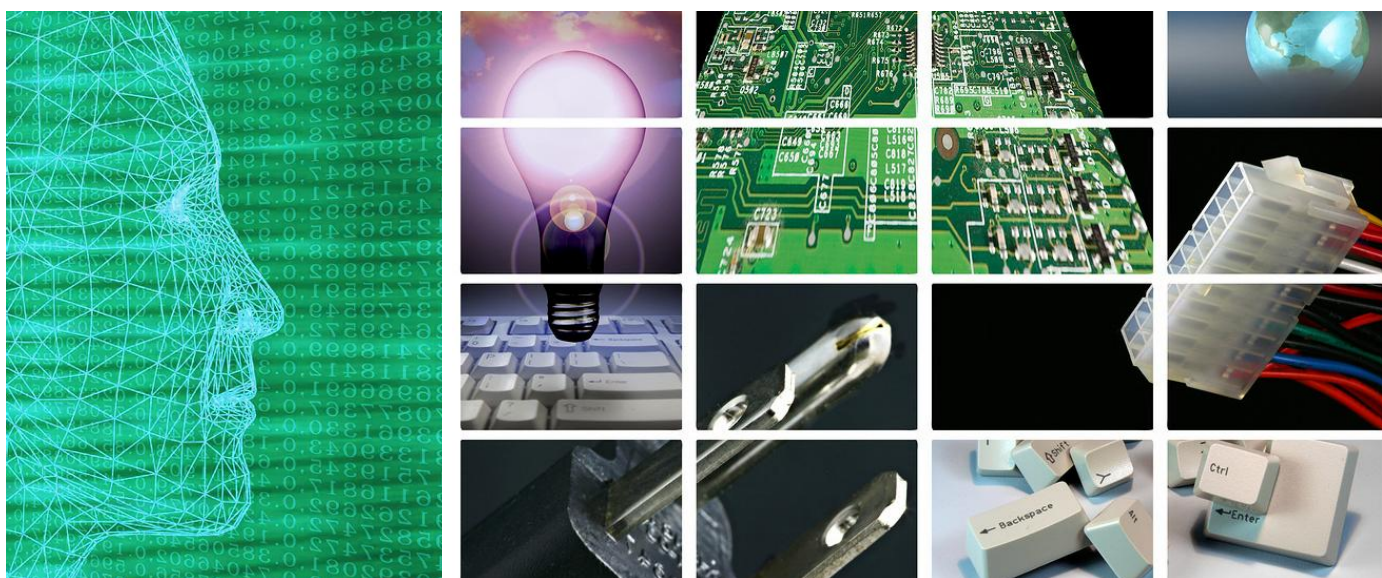




ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - ВАРНА
TECHNICAL UNIVERSITY OF VARNA

Година XIV, Брой 2/2016

КОМПЮТЪРНИ НАУКИ И ТЕХНОЛОГИИ



Bulgaria Communications Chapter

Faculty of Computing & Automation

COMPUTER SCIENCE AND TECHNOLOGIES

*Четвърта научна конференция с международно участие
"Компютърни науки и технологии"
30 септември - 1 октомври, 2016 г. Варна, България*

Year XIV, Number 2/2016

Компютърни науки и технологии

Издание

на Факултета по изчислителна техника и
автоматизация
Технически университет - Варна

Редактор: доц. д-р Ю. Петкова
Гл. редактор: доц. д-р Н. Николов

Редакционна колегия:

проф. д.н. Л. Личев (Острава)
проф. д.н. М. Илиев (Русе)
проф. д-р М. Лазарова (София)
доц. д-р П. Антонов (Варна)
доц. д-р Р. Димова (Варна)
доц. д-р Х. Вълчанов (Варна)
доц. д-р В. Божикова (Варна)
доц. д-р Н. Николов (Варна)

Печат: ТУ-Варна

За контакти:

Технически университет - Варна
ФИТА
ул. „Студентска” 1, 9010 Варна,
България
тел./факс: (052) 383 320
e-mail: ned.nikolov@tu-varna.bg
yulka.petkova@tu-varna.bg

ISSN 1312-3335

Computer Science and Technologies

Publication

of Computing and Automation Faculty
Technical University of Varna

Editor: Assoc. Prof. Y. Petkova, PhD
Chief Editor: Assoc. Prof. N. Nikolov, PhD

Advisory Board:

Prof. L. Lichev, DSc (Ostrava)
Prof. M. Iliev, DSc (Ruse)
Prof. M. Lazarova, PhD (Sofia)
Assoc. Prof. P. Antonov, PhD (Varna)
Assoc. Prof. R. Dimova, Varna (Varna)
Assoc. Prof. H. Valchanov, PhD (Varna)
Assoc. Prof. V. Bozhikova, PhD (Varna)
Assoc. Prof. N. Nikolov, PhD (Varna)

Printing: ТУ-Varna

For contacts:

Technical University of Varna
Faculty of Computing and Automation
1, Studentska Str., 9010 Varna,
Bulgaria
Tel/Fax: (+359) 52 383 320
e-mail: ned.nikolov@tu-varna.bg
yulka.petkova@tu-varna.bg

ISSN 1312-3335

СПОНСОРИ НА КОНФЕРЕНЦИЯТА

 http://www.researchmetrics.com	The complete ecosystem of tools and services for your research business™
 http://www.sirma.com	Сирма Груп Холдинг разработва експертни софтуерни решения и извършва системна интеграция. Натрупан опит в мобилните технологии, електронното управление (e-government), банкирането и застраховането, семантичните технологии и др.
 http://www.ardes.bg	Ardes.bg предлага на клиентите си: консултация при избора на информационно оборудване и софтуер; доставка на компютърни системи и компоненти, модули и софтуер; изграждане на компютърни и телевизионни локални мрежи и системи; изграждане на цялостни информационни решения за бизнеса; абонаментна поддръжка и сервиз на компютърна техника.
 http://www.eurorisksystems.com	EuroRisk Системи ООД предоставя софтуерни системи и услуги за банки, застрахователни и финансови институции и други доставчици на финансови услуги.
 http://www.asicdepot.com	ASIC Depot предоставя разширени услуги по проектиране, дизайн и верификация на мрежови процесори, мултимедия, контролери и периферия, мобилни приложения, софтуер за проверка и др.
 http://www.bg.adastragrp.com	АДАСТРА предлага информационен мениджмънт - пълна гама услуги, насочени към рационализация и ускоряване на бизнес процеси или обновяване на фирмена технологична инфраструктура.
 http://ciscoacademy.tu-varna.bg	Локалната Cisco Академия при ТУ-Варна провежда курсове по най-новите учебни програми на Cisco Networking Academy - Cisco CCNA Exploration и Cisco CCNA Security на български и английски език.
 http://www.taxback.com	Taxback Group предлага пакет от услуги, сред които специализирани данъчни декларации, финансови и туристически консултантски услуги за физически лица и корпоративни клиенти.
 www.411marketing.bg	Екип от амбициозни млади хора, които помагат на клиентите си да промотират своя бизнес. Зад гърба си имат успешно сътрудничество с над 32 000 малки и средни бизнеси както от България, така и от чужбина.

СБОРНИК ДОКЛАДИ

*Четвърта научна конференция с международно участие
"Компютърни науки и технологии"
30 септември – 1 октомври, 2016 г.
Варна, България*



*Fourth Scientific International Conference
Computer Sciences and Engineering
30 September – 1 October, 2016
Varna, Bulgaria*

PROCEEDINGS

Организационен комитет	Organizing committee
Председател: Елена Рачева Членове: Христо Вълчанов Виолета Божикова Марияна Стоева Юлка Петкова Милена Карова Гинка Маринова Венета Алексиева Венцислав Николов Ивайло Пенев Нели Калчева Мая Тодорова Антоанета Иванова	Chairman: Elena Racheva Members: Hristo Valchanov Violeta Bozhikova Mariana Stoeva Yulka Petkova Milena Karova Ginka Marinova Veneta Aleksieva Ventsislav Nikolov Ivaylo Penev Nely Kalcheva Maya Todorova Antoaneta Ivanova

Програмен комитет	Programme committee
Председател: Недялко Николов – декан на Факултета по изчислителна техника и автоматизация Членове: Надежда Рускова Райчо Иларионов Ангел Смрикаров Анатолий Антонов Радослав Вробел Никола Николов Розалина Димова Лъчезар Личев Марта Сийбауер Пламенка Боровска	Chairman: Nedyalko Nikolov – Dean of Computing and Automation Faculty Members: Nadezhda Ruskova Raycho Ilarionov Angel Smrikarov Anatoliy Antonov Radoslav Vrobel Nikola Nikolov Rosalina Dimova Lachezar Lichev Marta Seebauer Plamenka Borovska

СЪДЪРЖАНИЕ

CONTENTS

	АВТОМАТИКА И КОМПЮТЪРНИ СИСТЕМИ ЗА УПРАВЛЕНИЕ		AUTOMATION AND COMPUTER CONTROL SYSTEMS	
1	АЛГОРИТМИ ЗА УСКОРЯВАНЕ НА ИЗЧИСЛЕНИЯТА В МЕТОДА ЗА ДЕТЕКТИРАНЕ НА КОНТУРИ НА ДЖОН КЕНИ ПРИ ИМПЛЕМЕНТИРАНЕ В FPGA <i>Димитър Кромичев</i>	9	ALGORITHMS FOR FAST CANNY EDGE DETECTION COMPUTATIONS TO BE IMPLEMENTED ON FPGA <i>Dimitre Kromichev</i>	1
2	СИМУЛИРАНЕ В SCILAB НА FPGA ОРИЕНТИРАНА РЕАЛИЗАЦИЯ НА МЕТОДА ЗА ДЕТЕКТИРАНЕ НА КОНТУРИ НА ДЖОН КЕНИ <i>Димитър Кромичев</i>	17	SCILAB SIMULATED CANNY EDGE DETECTION REALIZATION TO BEIMPLEMENTED ON FPGA <i>Dimitre Kromichev</i>	2
3	ОСНОВНИ ПАРАМЕТРИ ЗА БЪРЗОДЕЙСТВИЕ НА ИЗЧИСЛЕНИЯТА В БАЗИРАНИЯ НА FPGA МЕТОД ЗА ДЕТЕКТИРАНЕ НА КОНТУРИ НА ДЖОН КЕНИ <i>Димитър Кромичев</i>	25	CAPITAL PARAMETERS OF FAST FPGA BASED CANNY EDGE DETECTION COMPUTATIONS <i>Dimitre Kromichev</i>	3
4	ФИЛТРИРАНЕ НА ЗАШУМЕНИ СИГНАЛИ ПОСРЕДСТВОМ УЕЙВЛЕТ ФУНКЦИИ <i>Марияна Тодорова, Ренета Първанова</i>	32	FILTRATION OF NOISY SIGNALS BY WAVELET FUNCTIONS <i>Mariana G. Todorova, Reneta D. Parvanova</i>	4
5	СРАВНИТЕЛЕН АНАЛИЗ НА РЕКУРСИВНИ МЕТОДИ ЗА ОЦЕНЯВАНЕ НА ПАРАМЕТРИ В СИСТЕМА ЗА УПРАВЛЕНИЕ НА ПОСТОЯННО ТОКОВ ДВИГАТЕЛ <i>Иван Григоров, Наско Атанасов</i>	39	COMPARATIVE ANALYSIS OF MORE COMMONLY USED RECURSIVE METHODS FOR PARAMETER ESTIMATION IN DC MOTOR CONTROL SYSTEM <i>Ivan V. Grigorov, Nasko R. Atanasov</i>	5

