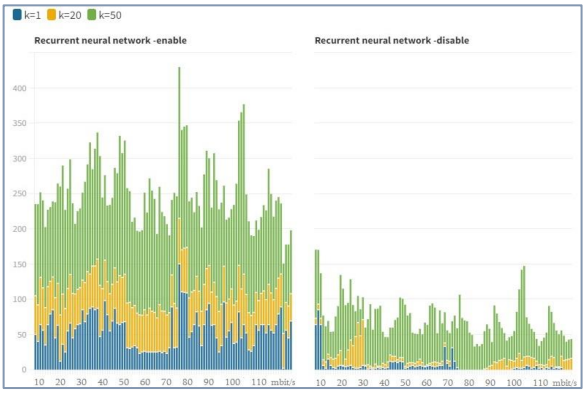

Fig. 9. Visualization of datasets classified by the kNN method of malware. k=1, 20, 50.

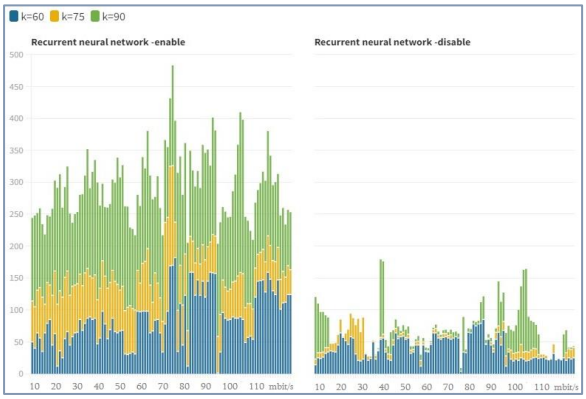

Fig. 10. Visualization of datasets classified by the kNN method of malware. k=60, 75, 90.

As part of the all research, was developed an IDS with ML. The results of the first model on a real infrastructure are presented in Fig. 11,12. At this research stage, the sixth version of the model has been developed and tested in SDN [20].

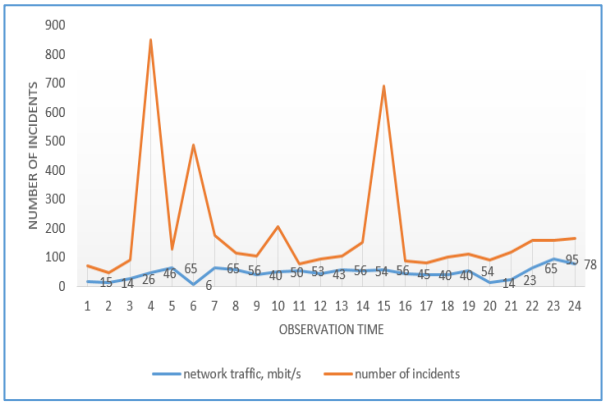

Fig. 11. Visualization of the work of the Snort IDS in a 24-hour period without a module with ML.

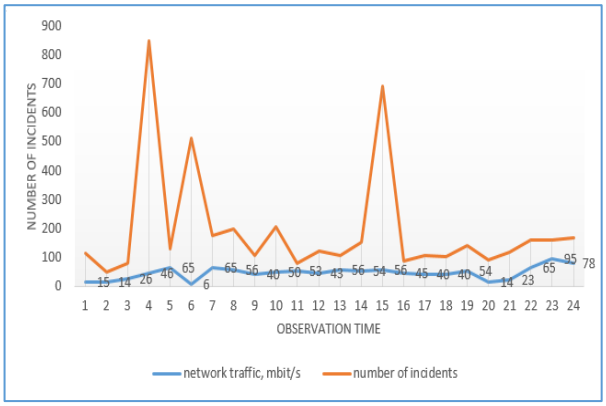

Fig. 12. Visualization of the work of the Snort IDS in a 24-hour period with a ML module.

Explanation of visualized results

The Fig. 5,6,7,8 present a visualization of the distribution of detected and classified malicious datasets embedded in network traffic at different iterations. The first and second iterations, the percentage of malware detection is about (7.6-8)%, the percentage of classification is less than 3%. The third iteration, the improvement in the solution of the detection problem is insignificant (7.9-8.02)%, but the solution of the classification problem becomes acceptable for practical use (14-16)%. An increase in the number of iterations on the same dataset leads to retraining of the RNN and an avalanche deterioration in the results of solving the problem of malware classification (Fig. 8). The most effective detection occurs at speeds up to 50-60 Mbps. The results of the work of the developed software integrated into the IDS Snort in various modes shows on Fig. 9,10. As can be seen from Fig. 9, 10, the use of a RNN at the level before the preprocessor increases the reliability of the data processed in the network IDS. An important factor when using a RNN before the preprocessor is the need for training datasets to differ not only quantitatively, but also variably.

Increase, in efficiency by (10-12)% managed to achieve only, the CTPH method.