	СТУДЕНТИ И ДОКТОРАНТИ		STUDENT SESSION	
1	НЕРЕШИМИ И НЕРЕШЕНИ ЗАДАЧИ <i>Ирена Георгиева</i>	48	UNDECIDABLE AND UNSOLVED PROBLEMS <i>Irena Y. Georgieva</i>	1
2	АЛГОРИТЪМ ЗА ПЪЛНО ТОПОЛОГИЧНО СОРТИРАНЕ НА АЦИКЛИЧЕН ГРАФ <i>Константин Георгиев</i>	55	ALGORITHM FOR FULL TOPOLOGICAL SORTING OF AN ACYCLIC GRAPH <i>Konstantin S. Georgiev</i>	2
3	СИСТЕМА ЗА СИМУЛИРАНЕ НА БОТНЕТ DOS АТАКИ <i>Юлия А. Алексиева</i>	60	BOTNET DOS ATTACK SIMULATION SYSTEM <i>Yuliya A. Aleksieva</i>	3
4	КАЧЕСТВО НА УСЛУГИТЕ В ОБЛАЧНИ ИЗЧИСЛЕНИЯ <i>Сияна Х. Вълчанова</i>	68	QUALITY OF SERVICES IN CLOUD COMPUTING <i>Siana H. Valchanova</i>	4
5	АНАЛИЗ И ИЗСЛЕДВАНЕ НА АЛГОРИТМИ ЗА НАМИРАНЕ НА ПЪТ В СРЕДА <i>Никола В. Василев, Милена Н. Карова</i>	74	ANALYSIS AND RESEARCH OF PATHFINDING ALGORITHMS <i>Nikola V. Vasilev, Milena N. Karova</i>	5
6	СЪВРЕМЕННИ СОФТУЕРНИ ПАКЕТИ ЗА ОБРАБОТКА НА ДАННИ В ПСИХОЛОГИЯТА. SPSS И R - СИЛНИ СТРАНИ И ОГРАНИЧЕНИЯ <i>Славена А. Лазарова, Вяра Н. Колева - Ефремова</i>	82	CONTEMPORARY SOFTWARE PACKAGES FOR STATISTICAL ANALYTICS IN PSYCHOLOGY. SPSS AND R – PROSPECTS AND CONSTRAINTS <i>Slavena A. Lazarova, Vyara N. Koleva - Efremova</i>	6
7	IPV6 АДРЕС ОТ VIN НОМЕР <i>Радослав Вробел, Христо Вълчанов</i>	91	IPV6 ADDRESS FROM VIN NUMBER <i>Radosław Wróbel, Hristo Valchanov</i>	

СЕКЦИЯ 3 АВТОМАТИКА И КОМПЮТЪРНИ СИСТЕМИ ЗА УПРАВЛЕНИЕ

*Четвърта научна конференция с международно участие
"Компютърни науки и технологии"
30 септември – 1 октомври, 2016 г.
Варна, България*



*Fourth Scientific International Conference
Computer Sciences and Engineering
30 September – 1 October, 2016
Varna, Bulgaria*

SECTION 3 AUTOMATION AND COMPUTER CONTROL SYSTEMS

ALGORITHMS FOR FAST CANNY EDGE DETECTION COMPUTATIONS TO BE IMPLEMENTED ON FPGA

Dimitre Zh. Kromichev

Abstract. The paper focuses on presenting and analyzing four new algorithms addressing the goal of speed enhanced Canny edge detection computations intended to be implemented on FPGA. A thorough description of each algorithm's peculiarities is set forth. The algorithms are scrutinized in terms of their applicability, computational reliability and speed characteristics. Their technological capabilities of efficiently avoiding speed eroding computational approaches taking into account the FPGA functionalities are exposed in detail. Satisfying the peremptory demand for mathematical accuracy and correctness of results pinpoints the feasibility of the proposed algorithms as tangible tools for optimizing the speed of Canny computations on FPGA.

Key words: Canny edge detection, FPGA, speed, algorithm

1. Introduction

Canny edge detection is a complicated low level digital image processing technology intended to be used in software. The FPGA hardware implementation imposes two basic requirements: correctness and speed. Speed depends upon several factors, among them of tremendous importance being the algorithms calculations are realized through. FPGA functionalities and characteristics should be taken into account by the computational algorithms in terms of utilizing those favourable to fast executions whereas avoiding operations and methods serving as bottlenecks.

Canny edge detection includes five modules: Gaussian smoothing, computing the orthogonal gradients, computing gradient magnitude and direction, non-maximum suppression, hysteresis thresholding. With respect to speed optimization, of particular interest are the linear weighed average filtering integer arithmetic effectiveness focused on division, gradient direction computations without resorting to inverse tangent function, dynamic calculation of high and low threshold values, and hysteresis thresholding edge detection approach.

Focusing on Gaussian smoothing, in the literature, presented are several methods aimed at improving the speed performance or entirely avoiding the conventional integer division algorithms. In [8] proposed is a method for realizing integer division on FPGA based on Look-up tables, without employing additions, subtractions and multiplication by the inverse of the divisor. In [12], proposed is an FPGA-based computational architecture in which integer division is substituted by multiplication and shifting. In [11], division is substituted with multiplication with the precomputed reciprocal value of the divisor. In [15], multiplication of the dividend by the inverse of the divisor is used instead of conventional integer division. In [14], proposed is a method for integer division based on subtraction. In the speed domain of FPGA calculations, all these methods share a common flaw - the use of register logic and iterations.

To avoid the FPGA implementation of inverse tangent function, proposed are techniques relying entirely upon the vertical or horizontal values' being positive or negative [13]. This approach does not provide the necessary correctness of gradient direction calculations.

For the high and low thresholds, in [7], uniformly quantized gradient magnitude histograms are computed on overlapped blocks. An intermediate classification threshold is calculated based on a set of pixels with gradient magnitudes larger than a defined value. The high threshold is computed on the basis of the histograms. In [9] [10], the OTSU method is applied to adaptively calculate high and low thresholds by basically splitting all the image pixels into two classes, and computing the best threshold value through the variance maximum value between the two classes. The approach here is to select a threshold which minimizes the within-class variance or maximizes the between-

class variance. In [1], a low-complexity 8-step non-uniform gradient magnitude histogram is used to compute block-based hysteresis thresholds. These approaches share the flaw of computing the high and threshold values in a sequential mode, thus impeding the pipelining effectiveness and diminishing speed.

For hysteresis thresholding, in the literature described are few approaches addressing FPGA implementation. Hysteresis thresholding relies on a moving window, two comparators, FIFO buffer and OR gates [13] [16]. With respect to speed the significant flaw is the reliance on recursive computations.

The objective of this paper is to present four new algorithms aimed at speeding up the Canny edge detection computations on FPGA. These algorithms are focused on the Gaussian filtering function, gradient direction calculation, dynamic computation of high and low thresholds, and hysteresis thresholding. The task is to describe in detail the sequence of steps, to thoroughly analyze the mathematical reliability, to expose the characteristics, and to point out the applicability of the proposed algorithms to the goal of speed optimized Canny computations in compliance with the specifics of FPGA implementation. The targeted hardware is Altera FPGAs. Relevant to the analyses and conclusions arrived at in this paper are only gray-scale images.

2. Algorithm for Gaussian smoothing

2.1. Gaussian smoothing computations

As a representative of the linear spatial filtering defined as a $Z \times Z$ neighbourhood operation (Z is an odd number and $Z \geq 3$), Gaussian smoothing is:

$$v(m, n) = \frac{1}{s} \sum_{k=-w}^w \sum_{l=-w}^w g(k, l) u(m + k, n + l) \quad , \quad (1)$$

where

$u(m, n)$ is the input image,

$v(m, n)$ is the filtered image,

$w = \frac{Z-1}{2}$, $w = \{1, 2, \dots\}$,

$g(k, l)$ is the Gaussian filter,

s is the sum of all coefficients in the mask; it is referred to as *normalization factor*; the coefficients are integers based on the binominal series.

On the basis of (1), indispensable for computing the Gaussian function are three integer arithmetic operations: multiplication, addition, and division. Multiplication and addition are two of the three top speed arithmetic operations on Altera FPGAs whereas division is not only among the slowest in terms of maximum frequency of execution [6] , but also provides enormous delays [2],[3],[4],[5]. Consequently, for the Gaussian filtering to be effective in terms of speed division has to be tackled in the first place.

2.2. Fast Gaussian smoothing algorithm

The goal of the proposed algorithm is to replace the conventional integer division and guarantee maximum execution speed. The technology of computations implemented by the proposed algorithm focuses on the following mandatory items:

- 1) A hundred percent mathematical correctness and exactness of results. No approximation whatsoever is admissible.
- 2) Utilizing only operations rated among the five fastest on Altera FPGAs.
- 3) Capability of being applied to Gaussian filters of any size.

- 4) The Gaussian mask coefficients can be considerably large in value. Therefore, the dividends calculated on the basis of multiplying a coefficient and the neighbourhood pixel under it can significantly exceed 2^{16} . Consequently, the peremptory demand is defined as: whatever the value of the dividend, the algorithm should guarantee no susceptibility whatsoever to input data width driven speed reduction.
- 5) In order to be capable of approximating the upper speed limit of exactly one clock cycle for the correct quotient to be calculated, the algorithm has to be based on as many as two operations.
- 6) The Gaussian mask coefficients are to be changed appropriately as to mathematically guarantee the application of bit slicing operation as optimal in all respects to accomplish the maximum speed goal.
- 7) Integer division requiring rounding, the arithmetic operation that proves to be indispensable to the algorithm is addition.
- 8) The algorithm does not admit speed erosion in terms of using iterations and registers.

The proposed integer division algorithm encompasses the following sequence of steps:

- 1) Defining the positive integer that serves as divisor.
- 2) Experimentally, an appropriate value equal to a power of 2 is selected. This value is divided by the integer number from step 1) .
- 3) Only the integer part of the division result calculated in step 2) is taken.
- 4) The number that will be divided is multiplied by the integer number from step 3) .
- 5) From the result calculated in step 4), a sequence of bits (starting with LSB and going leftwards) is dropped out. The number of the dropped out bits is equal to the number of the power of 2 selected in step 2).
- 6) The value of MSB of the bits that are dropped out in step 5) is added to the value represented by the remaining bits in the multiplication result calculated in step 4) after the sequence of bits is dropped out in step 5). The positional number of this bit in the multiplication result calculated in step 4) is equal to:

$$[\text{the number of the power of 2 selected in step 2)}] - 1.$$

- 7) The result of addition in step 6) is the needed quotient of the integer division operation.

This algorithm being focused on computing the Gaussian function, the positive integer which serves as divisor is a filter specific constant – the normalization factor. It has to be different from a power of 2. If the divisor has a value equal to some power of 2, only the operation bit slicing (steps 5), 6), and 7) of the algorithm) is relevant to the calculations. For the bit slicing to be both reliable in terms of Gaussian filtering, and mathematically correct and exact with respect to results, each of the filter mask coefficients is to be multiplied by the modification constant calculated in step 3) prior to starting Gaussian smoothing computations. Thus, the Gaussian filter employed in smoothing is practically replaced with a new filter modified only in terms of coefficients but unchanged as to normalization factor. Utilizing modified Gaussian mask coefficients has significant impacts on speed with respect to two facts:

- 1) The combination of only two fastest operations – bit slicing and addition to calculate a quotient of guaranteed correctness, defines the proposed algorithm as the optimal tool for performing fast integer division.
- 2) The modification constant having been calculated in advance, all the modified coefficients are ready for use before the start of Gaussian smoothing execution. Consequently, no excessive multiplications are required to perform the proposed integer division algorithm.

3. Algorithm for gradient direction

3. 1. Gradient direction computations

The mathematical expression for gradient direction calculation is:

$$G_D = \tan^{-1} \left(\frac{G_y}{G_x} \right), \quad (2)$$

where

G_D is the gradient direction,

G_y is the y gradient, and $G_y \in [-255, 255]$,

G_x is the x gradient, and $G_x \in [-255, 255]$.

Inasmuch as there are eight surrounding pixels in the 3x3 neighborhood, there are four possible axes determining the gradient directions. Consequently, all the calculation results are reduced to four values: 0, 45, 90, 135.

Calculating the gradient direction by applying expression (2) has the following inauspicious aspects:

- 1) FPGA implementation of inverse tangent function is computationally expensive, and therefore leads to substantial delays.
- 2) Integer division is among the slowest operations on Altera FPGAs..
- 3) The divisor G_x can be equal to 0. In that case calculating the gradient direction is paractically irrelevant, and, consequently, additional computational conditions should be introduced to ensure the correctness of results.

Executing (2) on FPGA being impropitious to optimized speed performance, a new approach is to be applied.

3. 2. Fast gradient direction algorithm

The goal of the proposed algorithm is to ensure maximum execution speed taking into account the characteristics of gradient direction calculation. The accomplishment requires that the focus be on the following:

- 1) The signs of G_y and G_x are of capital importance to determie the closed intervals each calculated gradient directon value is confined to.
- 2) Integer division is incompatible with speed optimization.
- 3) It is the ratio between dividend (G_y) and divisor (G_x) that defines the result of computations in terms of one of the four values: 0, 45, 90, 135.
- 4) Employing the smallest possible number of reference points to define the required closed intervals.
- 5) The values of the reference points should be the closest possible approximations to the targeted angles around the unit circle to guarantee maximum mathematical correctness.
- 6) Using the minimum amount of integer arithmetic operations.
- 7) Excluding comparison functions on the basis of their being relatively slow and capable of serving as bottlenecks to speed performance.
- 8) Utilizing only the fastest integer arithmetic on Altera FPGAs.

The proposed gradient direction algorithm includes the following sequence of steps:

- 1) Determining all the possible combinations between the signs of G_y and G_x . Their number is 2^2 .
- 2) The combinations from step 1) are divided into two sets.
- 3) One of the sets from step 2) includes the following relations:

$$G_y > 0 \ \& \ G_x > 0$$

$$G_y < 0 \ \& \ G_x < 0 . \quad (3)$$

4) The other set defined in step 2) includes the following relations:

$$\begin{aligned} G_y > 0 \ \& \ G_x < 0 \\ G_y < 0 \ \& \ G_x > 0 . \end{aligned} \quad (4)$$

5) Determining the two angles from the unit circle that define four angular intervals in which the x axis, the y axis, the positive diagonal, and the negative diagonal are bisectrices. These angles are 22.5° and 67.5° .

6) Determining the most appropriate approximation for the angle 22.5° . Taking into account that $\tan^{-1}\left(\frac{2}{5}\right) = 21.8014094^\circ$, this is the fraction $\frac{2}{5}$. Then, determine the most appropriate approximation for the angle 67.5° . With respect to $\tan^{-1}\left(\frac{5}{2}\right) = 68.1985905^\circ$, this is the fraction $\frac{5}{2}$.

7) On the basis of the set of sign relations defined in step 3), the gradient direction values are calculated as follows:

$$\begin{aligned} \text{If } (|G_x|^2 - |G_y|^5) &\geq 0 & G_D &= \mathbf{0} \\ \text{If } [(|G_x|^2 - |G_y|^5) < 0] \ \& \ [(|G_x|^5 - |G_y|^2) > 0] & G_D &= \mathbf{45} \\ \text{If } (|G_x|^5 - |G_y|^2) &\leq 0 & G_D &= \mathbf{90} . \end{aligned} \quad (5)$$

8) On the basis of the set of sign relations defined in step 4), the gradient direction values are calculated as follows:

$$\begin{aligned} \text{If } (|G_x|^2 - |G_y|^5) &\geq 0 & G_D &= \mathbf{0} \\ \text{If } [(|G_x|^2 - |G_y|^5) < 0] \ \& \ [(|G_x|^5 - |G_y|^2) > 0] & G_D &= \mathbf{135} \\ \text{If } (|G_x|^5 - |G_y|^2) &\leq 0 & G_D &= \mathbf{90} . \end{aligned} \quad (6)$$

In Canny edge detection gradient direction and gradient magnitude computations are executed in parallel. Taking into account that gradient magnitude requires only the fastest integer arithmetic operation on Altera FPGAs – addition, the optimal speed characteristics provided by the proposed algorithm are a feasible tool to reduce the considerable difference, measured in clock cycles, between gradient direction and gradient magnitude calculations, thus preventing bottlenecks to speed performance.

4. Algorithm for high and low thresholds

4. 1. High and low thresholds computations

A key characteristic of Canny edge detection is the utilization of two thresholds. Calculating them through using the image histogram, although being a most reliable approach in terms of statistical accuracy, is incompatible with the speed optimization goal on the basis of the following:

- 1) All the image pixels being required for immediate processing, the pipelining is practically ineffective until the histogram computation and the subsequent analysis of the pixel values have been completed.
- 2) The time, in terms of clock cycles, during which the pipelining is blocked, is directly proportional to the size of the image.

On the other hand, the most computationally inexpensive way for the two thresholds to be put into use is to select in advance two values that are to be applied to all the images being edge detected. As beneficial to speed performance as it is, this technique is characterized by a substantial unreliability with different images.

All that being taken into account, for both speed and accuracy to be appropriately addressed, a new approach needs to be applied.

4. 2. Fast high and low thresholds algorithm

The goal of the proposed algorithm is to provide a speed optimized computing of the high and low threshold values through satisfying the following computational requirements:

- 1) Dynamic computation of the thresholds.
- 2) Maximum reliability of results.
- 3) Adaptiveness to the different noise conditions presented by the different images.
- 4) Working in parallel with the Canny computational modules.
- 5) Avoiding the blocking of pipelining.
- 6) Utilizing only the fastest Altera FPGAs' arithmetic.
- 7) Avoiding iterative computations.

The algorithm covers the following sequence of steps:

- 1) Defining the range of consecutive positive values the high threshold calculation will be based upon. It is within a closed interval whose minimum and maximum values are selected in such a way that the difference between the maximum decremented by 1 and the minimum is a positive number whose factor is 7. The minimum is as many as three times smaller than the largest pixel value.
- 2) The closed interval from step 1) is divided into equal subintervals whose number is as many as three times bigger than 7. Each subinterval encompasses 7 consecutive values, and is associated with one reference value which is equal to the fourth out of the seven consecutive values contained in this subinterval.
- 3) Each of the reference values defined in step 2) is paired with a table value that is 2.8 times smaller than the reference value.
- 4) Defining a set of counters. Each counter is related to one of the subintervals from step 2).
- 5) Starting with image pixel #1 being processed, each pixel is assessed with respect to its value falling/not falling within one of the subintervals from step 2). If the pixel value is within an interval, the counter associated with this particular interval is incremented by 1.
- 6) With all the pixels gone through the assessment procedure from step 5), the values in each of the 16 counters defined in step 4) are compared to select the largest among them. On the basis of this largest value, one of the subintervals, and the reference value associated with it, as defined in step 2), is further selected. This reference value is the high threshold.
- 7) The table value defined in step 3) that is paired with the reference value as calculated in step 6) is selected. This table value is the low threshold.

5. Algorithm for hysteresis thresholding

5. 1. Hysteresis thresholding computations

The high threshold $T1$ and the low threshold $T2$ having been calculated, the hysteresis thresholding's being rather costly in terms of computations is focused mainly upon the connectivity/non-connectivity determination with respect to $T1$. Processing the pixels sequentially on one-by-one basis, the way they are pipelined into the hysteresis thresholding module from the preceding non-maximum suppression module, to track connectivity leads to:

- 1) Excessive repetitive calculations.

- 2) Recursive computations.
- 3) Traversing the entire image a number of times to determine the possible indirect connectivity of a pixel to $T1$.
- 4) Significant delays.

Inasmuch as the aforementioned are incongruous with the speed optimization goal, a different approach has to be employed.

5. 2. Fast hysteresis thresholding algorithm

The goal of the proposed algorithm is to guarantee optimal speed of determining the edge pixels through fulfilling the following requirements:

- 1) Diminishing repetitive calculations to the bare minimum.
- 2) Absence of recursive computations.
- 3) Computational operations performed only on pixels belonging to consecutive image rows.
- 4) Reducing the number of calculations one and the same pixel value participates in.
- 5) Employing two nested square neighbourhood windows to track connectivity.
- 6) Utilizing the fastest Altera FPGAs' integer arithmetic.
- 7) Single traverse of the entire image to define edge pixels.

The algorithm is presented by the following sequence of steps:

- 1) The value of each pixel is tested for being an edge (bigger or equal to $T1$), not being an edge (less or equal to $T2$), and being a possible edge (smaller than $T1$ but bigger than $T2$).
- 2) A 3x3 window centered around each pixel satisfying the condition for being a possible edge is applied. All the neighbouring pixels – maximum eight, are tested in reference to $T1$.
- 3) If there is a value bigger than $T1$, the central pixel from step 2) is defined as an edge and its coordinates in the image matrix are stored in a separate memory buffer to avoid redundant double check of up to five magnitudes when the window moves on to the next pixel.
- 4) If none of the neighbouring magnitudes is bigger than $T1$, but at least one falls between $T1$ and $T2$, only peripheral magnitudes of a 5x5 window are checked for values bigger than $T1$.
- 5) If such a value is available, then it is an edge pixel.
- 6) If one or more of these sixteen peripheral magnitudes in the 5x5 window falls between $T1$ and $T2$, then its (their) coordinates are stored in another memory buffer.
- 7) If such a magnitude is not available, then the central pixel from 2) is not an edge.
- 8) The procedure encompassing steps 1) through 7) is repeated for the next pixel in the image row, taking into account that from the second pixel downwards the contents of the two memory buffers are constantly checked. The second buffer contains those pixels which can eventually be determined as edge pixels on the basis of indirect connectivity.

6. Conclusions

Presented are four new algorithms targeting the speed optimization of the Canny computations. They are focused on Gaussian smoothing, gradient direction, high and low thresholds, and hysteresis thresholding. A thorough description of each algorithm's characteristic is set forth. The algorithms are scrutinized in terms of their applicability, computational reliability and functionality in the speed domain. Exposed in detail is the feasibility of their implementation on Altera FPGAs aimed at efficiently avoiding speed eroding computational approaches. The algorithms' providing optimal speed performance serves as a tangible tool for achieving a speed enhanced FPGA based realization of Canny edge detection.