8. Conclusion

The paper considers a software model for detecting malware using a RNN as part of the *Snort* version 2.9.18.1 IDS. A *pcap* network traffic file with embedded malware was used as a dataset. The training datasets for RNN are based on the source code of malware obtained from open sources. The *k nearest neighbors* method was used as a mathematical apparatus for solving the classification problem.

Based on the research, it can be concluded:

The use of the *k nearest neighbors* method at the *preprocessor level* is justified in the presence of a large and unique training dataset.

The use of augmentation for training a, RNN included in the IDS before the preprocessor is inappropriate, since solving the classification problem using the *k nearest neighbors* method requires a data set with unique data that differ from each other in many criteria, which is difficult to achieve using the augmentation method. The use of RNN as part of an IDS at the preprocessor level is justified in the presence of a large computing resource (a special role is played by the amount and type of RAM).

References

[1] G.Stoneburner, "*Underlying Technical Models for Information Technology Security*", NIST Special Publication 800-33, 2001.

[2] R.Atefinia, M.Ahmadi, Performance Evaluation of Apache Spark Mlib Algorithms on an Untrusion Detection Dataset. [Online].Available:https://arxiv.org/abs/2212.05269

[3] M. Bachi, A. Harti, J. Fabini and T. Zseby, Walling up Backdoors in Intrusion Detection Systems. [Online].Available:https://arxiv.org/abs/1909.07866

[4] National standard of the Russian Federation, "Quality of official information", GOST R-51170-98, (2020)// 12, Moscow, Standardinform.

[5] B.E.Zolbayar et al, "Generating practical adversarial network traffic flows using NIDSGAN", [Online].Available:https://arxiv.org/abs/2203.06694

[6] F. Zhong et al, "MalFox: Camouflaged adversarial malware example generation based on Conv-GAN againist black—box detectors", [Online].Available:https://arxiv.org/abs/2011.01509

[7] Dominik Kus et al, "A false sense of security? Revisting the state of machine learning-based industrial intrusion system", [Online].Available:https://arxiv.org/abs/2205.09199

[8] K.Jallad, M. Aljnidi and M.Desoki, «Big data analysis and distributed deep learning for next-generation intrusion detection system optimization», (2022)//[Online].Available: https://arxiv.org/abs/2209.13961

[9] A. Branitsky and I. Kotenko, «Analysis and classification of methods for detecting network attacks», Proceedings of SPIIRAS, (2016) // issue 45, pp. 207-244.

[10] Electronic resource dedicated to digital transformation technologies. [Online].Available:https://www.osp.ru/os/2020/03/13055601

[11] T. V. Jamgharyan and V.H.Ispiryan, "Network infrastructures assessment stability" *Proceedings of 13th International Conference on Computer Science and Information Technologies (CSIT)*, Yerevan, Armenia, pp. 199-203, 2021.

[12] Malware Bazaar Database. [Online]. Available:https://bazaar.abuse.ch/browse/

[13] Malware database. [Online]. Available:http://vxvault.net/ViriList.php

[14] Malware repository. [Online]. Available:https://avcaesar.malware.lu/

[15] Viruses repository. [Online]. Available:https://virusshare.com/

[16] G.Campos, A.Zimek, et al, «On the evaluation of unsupervised outlier detection: measures,datasets, and an empirical study». [Online].Available:https://link.springer.com/article/10.1007/s10618-015-0444-8

[17] Professional information and analytical resource dedicated to machine learning, pattern recognition and data mining. [Online].Available: http://www. machinelearning.ru

[18] T.Jamgharyan, “Research of obfuscated malware with a capsule neural network”, *Mathematical Problems of Computer Science*, vol. 58, 67–83, 2022.
[19] Website for identifying, defining and cataloging publicly disclosed cybersecurity vulnerabilities.
[Online].Available:https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-20685
[20] T.Jamgharyan, “Modernization of intrusion detection system via the generative model”, *«Haikakan Banak» («Armenian Army») Defense-Academic journal, National Defense Research University*, Ministry of Defense, Republic of Armenia, no. 2, pp.75-79, 2021.
[Online].Available:https://razmavaraget.files.wordpress.com/2022/01/hb2-final.pdf

Ներխուժումների հայտնաբերման համակարգի հավաստիության բարձրացման մոդելի հետազոտում

Թիմուր Վ. Ջամղարյան

Հայաստանի ազգային պոլիտեխնիկական համալսարան, Երևան, Հայաստան
e-mail: t.jamgharyan@yandex.ru