References

- [1]. Aasiya anjum, Sanjay asutkar. FPGA Implementation of Efficient Edge Detection Using Canny Algorithm, International Journal on Recent and Innovation Trends in Computing and Communication, 2015, Vol. 3 (2), pp. 20-22
- [2]. Altera Corporation. Cyclone IV Device Handbook. 2014, Volume 1
- [3]. Altera Corporation. Cyclone V Device Handbook. 2014, Volume 1
- [4]. Altera Corporation. Stratix III Device Handbook, 2014, Volume 1
- [5]. Altera Corporation. Stratix IV Device Handbook. 2014, Volume 1
- [6]. Altera Corporation. Integer Arithmetic IP Cores User Guide, 2014
- [7]. Chandrashekar N.S., K.R Nataraj. Design and Implementation of a Modified Canny Edge Detector Based on FPGA, International Journal of Advanced Electrical and Electronics Engineering, (IJAEEE). 2013, Vol. 2 (1), pp. 16-21
- [8]. Dinechin, F., L Didier . Table-Based Division by Small Integer Constants. Reconfigurable Computing: Architectures, Tools and Applications. Lecture Notes in Computer Science, 2012, Volume 7199, pp. 53-63
- [9]. Gao Jie, Liu Ning. An improved adaptive threshold canny edge detection algorithm. IEEE Transactions, International Conference on Computer Science and Electronics Engineering (ICCSEE). 2012, Vol. 1, pp.164-168
- [10]. Mei Fang, Guang Xue Yue, Qing Cang Yu. The Study on An Application of Otsu Method in Canny Operator, Proceedings of the International Symposium on Information Processing (ISIP'09), 2009, pp.109-112
- [11]. Moller, N., T. Granlund. Improved division by invariant integers, *IEEE Transactions on Computers*, 2011 (.60), pp. 165-175
- [12]. Rao, D. V., S. Patil, N. A. Babu, N. A., V. Muthuhumar. Implementation and Evaluation of Image Processing Algorithms on Reconfigurable Architecture using C-based Hardware Descriptive Languages, International Journal of Theoretical and Applied Computer Sciences, 2006, Vol. 1 (1), pp. 9-34
- [13]. Rupalatha, T., C. Leelamohan, M. Sreelakshmi, Implementation of distributed Canny edge detection on FPGA. International Journal of Innovative Research in Science, Engineering and Technology. 2013, Vol. 2, (7), pp. 2618-2614
- [14]. Takagi, N., S. Kadowaki, K Takagi. A Hardware Algorithm for Integer Division, IEICE Transactions on Electronics, Communications and Computer Sciences. 2006, Vol. E89-A, pp. 2874-2881
- [15]. Werbrouc, A. E., D. Cavagnino. Efficient Algorithms for Integer Division by Constants Using Multiplication. The Computer Journal. 2008. Vol. 51 (4), pp. 470-480
- [16]. Yang Yongjin Zhou Xinmei Xiang Zhongfan, Research of Image Pre-processing Algorithm Based on FPGA, International journal on smart sensing and intelligent systems. 2013, Vol. 6, (4), pp. 101-106

For contacts:

Ass. Prof. Dimitre Zh. Kromichev
Department of Marketing and International Economic Relations
University of Plovdiv
e-mail: dkromichev@yahoo.com

SCILAB SIMULATED CANNY EDGE DETECTION REALIZATION TO BE IMPLEMENTED ON FPGA

Dimitre Zh. Kromichev

Abstract. This paper is focused on testing the mathematical correctness and algorithmic effectiveness of a proposed new Canny edge detection realization intended to be implemented on FPGA. Scilab is the tool used for the software simulation. The purpose is to assess the reliability of the proposed computational flow in terms of appropriateness and ability to fulfil the task of properly detecting image edges. The steps in the simulation program are described in detail. Taking into account the characteristics of the Scilab language, all the calculations in the Canny modules are organized accordingly as to thoroughly present the functionality and computational capabilities of the proposed approach. The results are analyzed and evaluated on the basis of edge detected real life images.

Key words: Canny edge detection, Scilab, simulation, FPGA

1. Introduction

Canny edge detection is a precise yet computationally complicated algorithm. As a representative of the programmable logic, FPGA has been steadily expanding its market share over the last two decades. One of the most beneficial aspects of Canny's being implemented on FPGA is speed. The latter is a function of appropriate and purposeful algorithmic organization of computations in each of the Canny modules, and is relevant only on the peremptory platform of mathematical accuracy and algorithmic effectiveness of calculations as reflected in the plausibility of detected edges. Therefore, in the first place it is obligatory to scrutinize the abilities of a proposed Canny realization to satisfy the demand of truthfulness with respect to edge mapped image qualities. This is basically accomplishable through simulating the proposed technology of Canny edge detection computations employing a highly specialized software tool.

Scilab is a widely used programming language of technical and scientific computing. The remarkable reliability in handling matrix based calculations being one of its most salient characteristics, Scilab is a much viable option for digital image processing. The latter benefits from the Scilab Image and Video Processing Toolbox's being able to support a number of formats, which enhances the feasibility of its practical use in a variety of applications. In the literature focused on the Scilab characteristics and utilization in the field of digital image processing, Canny edge detection has been traditionally paid much attention to. The focus is mainly on its computational modules and their realization in Scilab in terms of employing the language provided functions [1].

In the literature, simulated are several FPGA implemented Canny realizations. In [3], a 3x3 Gaussian filter is used for smoothing based on two FIFO buffers with a depth of one image row. The pixels are presented as arrays of values stored in a text file which is used as input during the smoothing process. The non-maximum suppression block consists of 3x3 window, selector, arithmetic, and comparator units. Window 3x3 unit holds the 8 pixel values at a time and feeds the input to the selector unit and the middle value is input to the comparator unit. In the selector block, angle is calculated at each pixel location simultaneously after the calculation of gradient of an image. Hysteresis thresholding relies on a moving window, two comparators, FIFO buffer and OR gates. In [2], described is a new realization of Canny using a low-complexity 8-bin non-uniform gradient magnitude histogram to compute block-based hysteresis thresholds. In [4], [5], a distributed Canny focused on significantly reducing memory requirements is implemented. The image is divided into blocks and each block is processed in parallel. The input image is divided into $m \times m$ overlapping blocks. The adjacent blocks overlap by $(L - 1)/2$ pixels for a gradient mask $L \times L$. For each block, only edges in the central $n \times n$ (where $n = m + L - 1$) non-overlapping region are

included in the final edge map. In [6], a novel distributed Canny computes edges of multiple blocks at the same time. An adaptive technique calculates the high and low thresholds of the entire image while only processing the pixels of an individual block. In [7], a novel implementation of Canny uses 4-pixel parallel computation. Five memory elements are utilized, each one storing the data of one image line. Double thresholding is executed by a double comparator with the data being stored in a specialized cache for the hysteresis thresholding with a size of $1 \times \text{image width} \times 2$ bits. The comparison is achievable as long as a second pass is executed with the pixels read in the opposite direction. The comparison is also executed for 4 pixels in parallel. The computed data from pass # 1 of hysteresis is still 2 bits for each pixel and it is stored in an external on-board memory. In [8], Gaussian filter being separable, the implementation of the 2-D convolution is achieved using row and column 1-D convolutions. For accessing all the pixels in the neighbourhood for non-maximum suppression two FIFO buffers are employed. For the two thresholds, an 8-step non-uniform quantizer is used to obtain the discrete histogram for each processed block. The hysteresis thresholds are computed at block level. In [9], proposed is a modified Canny adaptively computing the edge detection thresholds based on the block type and the local distribution of the gradients in the image block. The hysteresis high and low thresholds calculation, is modified to enable parallel block-level processing without degrading the edge detection performance. In [10], on the basis of the statistical fact that in gradient magnitude histograms after non-maximum suppression edge pixels present a series of smaller peaks wherein each peak defines a class of edges having similar gradient magnitudes, it is proposed that the high threshold of Canny should be selected between the largest peak and the second largest edge peak. A non-uniform quantizer to discretize the gradient magnitude histogram is proposed to compute the high threshold value. The quantizer needs to have more quantization levels in the region between the largest peak and the second largest peak, and few quantization levels in other parts.

The objective of this paper is to prove the mathematical correctness and algorithmic effectiveness of a new Canny edge detection realization intended to be implemented on FPGA. The task is to exhibit and scrutinize the proposed approach in terms of its major computational steps, characteristics, and edge detection capabilities. The accomplishment is based on simulating the proposed Canny realization functionalities in Scilab and demonstrating its image processing qualities through the resultant edge detected images. For the purpose, Scilab version 5.4.1 and Scilab Image and Video Processing Toolbox version 0.4.3 are employed. The targeted hardware is Altera FPGAs. The proposed Canny realization is intended to work with gray-scale images.

2. Canny edge detection flow of computations

As a low level digital image processing technology intended to be used in software, Canny edge detection covers the following sequence of computational stages:

- 1) Applying Gaussian filtering to the input gray-scale image.
- 2) Calculating the horizontal gradient.
- 3) Calculating the vertical gradient. Inasmuch as both stage 2) and stage 3) utilize the same values computed in stage 1), their positions in the succession are interchangeable with no effect on the mathematical correctness.
- 4) Calculating the gradient magnitude.
- 5) Calculating the gradient direction. As long as both stage 4) and stage 5) rely on the values computed in stages 2) and 3), the order of their execution can be changed with no influence on the reliability of the total flow of computations.
- 6) Determining the local maxima on the basis of magnitudes and directions.
- 7) Defining the high and low thresholds on the platform of the image magnitudes. The upward dependence of this stage is mandatory in case of employing histogram based analysis which requires the entire image statistics.

8) Applying the hysteresis thresholding to achieve the edge detected image.

3. The proposed Canny realization in Scilab

To test the mathematical correctness and algorithmic effectiveness of the proposed new FPGA orientated Canny realization a Scilab syntax based software simulation is employed. Unlike the FPGA based hardware implementation, which is capable of thoroughly utilizing the beneficial characteristics of parallel computations, the software approach to Canny allows only for consecutive calculations. Thus, the Scilab program used to simulate the proposed Canny realization encompasses the following sequence of essential steps:

- 1) Selecting a real life colour image using the specialized Scilab function.
- 2) Converting the colour image into a gray-scale image utilizing the specialized Scilab function.
- 3) Gaussian smoothing is applied to the image matrix. Weighted average filtering is a neighbourhood operation. Depending on the particular size of the Gaussian mask, filtering a single pixel requires a certain number of multiplications and additions to compute the numerator of a fraction whose denominator is the sum of all the coefficients in the mask. The Gaussian function essential conventional integer division operation is replaced by a procedure mathematically focused on selecting an exact number of bits in the numerator. The positions of these bits are determined by the Gaussian filter normalization factor, and an appropriate value of capital importance to the procedure - the modification constant. Being filter specific, this constant is calculated prior to starting the smoothing. Each Gaussian mask coefficient is multiplied by that constant, and then the result is multiplied in its turn by the square neighbourhood pixel positioned under the coefficient. To successfully complete the procedure in terms of calculating a result equal to the quotient of conventional integer division, it is necessary that a bit characterized by a special, determined by the value of normalization factor, position in the binary dividend should be added to the already selected succession of bits. This is equivalent to the rounding operation. The result of the procedure thus described presents the value of a Gaussian filtered pixel. Having already been filtered, each pixel is stored into a new, Gaussian smoothed, image matrix which is smaller than the input image matrix on the basis of dropping out a certain number of border pixels during the execution of neighbourhood operations.
- 4) The Gaussian smoothed image matrix values are used to calculate the horizontal gradient through employing the Sobel gradient filter for the x axis. Taking into account that three of the coefficients in the Sobel mask are positive, and three are negative, the utilized integer arithmetic peremptorily comprises addition and subtraction. The largest coefficient value being a power of 2, the required multiplication is executed in terms simulating shift left operation. Inasmuch as the largest calculated value for the horizontal gradient can be equal to the sum of positive (or negative) coefficients multiplied by the largest pixel value, avoidance of scaling is achieved through simulating the shift right operation followed by addition of the rounding value. Therefore, the possible horizontal gradient values are within the interval [-255,255]. They are stored into a matrix which is smaller than the matrix of Gaussian smoothed values, the border pixels cut out in the Sobel neighbourhood filtering process having been taken into account.
- 5) The Gaussian filtered image matrix values serve to compute the vertical gradient applying Sobel for the y axis. All the mathematics described in step 4) is applied here in exactly the same mode.
- 6) The results obtained in 4) and 5) are employed in the gradient magnitude calculation. Here, in the two-dimensional domain of digital image processing, the Manhattan distance is used to substitute for the Euclidean distance computation. The approximation in its mathematical appropriateness for the purposes of edge detection is applied on the basis of satisfying the requirement of utilizing only positive values. Therefore, the negative gradients from 4) and 5)

have to be converted into their positives prior to commencing the gradient magnitude calculations.

7) The horizontal and vertical gradients are involved in calculating the gradient direction. The significance of the values computed in steps 4) and 5) being positive or negative is reflected into determining the platform of sign relations in terms of defining a set of equal intervals representing the four quadrants within the unit circle. Then two reference values are utilized to optimally approximate the two angles of crucial importance to the mathematical correctness of gradient direction calculation. The essence of determining the four possible values of gradient direction implies two capital groups of computations. In case of both vertical and horizontal gradients being positive or negative, or vertical is positive, and horizontal is negative, or vertical is negative, and horizontal is positive, all the computations are focused on: a) the gradient direction value of 0 is calculated, if the doubled positive value of the horizontal gradient is bigger or equal to five times the positive vertical gradient; b) the gradient direction value of 45 is obtained, if the doubled positive value of the horizontal gradient is less than five times the positive vertical gradient and, at the same time, five times the positive horizontal gradient is bigger than the doubled positive vertical gradient. In case of both vertical and horizontal gradients being positive or negative, the gradient direction value of 90 is achieved when five times the positive horizontal gradient is less or equal to the doubled positive value of vertical gradient. In case of vertical gradient is positive and horizontal is negative, or vertical gradient is negative and horizontal is positive, the gradient direction value of 135 is calculated for the doubled positive value of the horizontal gradient being less than five times the positive vertical gradient and, at the same time, five times the positive horizontal gradient being bigger than the doubled positive vertical gradient.

8) Calculating the local maximum is based on a gradient magnitude from step 6) with its two adjacent magnitudes along the four angular axes determined in step 7). In fact, all the mathematics is realized on the platform of a square neighbourhood operation. If a magnitude is bigger than both of its adjacent magnitudes along the gradient direction, its value remains unchanged; otherwise it is replaced by 0. The result of every image matrix pixel's being subjected to non-maximum suppression processing is essential to the computation of the high and low threshold values. For the purpose, a definite number of counters are employed, each of them containing values within a certain interval. Every single magnitude calculated in the non-maximum suppression immediately increments one of the counters. The local maximum computations having been thoroughly realized, all the numbers contained in the counters are compared in order to select the largest, which is eventually defined as the high threshold. Then table values calculated on the basis of multiplying a reference value within each of the intervals by a predefined coefficient are used to determine the low threshold value.

9) Hysteresis thresholding requires that all the magnitudes calculated in step 8) should be compared with the high and low threshold in a rowwise mode. Inasmuch as the magnitudes bigger or equal to the high threshold are straightforwardly defined as edges while those less than or equal to the low threshold are replaced by 0, the essential focus of computations is on determining the direct/indirect connectivity in relation to the high threshold. This is achieved through a neighbourhood operation based on two nested windows of size 3x3 and 5x5. The 3x3 window is centered around each pixel meeting the requirement of being a possible edge. All the eight neighbouring pixels are compared to the high threshold. If a pixel is bigger or equal to the high threshold, then the central pixel is an edge. In case of all neighbouring magnitudes are less than high threshold but at least one falls between the high and low thresholds, only the peripheral magnitudes of the 5x5 window are tested for values bigger than the high threshold. Such a value being available, it is an edge. Then the sixteen peripheral magnitudes in the 5x5 window are scrutinized for falling between the high and low thresholds. If there is such a magnitude, its coordinates are stored in a special buffer containing those pixels which can eventually be

determined as edge pixels on the basis of indirect connectivity. If not, the central pixel in the 3x3 window is not an edge. The next pixel in the row is subjected to the same computational sequence until the whole image matrix has been covered.

10) The edge detected image is displayed using the specialized Scilab function.

4. Terms of simulation

To successfully simulate the proposed Canny realization in Scilab a concrete Gaussian filter has to be selected. Two basic facts are to be taken into account here:

1) The impact of the Gaussian smoothing on the edge detected image is considerable on the basis of the following dependence: the bigger the size of the Gaussian filter, the better the smoothing, and, at the same time, the worse the localization.

2) It is a practical procedure to choose the standard deviation of Gaussian distribution σ taking into account that the mask is to represent only those deviations of the Gaussian curve that have the most substantial effect on the weighted average filtering. Thus, in terms of functionality, the relevant Gaussian filter sizes are reduced to 3x3, 5x5, and 7x7.

The $\frac{1}{159}$ Gaussian filter (Fig. 1.) used in this simulation has $\sigma=1.4$ and size 5x5.

2	4	5	4	2
4	9	12	9	4
5	12	15	12	5
4	9	12	9	3
2	4	5	4	1

Fig. 1. Gaussian filter $\frac{1}{159}$

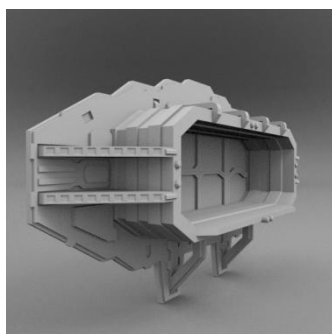
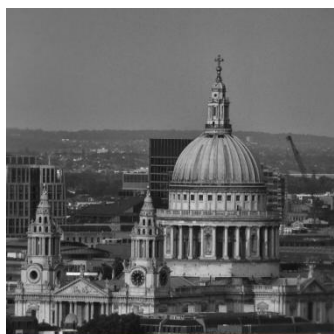
The modification constant calculated for that particular filter is 103. Therefore, each coefficient is to be multiplied by 103 as required by the Gaussian smoothing computations of the proposed Canny realization. Actually, the modified filter employed for the simulation, in terms of appearance, retains unchanged only its normalization factor as the sum of coefficients in the non-modified filter, which fulfils the essential task of the presented new technology of Gaussian filtering.

A set of real life colour images is utilized to demonstrate the edge detection characteristics of the proposed Canny realization. All the images are of size 512x512, and the format is JPG.

5. Simulation results

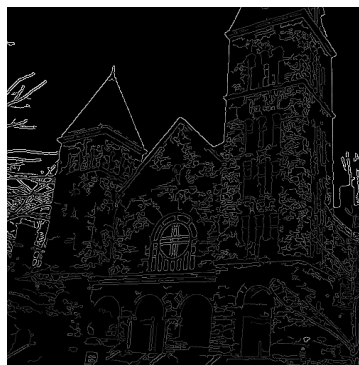
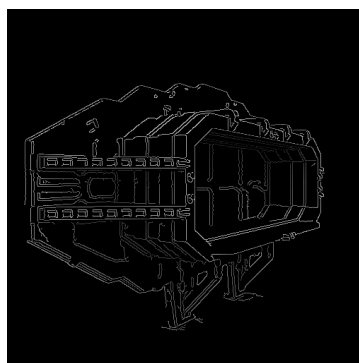
The mathematical and algorithmic functionalities of the proposed Canny realization are tested and proved by means of demonstrating and analyzing the output of the Scilab software implementation. The obtained simulation results are presented below:

1) The real life input images converted into gray-scale:



2) The edge detected images:





The edge analysis of the resultant processed images provides solid evidence of the proposed new Canny realization's being capable of guaranteeing high edge detection performance. The latter exhibits the achievement of the following targeted characteristics:

- 1) Dependability of the low level integer arithmetic in all respects.
- 2) Accuracy of throughput for a single consecutive step.
- 3) Veracity of calculations as the indispensable basis for edges' plausibility.
- 4) Appropriateness of the employed algorithms with respect to computational organization and effectiveness.

6. Conclusions

The focus of the paper is on testing the mathematical correctness and algorithmic effectiveness of a proposed new Canny edge detection realization intended to be implemented on FPGA. For the purpose, employed is a Scilab programming language based software simulation. A set of real life images is used to prove and analyze the image processing capabilities of the proposed Canny realization in terms of detected edges' qualities. The resultant edge mapped images show that the mathematical and algorithmic functionalities characterize the proposed realization as working and reliable with respect to the dependability of the low level integer arithmetic, the accuracy of every single consecutive step throughput, the edges' plausibility based upon veracity of calculations, and the employed algorithms' appropriateness to fulfil the task of correctness and effectiveness.

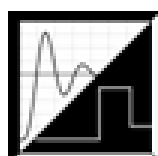
References

- [1]. Jayaraman S., S. Esakkirajan, T. Veerakumar. Digital Image Processing, New Delhi, Tata McGraw - Hill Education Pvt. Ltd, 2010, pp. 112-121
- [2]. Rupalatha, T., G. Rajesh, K. Nandakumar. Implementation of Distributed Canny Edge Detector on FPGA. International Journal of Computer Engineering Science (IJCES). 2013, Vo;. 3 (5), pp. 22-28

- [3]. Chaithra, N. M., K. V. Ramana Reddy. Implementation of Canny Edge Detection Algorithm on FPGA and displaying Image through VGA Interface, International Journal of Engineering and Advanced Technology (IJEAT), 2013, Volume 2 (6), pp. 243-247
- [4]. Aravindh G., C. S. Manikandababu. Algorithm and Implementation of Distributed Canny Edge Detector on FPGA, ARPN Journal of Engineering and Applied Sciences, 2015, Vol. 10 (7), pp. 3208-3216
- [5]. Xu, C. Chakrabarti, L. J.Karam. A distributed Canny edge detector and its implementation on FPGA, Digital Signal Processing Workshop and IEEE Signal Processing Education Workshop (DSP/SPE), 2011, pp. 500 – 505
- [6]. Veeranagoudapatil, Chitra Prabhu. Distributed Canny Edge Detector: Algorithm & FPGA Implementation, International Journal for Research in Applied Science & Engineering Technology (IJRASET), 2015, Vol. 3 (5), pp. 586-588
- [7]. Gentsos, C., C. Sotiropoulou, S. Nikolaidis, N. Vassiliadis. Real-time canny edge detection parallel implementation for FPGAs, 17th IEEE International Conference on Electronics, Circuits, and Systems (ICECS), 2010, pp. 499 – 502
- [8]. Chandrashekar, N.S., K.R. Nataraj. NMS and Thresholding Architecture used in FPGA based Canny Edge Detector for Area Optimization, Proceeding. of International Conference on Control, Communication and Power Engineering, 2013, pp. 80-84
- [9]. Ramgundewar, Pallavi, S .P. Hingway, K. Mankar. Design of modified Canny Edge Detector based on FPGA for Portable Device, Journal of The International Association of Advanced Technology and Science. 2015, Vol. 16 (2), pp. 210-214
- [10]. Anup Singh Ramprakash Singh Rajput, Samina Jafar. Improved Distributed Canny Edge Detector in VHDL, International Journal of Electrical, Electronics & Communication Engineering, 2013, Vol. 3 (6), pp. 291-295

For contacts:

Ass. Prof. Dimitre Zh. Kromichev
 Department of Marketing and International
 Economic Relations
 University of Plovdiv
 e-mail: dkromichev@yahoo.com



CAPITAL PARAMETERS OF FAST FPGA BASED CANNY EDGE DETECTION COMPUTATIONS

Dimitre Zh. Kromichev

Abstract. The paper focuses on a set of capital parameters serving as a feasible basis for a speed optimized Canny edge detection which accommodates the technical characteristics and computational peculiarities of FPGA implementation. In compliance with their functionalities, all the parameters are presented and analyzed as belonging to one of three levels. The parameters are scrutinized with respect to their capability of impacting the Canny modules' specific computational requirements for the purpose of speed enhancement. Thus, defined is a set of plausible tools marking the guidelines for an effective organization of computations in an FPGA based Canny edge detection implementation aimed at speed.