Ամփոփում

Հոդվածում ներկայացված են Snort 2.9.18.1 ներխուժումների հայտնաբերման համակարգի կազմում ռեկուրենտ նեյրոնային ցանցի կիրառման հետազոտության արդյունքները: Հետազոտությունն իրականացվել է athena, dyre, engrat, grum, mimikatz, surtr վնասաբեր ծրագրային ապահովման ելակետային կոդի հիման վրա *կառուցած* տվյալների հավաքածուներով: Շահագործվել է CVE-2022-20685 Snort ներխուժումների հայտնաբերման համակարգում խոցելիությունը: Մուտքային թրաֆիկի մշակումը իրականացվել է մինչ frag-3 և modbus պրեպրոցեսորները: Որպես մաթեմատիկական ապարատ օգտագործվել է k մոտակա հարևանների մեթոդը: Իրականացվել է ծրագրային ապահովման իրագործման մոդելավորում տարբեր կրկնություններում և արդյունքների արտացոլում: Հոդվածում չներառված հետազոտության արդյունքները հասանելի են https://github.com/T-JN կայքում:

Բանալի բառեր՝ մեքենայական ուսուցում, տվյալների հավաքածու, վնասաբեր ծրագրային ապահովում, k մոտակա հարևանների մեթոդը, ներխուժումների հայտնաբերման համակարգ, *CVE-2022-2068*:

Исследование модели повышения достоверности системы обнаружения вторжений

Тимур В. Джамгарян

Национальный политехнический университет Армении, Ереван, Армения
e-mail: t.jamgharyan@yandex.ru

Аннотация

В статье представлены результаты исследования применения рекуррентной нейронной сети для обнаружения вредоносного программного обеспечения в составе системы обнаружения вторжений *Snort*. Исследование проводилось на наборах данных сформированных на основе вредоносного программного обеспечения *athena, dyre, engrat, grum, mimikatz, surtr* с эксплуатацией в системе обнаружения вторжений *Snort* версии 2.9.18.1 уязвимости *CVE-2022-20685*. Обработка данных входного трафика осуществлялась *до препроцессоров frag-3* и *modbus*. В качестве математического аппарата использовался метод *k ближайших соседей*. Проведено моделирование работы программного обеспечения при разных итерациях и визуализация результатов. Результаты исследования не внесенные в статью представлены по адресу https://github.com/T-JN

Ключевые слова: машинное обучение, вредоносное ПО, метод ближайших соседей, система обнаружения вторжений, препроцессор, *CVE-2022-2068*.

Կանոններ հեղինակների համար

ՀՀ ԳԱԱ ԻԱՊԻ “Կոմպյուտերային գիտության մաթեմատիկական խնդիրներ” պարբերականը տպագրվում է 1963 թվականից: Պարբերականում հրատարակվում են նշված ոլորտին առնչվող գիտական հոդվածներ, որոնք պարունակում են նոր՝ չհրատարակված արդյունքներ:

Հոդվածները ներկայացվում են անգլերեն՝ ձևավորված համապատասխան “ոճով” (style): Հոդվածի ձևավորման պահանջներին ավելի մանրամասն կարելի է ծանոթանալ պարբերականի կայքէջում՝ http://mpcs.sci.am/:

Rules for authors

The periodical “Mathematical Problems of Computer Science” of IIAP NAS RA has been published since 1963. Scientific articles related to the noted fields with novel and previously unpublished results are published in the periodical.

Papers should be submitted in English and prepared in the appropriate style. For more information, please visit the periodical's website at http://mpcs.sci.am/.

Правила для авторов

Журнал «Математические проблемы компьютерных наук» ИПИА НАН РА издается с 1963 года. В журнале публикуются научные статьи в указанной области, содержащие новые и ранее не опубликованные результаты.

Статьи представляются на английском языке и оформляются в соответствующем стиле. Дополнительную информацию можно получить на веб-сайте журнала: http://mpcs.sci.am/.

The electronic version of the periodical “Mathematical Problems of Computer Science” and rules for authors are available at

http://mpcs.sci.am/

Phone: (+37460) 62-35-51
Fax: (+37410) 28-20-50
E-mail: mpcs@sci.am
Website: http://mpcs.sci.am/

Ստորագրված է տպագրության՝ 25.05.2023
Թուղթը՝ օֆսեթ:
Հրատարակված է ՀՀ ԳԱԱ Ինֆորմատիկայի և ավտոմատացման պրոբլեմների ինստիտուտի կողմից
Ծավալը՝ 83 էջ: Տպաքանակը՝ 100
ՀՀ ԳԱԱ ԻԱՊԻ Համակարգչային պոլիգրաֆիայի լաբորատորիա
Երևան, Պ. Սևակի 1
Հեռ. +(374 60) 623553
Գինը՝ անվճար

Подписано в печать 25.05.2023
Офсетная бумага.
Опубликовано Институтом проблем информатики и автоматизации НАН РА
Объём: 83 страниц. Тираж: 100
Лаборатория компьютерной полиграфии ИПИА НАН РА.
Ереван, П. Севака 1
Тел.: +(374 60) 623553
Цена: бесплатно

Signed in print 25.05.2023
Offset paper
Published by the Institute for Informatics and Automation Problems of NAS RA
Volume: 83 pages
Circulation: 100
Computer Printing Lab of IIAP NAS RA
Yerevan, 1, P. Sevak str.
Phone: +(374 60) 623553
Free of charge