Key words: Canny edge detection, FPGA, speed, parameter, level, module

1. Introduction

As a low level digital image processing technology intended to be used in software, Canny edge detection's being precise yet complicated renders its hardware implementation a multilateral task especially when the targeted achievement is speed. As a programmable logic representative, FPGA has been steadily increasing its market share and applications. For an FPGA-based realization to capitalize on the goal of speed, the Canny algorithm computations should be adapted and organized as to be capable of exploiting optimally FPGA peculiarities. Accordingly, the benchmark qualities of these parameters serve as a standard basis for assessing the feasibility of fast Canny computations on FPGA.

Described in the literature are several approaches aimed at speeding up the computations in FPGA based Canny. In [10], pipelining is used with input image fragmentation for fast Gaussian smoothing. In terms of speed, the flaws are: a substantial quantity of repeated excessive computations for fragment boundary pixels, recursive computations. In [9], an attempt to economize on the on-chip memory through performing the computations at block level is presented. Design flaws: parallel computations speed is eroded by excessive defragmentation calculations; recursive computations. In [7], the image is partitioned into q sub-images and each sub-image is further divided into $p \times m \times m$ blocks. Pipelining is realized by q processing units. Design flaws: heavy demands on memory; cumbersome repetitious computations. In [8], focus is mainly on new computational method in the non-maximum suppression and hysteresis thresholding modules. Design flaws: excessive sequential multiplications; recursive computations.

The objective of this paper is to present the set of capital parameters capable of guaranteeing speed optimization of Canny edge detection computations in compliance with FPGA characteristics. The task is to scrutinize each parameter with respect to its impact upon the speed enhancement of calculations in the Canny modules, and its serving as a reliable effectiveness measure to the implemented organization of computations in terms of speed. The targeted hardware is Altera FPGAs. Relevant to the analyses and conclusions arrived at in this paper are only gray scale images.

2. Canny algorithm modules

With respect to their functions, particular image processing tasks, stages, and results, all Canny computations are grouped to present five distinctive successive modules (Fig. 1.):

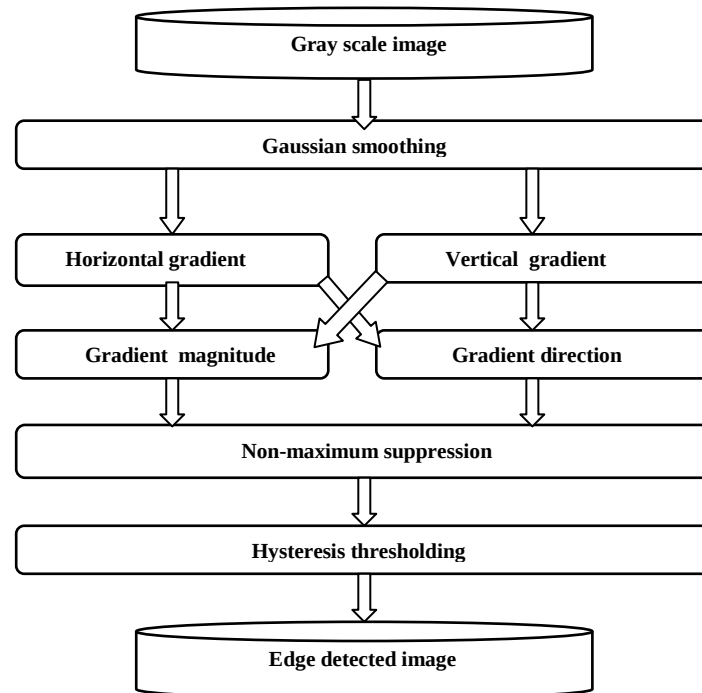


Fig. 1. Canny edge detection block diagram

Complicated as it is, both at each stratum and as a whole, Canny yet provides tangible opportunities for speed enhancement on the basis of organizing the computations by means of effectively applying appropriate parameters addressing the characteristic features and capabilities of the FPGA based hardware realization.

Fulfilment of particular tasks in the thorough accomplishment of the speed optimization goal, as well as the distinctiveness of the operational focus based upon the specifics of FPGA technology, renders it paractical for all the parameters to be analyzed within three levels.

2. High level parameters

2.1. Pipelining

Pipelining is indispensable to the FPGA-based Canny edge detection implementation focused on speed. The start of pipelined computations is signified by the execution of the first integer division operation in the Gaussian smoothing module. At this clock cycle the multipliers are ready to commence the new set of convolution operations employing the square neighbourhood pixel values centered around the second image pixel to be filtered. The effectiveness of pipelining imposes substantial demands upon the organization of computations as well as the efficiency, measured in terms of clock cycles, characterizing each of the techniques utilized for executing the essential integer arithmetic which the Gaussian smoothing function is based upon. With respect to this, the set of prioritized requirements pipelining should satisfy prove to be:

- 1) No clock cycle(s) empty of correct computational results for each of the image pixels being filtered.
- 2) No clock cycle(s) empty of corect computational results between two consecutive image pixels being filtered.
- 3) No intermediate storage between two consecutive image pixels being filtered.

Thus, for a Gaussian filter of size $Z \times Z$, and image of size $M \times N$, the upper speed limit the execution of pipelined calculations eventually approximates can be expressed as the least total number of clock cycles required for the linear weighted average filtering of the input image. Consequently, the smallest total amount of clock cycles presents the optimal quantitative

measurement of the pipelining parameter for a Canny edge detection module. For Gaussian smoothing it is:

$$M*N - \{[M - (Z-1)] * (Z-1) + N * (Z-1)\}. \quad (1)$$

In the second Canny module, computing the orthogonal gradients by employing the two Sobel filters is a neighbourhood operation. The Sobel masks being of size 3x3, the exact number of clock cycles required for the parallel computations of the vertical and horizontal gradients to commence is:

$$2*(N-(Z-1)) + 4. \quad (2)$$

Depending on the organization of computations, the upper speed limit of the pipelined computations in terms of the smallest possible amount of clock cycles taken for all the vertical and horizontal gradients to be calculated is:

$$M*N - \{[M - (Z-1)] * (Z-1) + N * (Z-1)\} + 2. \quad (3)$$

In the module computing the gradient magnitude and direction, there are no neighbourhood operations. Consequently, the pipelined computations start as soon as the first horizontal and the first vertical gradient values are available at the inputs of the module. Although being subject to simultaneous processing, magnitude and gradient computations differ as to the number of clock cycles required for the correct results to be obtained. On that basis, the speed of pipelining is determined by the gradient direction calculations, which prove to be considerably more demanding in terms of arithmetic operations. Thus, the smallest possible amount of clock cycles presenting the upper speed limit of the pipelined computations for this module is:

$$M*N - \{[M - (Z+1)] * (Z+1) + N * (Z+1)\} + 4. \quad (4)$$

The non-maximum suppression module involves neighbourhood operations. Computing the local maximum being based on comparing one magnitude with two adjacent magnitudes positioned along the direction of the gradient, the exact number of clock cycles required for the module to start executing is:

$$2*(N-(Z+1)) + 4. \quad (5)$$

The essence of calculations in this module being considered, the upper speed limit of the pipelined computations is presented by:

$$M*N - \{[M - (Z+3)] * (Z+3) + N * (Z+3)\} + 2 \quad (6)$$

clock cycles.

The hysteresis thresholding is the most computationally expensive module. The complexity of the algorithm employed here demands that the exact number of clock cycles required for the module to start executing should be:

$$4*(N-(Z+3)) + 6. \quad (7)$$

Each pipelined magnitude being compared to the two calculated in advance threshold values (high - $T1$, and low - $T2$), the essential focus here is on the determining of connectivity/non-connectivity with respect to $T1$. Consequently, the upper speed limit in terms of required clock cycles, is:

$$M*N - \{[M - (Z+5)] * (Z+5) + N * (Z+5)\} + 8. \quad (8)$$

2.2. Memory

For the Canny edge detection to be implemented on FPGA, memory plays a crucial role. With memory, several aspects are to be taken into consideration:

- 1) The on-chip memory on Altera FPGAs is limited in quantity.
- 2) Canny algorithm involves neighbourhood operations.
- 3) The speed limitations imposed by the on-chip memory.

As to addressing the memory parameter of Canny edge detection computations, it has to be taken into account that the hysteresis thresholding module demands that there be enough storage

space for the whole edge detected image. This serves as a basis for estimating the bare minimum of memory required for the implementation of Canny algorithm. The portions of memory required by the neighbourhood operations should also be added up to that basis.

The neighbourhood operations' demands on memory are focused on:

- 1) Memory should be organized appropriately as to provide a single clock access to as many as possible image pixels required for executing the neighbourhood operations. The most suitable option here, provided by Altera is Dual-port RAM [1],[2],[3],[4].
- 2) Dual-port RAMs have to contain whole rows of the input image. The depth of a single Dual-port RAM depends on the restrictions imposed on the memory organization by Altera[1],[2],[3],[4]. To this end, whatever the size of the image, the memory utilization should guarantee an even number of Dual-port RAMs needed for a particular neighbourhood operation. All the Dual-port RAMs should be of equal depths.
- 3) For one thing, the technique of revolving Dual-port RAMs - storing one image row through replacing another that is not usable (equal to storing only the image rows required by the neighbourhood operation), should be employed for the purpose of sparing the fastest memory, thus providing the opportunity images of different sizes to be processed.

Consequently, the total memory required by the FPGA implemented Canny edge detection depends on the image size as presented by the expression:

$$M*N + (Z+1)*N + 21*N . \quad (9)$$

The on-chip memory has a direct impact upon the speed optimization objective since the highest frequency[1],[2],[3],[4] this memory can be written to or read from presents the upper speed limit the execution of calculations can eventually approximate. Consequently, the on-chip memory speed dimensions actually defines the optimal speed standard for Canny computations.

3. Mid-level parameters

3.1. Parallelism

In terms of speed, one of the significant advantages of FPGA-based hardware implementation of Canny is the capability of executing calculations in parallel mode. The organization of computations in Canny edge detection is generally responsible for employing that FPGA characteristic to the fullest. This especially impacts the speed optimization of calculations in the modules whose computational algorithms are based on neighbourhood operations. The extent to which the architecture benefits from the parallel computations actually presents one of the platforms for evaluating the feasibility of speed effectiveness.

Parallelism with its significance to speed enhancement proves to be particularly important in cases of processing multiple operands of the same type. The organization of computations taken into account, both in Gaussian smoothing and the orthogonal gradients modules it is addition that requires a substantial focus on parallelism. While the number of addends with the Sobel filtering is a constant, in Gaussian smoothing this number varies according to the size of the linear weighted average filter. With respect to propagation delay times on Altera FPGAs [6], executing integer arithmetic operation addition by employing multi-staged general purpose two-input adder structure has a negative impact on speed in compliance with the following expression:

$$\begin{aligned} L &= P & \text{if } N = 2^P \\ L &= P+1 & \text{if } N > 2^P \text{ \& } N < 2^{P+1} , \end{aligned} \quad (10)$$

where:

- L is the number of consecutive levels of adders ,
- N is the number of addends,
- P is a positive integer.

For one thing, Altera provides specialized functions [5] to optimize such calculations in terms of speed so as to avoid the possibility of running into performance eroding delays determined by the implementation of one integer arithmetic operation in several consecutive stages.

As to blocks within one and the same computational module (Fig. 1.), parallelism is mainly focused on computing the vertical and horizontal gradients, as well as gradient magnitude and direction. This is based on the fact that within the flow of pipelined computations the two components of both modules utilize one and the same value calculated in the previous Canny module. The parallel approach is also applicable to non-maximum suppression module in relation to the values calculated in it being the platform for computing both high and low thresholds to be further used in the hysteresis thresholding module. The benefits of parallelly computing thresholds focus on the dynamic character of obtained values, which definitely results in a much more reliable detection of true edges. Consequently, this parallelism provides two advantages. On the one hand, the speed is enhanced in view of the fact that no additional computational time is required to calculate the two thresholds on the basis of the entire image statistics. On the other hand, in terms of plausibility and correctness of the final edge detected image, this technology provides much more superior results as opposed to using two static generalized threshold values without taking into account the particulars of the concrete image being processed.

3.2. Absence of recursive computations

The focus of this parameter is mainly on the hysteresis thresholding module whose essential computational effort is framed by the true/false edge discrimination on the basis of connectivity/non-connectivity criterion. The particulars of calculations employing two threshold values being taken into account, excluding recursive computations relates to satisfying the following speed optimization orientated requirements:

- 1) Reducing the number of computational operations a single pixel participates in to the possible minimum.
- 2) Algorithmically being based on neighbourhood operations, all the calculations performed should be confined to consecutive pixel rows.

The pixel values obeying the connectivity requirement theoretically being positioned across the whole image, the accomplishment of the task for a thresholding without recursive computations also demands an efficient memory organization that is capable of both statically mapping the final edge detected picture, and dynamically controlling the different sets of pixel values falling between the high and low thresholds.

4. Low level parameters

4.1. Optimization of integer arithmetic operations

With respect to speed, integer arithmetic operations differ considerably, the fastest among them being addition, subtraction, and multiplication (especially in terms of utilizing hard multipliers) [1],[2],[3],[4]. Consequently, the speed optimization objective benefits from:

- 1) Reducing the whole bulk of integer arithmetic required by the FPGA implementation of Canny edge detection to only a small set of operations capable of providing optimal speed characteristics.
- 2) Employing only integer arithmetic operations whose execution takes exactly one clock cycle to provide correct computational results.

Apart from multiplication and addition, the linear weighted average filtering in Gaussian smoothing comprises division, which is quite a slow operation on Altera FPGAs [5], thus being hardly compatible with fast Canny computations both in terms of optimal clock frequency and the number of clock cycles required to obtain reliable results. Consequently, to prevent speed erosion and significant delays, it is replaced with a technique of equivalent computational correctness which

definitely matches addition, subtraction, and multiplication on the platform of speed characteristics, and is optimally capable of reducing the computational time retaining the accuracy of results.

Computing the orthogonal gradients utilizes addition, subtraction, and, taking into account that the divisor there is a power of 2, a fast equivalent to right shift is employed to address the speed optimization goal accomplishment in that Canny module.

In computing the gradient magnitude and direction, besides indispensable addition and multiplication, comparison functions come into play. They are way too slower [5], and are consequently efficiently replaced by subtraction combined with positive/negative sign characterization of the values calculated for both horizontal and vertical gradients. Non-maximum suppression and hysteresis thresholding modules are entirely based on subtraction as a replacement for comparison functions.

4.2. The width of input data

Confining the input data to 8 bits is of significant importance in the Canny modules as a tool for diminishing delays. On Altera FPGAs, the fastest integer arithmetic execution in terms of clock cycles is guaranteed for 8-bit values [5]. The focus on the data widths providing optimal computational speed is analyzable with respect to the following aspects:

- 1) The organization of computations within each of the Canny modules.
- 2) The relevance of calculated values larger than the maximum pixel value in a gray scale image for each of the Canny modules.
- 3) The proper scaling techniques.
- 4) The need for employing certain Altera defined functions whose execution speed is inversely proportional to the widths of input data.

Depending on the linear weighted average filter coefficients, the values calculated in the Gaussian smoothing module employing multiplication and addition are the largest possible ones in the entire Canny. For the purpose of preventing delays measurable in more than two clock cycles per single addition execution [5], division by the sum of all the coefficients in the Gaussian filter can be executed as soon as the result of each individual multiplication is available. Thus, the quotient is 8 bits wide, which is definitely beneficial to speed when addition involves more than two addends [5] to obtain the final value of the smoothed image pixel.

The technology of computing orthogonal gradients keeps the magnitude of calculated results within the limits of 255, thus no scaling, which introduces excessive delays on the basis of comparison, is required.

Taking into account that computing the gradient magnitude refers to the Pythagorean theorem, and magnitude values larger than 255 are irrelevant to the calculations of local maxima in the next Canny module (Fig. 1.), all the results equal or exceeding 2^8 are scaled down to the appropriate 255. With respect to the fact that the calculations of gradient direction can only be one of four different values – 0, 45, 90, 135, the 8-bit data demand is met by default.

The values calculated for the gradient magnitude and direction are straightforwardly used in the non-maximum suppression module. As long as comparison of values limited of up to 255 is employed, keeping within 8 bits has no alternative. The same applies exactly to the hysteresis thresholding module.

Actually, dynamic calculation of the high and low threshold values required by the hysteresis thresholding is the only segment of Canny edge detection computations where the 8-bit approach cannot be realized on the basis of the accumulators' being employed in the processing. Depending on the size and characteristics of the image being subjected to edge detection, the values finally stored in the accumulators exceed too much the width of 8 bits. Yet, this cannot negatively affect speed taking into account that:

- 1) It is a computational mechanism contemporaneous with the non-maximum suppression calculations, and, on the basis of non-maximum suppressions's being a much more complicated

algorithm, the delays incurred by manipulating the values larger than 8-bits do not actually erode speed.

2) The algorithm's characteristics do not demand any repetitive procedures to execute the comparing of values stored in the accumulators.

5. Conclusions

In this paper presented is a set of six capital parameters defining the specifications of a speed optimized Canny edge detection to be implemented on Altera FPGAs. The parameters are scrutinized and analyzed in terms of three levels depending on each parameter's particular role in the thorough accomplishment of the speed enhancement goal, as well as the distinctiveness of the operational focus: high level - pipelining and memory; midlevel – parallelism and absence of recursive computations; low level – optimization of integer arithmetic operations and the width of input data. The smallest total amount of clock cycles presents the optimal quantitative measurement of the pipelining parameter for a particular Canny module. The essential demands on memory are focused on providing a single clock access to as many as possible image pixels required for executing the neighbourhood operations, and the technique of revolving Dual-port RAMs. Parallelism is particularly applicable in processing operands of the same type, computing the vertical and horizontal gradients, the gradient magnitude and direction, and calculating both high and low thresholds in a dynamic fashion. Excluding recursive computations relates to essentially satisfying two requirements: reducing the number of computational operations a single pixel participates in to the possible minimum; all the calculations performed should be confined to consecutive pixel rows. The optimization of integer arithmetic operations renders the Canny computations to rely only upon the fastest integer arithmetic on Altera FPGAs. Confining the input data to 8 bits is a tangible tool for achieving optimal execution speed, and diminishing delays. In their totality, these parameters provide the guidelines for feasible results in the speed optimization of Canny edge detection on the platform of Altera FPGAs' essential characteristics.

References

- [1]. Altera Corporation. Cyclone IV Device Handbook. 2014, Volume 1
- [2]. Altera Corporation. Cyclone V Device Handbook. 2014, Volume 1
- [3]. Altera Corporation. Stratix III Device Handbook. 2014, Volume 1
- [4]. Altera Corporation. Stratix IV Device Handbook. 2014, Volume 1
- [5]. Altera Corporation. Integer Arithmetic IP Cores User Guide, 2014
- [6]. Altera Corporation. The Quartus II TimeQuest Timing Analyzer, 2014
- [7]. Chandrashekar N.S., K. R. Nataraj. Design and Implementation of a Modified Canny Edge Detector based on FPGA, International Journal of Advanced Electrical and Electronics Engineering, (IJAEED). 2013, Vol.2, (1), pp. 17-21
- [8]. Chandrashekar N.S., K. R. Nataraj. NMS and Thresholding Architecture used for FPGA based Canny Edge Detector for Area Optimization, Proceeding of International Conference on Control, Communication and Power Engineering. 2013, pp. 80-84
- [9]. Divya. D., P. S. Sushma. FPGA Implementation of a Distributed Canny Edge Detector, International Journal of Advanced Computational Engineering and Networking. 2013, Vol. 1, (5), pp. 46-51
- [10]. Shamlee V., Jeyamani. A Split Canny Edge Detection: Algorithm and its FPGA Implementation. International Journal of Science and Research (IJSR). 2014, Vol. 3, pp. 1198 -1205

For contacts:

Ass. Prof. Dimitre Zh. Kromichev
Department of Marketing and International Economic Relations
University of Plovdiv
e-mail: dkromichev@yahoo.com

Марияна Г. Тодорова, Ренета Д. Първанова

Резюме: Уейвлет функциите намират широко приложение в инженерната дейност, в това число и при обработка на сигнали. В настоящата статия са разгледани възможностите на уейвлет функциите при филтриране на зашумени сигнали. Оценена е ефективността на филтрирането при различни нива на отношението шум/сигнал на разглеждания сигнал. Получените резултати са сравнени и анализирани.
Ключови думи: зашумени сигнали, уейвлет функции, уейвлет преобразуване, филтриране.

Filtration of noisy signals by wavelet functions

Mariana G. Todorova, Reneta D. Parvanova

Abstract: Wavelet functions are widely used in engineering activities, including the signal processing. The possibilities of wavelet functions in filtration of noisy signals are discussed in this paper. The effectiveness of filtering at different levels of the ratio noise / signal is evaluated. The obtained results are compared and analyzed.

Keywords: noisy signals, wavelet function, wavelet transformation, filtration

1. Въведение

При изследване на функционирането на дадена автоматична система е необходимо да се разбере как постъпва и как се преобразува информацията в нея. Тъй като тя се съдържа в сигналите, преминаващи през отделните ѝ елементи, точността на снемане на входно–изходните данни на системата и последващата им обработка са от съществено значение.

На практика се налага да се обработват зашумени сигнали. Източниците на шум могат да бъдат много разнообразни. Например, измерване на данните с грешки; наслагване върху полезния сигнал на друг сигнал, постъпващ от околната среда; възникване на смущения при предаване на сигнала и други.

Извличането на полезна информация от зашумени данни е често срещан проблем в инженерната дейност.

Уейвлет функциите намират все по-широко приложение, особено в областите, където е необходимо да бъдат обработвани силно зашумени сигнали. Уейвлет анализът дава възможност за филтриране на информацията. Той осигурява по-гъвкава техника за обработка на сигнали в сравнение с класическия анализ на Фурие.

В настоящия доклад са разгледани възможностите на често използвани уейвлет функции при филтриране на шум от сигнали. Използвани са сигнали с различно отношение шум/сигнал. Оценена е ефективността на филтрирането и получените резултати са сравнени.

2. Уейвлет преобразуване

Уейвлет преобразуването представлява представяне на функции посредством уейвлети. Уейвлетите са специален вид ортогонални функции, които се използват за разлагане на функции или непрекъснати във времето сигнали по честотни елементи и изучаването на всеки честотен елемент с разделителна способност, съответстваща на мащаба му. Те осигуряват много добра разделителна способност по време за високи честоти и удовлетворително разрешение по честота за ниски честоти. Това е възможно дори при отсъствие на информация за характера на времевите и честотните параметри на сигнала.

Уейвлет функциите принадлежат на $L^2(\mathbb{R})$ пространството и имат нулева средна стойност.

$$\int_{-\infty}^{\infty} \psi(t) dt = 0 \quad (1)$$

При някои уейвлет функции, наречени уейвети от m -та степен, е възможно не само нулеви, но и всички m момента да са равни на нула [9].

$$\int_{-\infty}^{\infty} t^j \psi(t) dt = 0, \quad j = 0, 1, \dots, m \quad (2)$$

Уейветите от дадено семейство $\psi_{a,b}(t)$ се определят чрез функцията $\psi(t)$ (майчин уейвлет) и мащабиращата функция $\varphi(t)$ (бащин уейвлет) във времевата област. Те са мащабирани и транслирани копия на вълната майка $\psi(t)$, която обикновено е бързо затихващо трептение или такова с крайна дължина.

Основната идея на уейвлет преобразуването е, че то трябва да позволява само промените в големината, но не и във формата на вълната. Това се влияе от избора на подходящи базисни функции.

Уейвлет преобразуването на дадена функция $f(t)$ може да се представи като

$$Wf(b, a) = \int_{-\infty}^{\infty} f(t) \frac{1}{\sqrt{a}} \psi^*\left(\frac{t-b}{a}\right) dt \quad (3)$$

Вълновите коефициенти c_{jk} могат да бъдат изчислени по формулата

$$c_{jk} = [Wf(b, a)](2^{-j}, k2^{-j}), \quad (4)$$

където $2^{-j}a$ се нарича двоична дилатация, а $k.2^{-j}b$ - двоична позиция.

3. Филтрация на зашумени сигнали посредством уейвлет преобразуване

Филтрацията посредством използване на уейвлет преобразуване е ефективна в много инженерни приложения. Тя може да бъде използвана преди съществената обработка на сигнала за намаляване на белия шум на сензорите.

Всеки отчет от сигнала на сензора е сума от истинската му стойност и случайна шумова стойност с Гаусово разпределение. Моделът на зашумен сигнал може да бъде представен с израза

$$y_n(t) = y(t) + v(t) \quad (5)$$

където: $y_n(t)$ – зашумен сигнал; $y(t)$ – оригинален сигнал; $v(t)$ – Гаусов шум.

Полученият след филтрация сигнал може да бъде представен с $Y(t)$.

С цел по-добра ефективност при филтрация на шума от сигнала е необходимо да се определят:

- вида на уейвета, с който да се извърши преобразуването;
- нивото на декомпозиция на сигнала;
- стойността на прага, спрямо който коефициентите от преобразуването се нулират;
- методът, по който да се приложи определеният праг;
- начинът за декомпозиране на сигнала.

Изборът на подходящ уейвлет се определя от това каква информация е необходимо да бъде извлечена от сигнала. При филтриране на сигнали уейветът трябва да е ортогонален

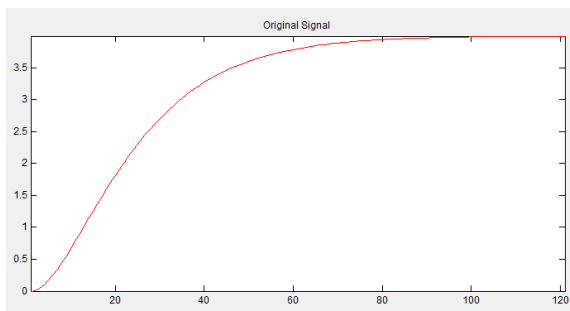
или биортогонален. Това ограничава изборът на уейвлет измежду фамилиите Daubechies, Haar, Symlets, Coiflets, BiorSplines, ReverseBior и Dmeyer.

Изборът на ниво, до което да се декомпозира статичен сигнал, се базира на естеството на сигнала и на базата на количествена оценка на някои от статистическите параметри на получаваните коефициенти. Като критерий за избор на ниво на декомпозиция при статични данни често се използва коефициентът на корелация. Оптимално ниво на декомпозиция е нивото с минимална стойност на коефициента на корелация. При динамични данни е необходимо предварително да се определи честотната област на информативните компоненти в сигнала. След това се прави декомпозиране на сигнала до ниво, при което се запазват тези честотни компоненти, а се премахват честотните съставки с по-високи честоти.

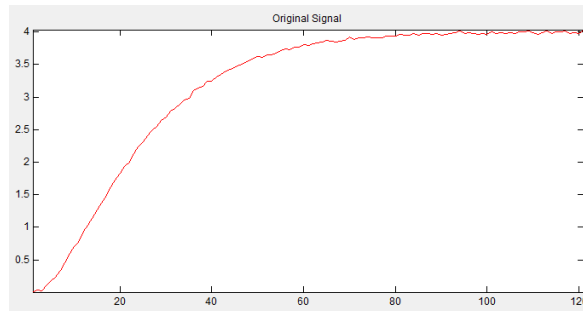
Необходимо е изборът на праг да бъде функция на статистическите характеристики на шума, за да се запази информативната част от сигнала. Често използвани методи за избор на прагова стойност са: VisuShrink, SureShrink, BayesShrink, минимакс. Прилагането на праговата стойност може да бъде по методите «твърд» праг или «мек» праг. При метода «твърд» праг сигналът се запазва, ако е по-голям по абсолютна стойност от праговата стойност. Този метод предизвиква по-големи трептения в областите, в които има прекъсвания в сигнала, в сравнение с метода «мек» праг. Методът «мек» праг има добри математически свойства и е за предпочитане, тъй като добавъчният шум постепенно се намалява.

4. Експериментални резултати

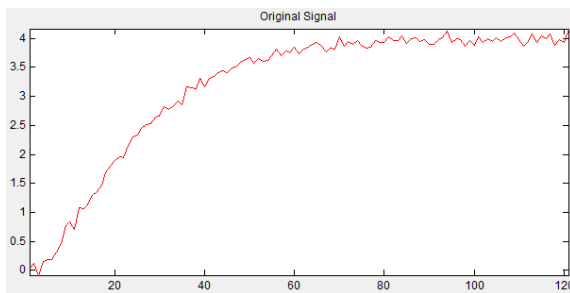
Симулиран е често срещан в автоматичните системи сигнал, представен на фигура 1, към който е добавен бял Гаусов шум до достигане на зададено отношение шум/сигнал (γ). Зашумените сигнали при различни нива на шума са показани на фигури 2-4.



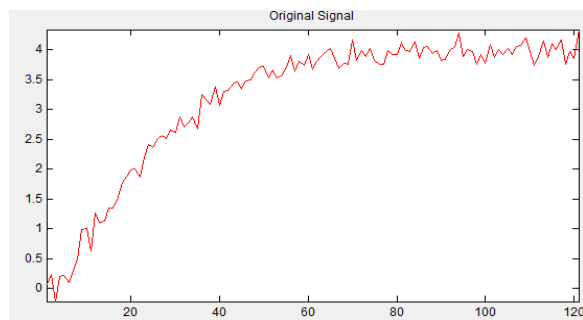
Фиг. 1. Незашумен сигнал



Фиг. 2. Зашумен сигнал при $\gamma=1\%$



Фиг. 3. Зашумен сигнал при $\gamma=5\%$



Фиг. 4. Зашумен сигнал при $\gamma=10\%$

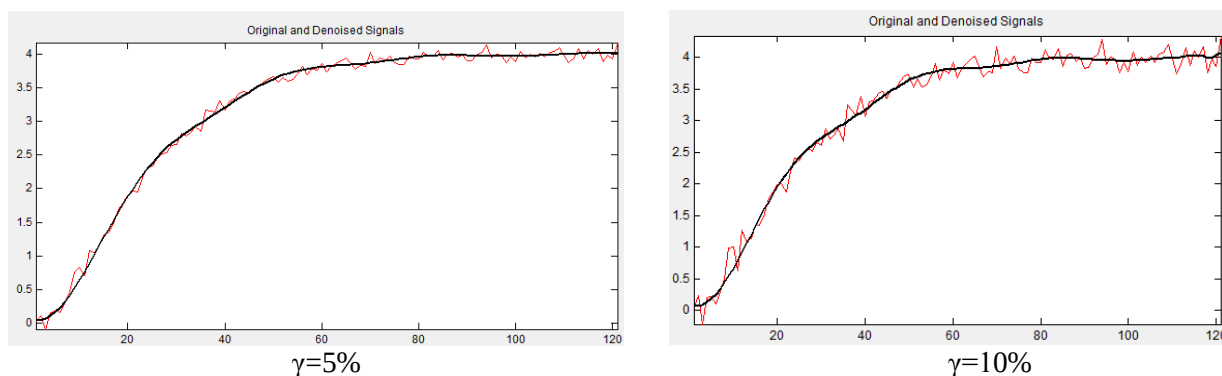
След това е извършено филтриране на получените зашумени сигнали посредством използване на уейвлети от фамилиите Daubechies, Symlets, Coiflets и Dmeyer.

Изборът на подходящ уейвлет е извършен на базата на количествена оценка на ефективността на филтрацията. Използваните критерии са: максимална относителна грешка (δ_{max}) и коефициент на намаляване на шума (CNR), определени съответно с изразите

$$\delta_{\max} = \max_i \frac{|y(t_i) - Y(t_i)|}{y(t_i)} \cdot 100[\%] \quad (6)$$

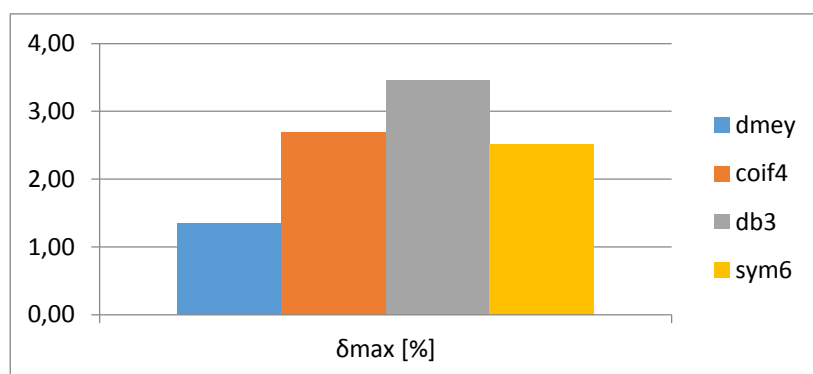
$$CNR = \frac{\sum_{i=1}^N [y_n(t_i) - y(t_i)]^2}{\sum_{i=1}^N [Y(t_i) - y(t_i)]^2} \quad (7)$$

Част от получените след филтрацията сигнали са представени на фигура 5.

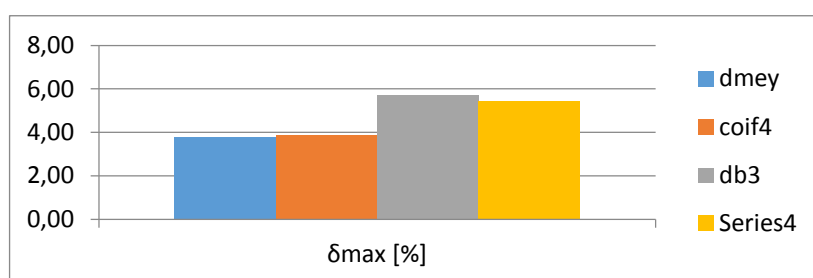


Фиг. 5. Зашумени и филтрирани посредством dmey вълничка сигнали

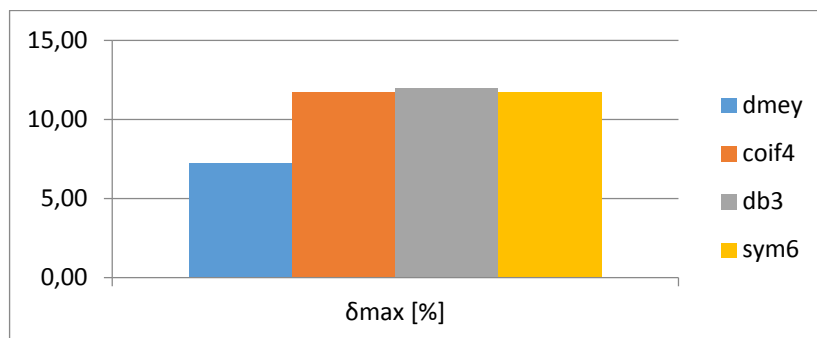
Изчислените максимални относителни грешки и CNR са дадени на фигури 6 – 11 и в таблица 1.



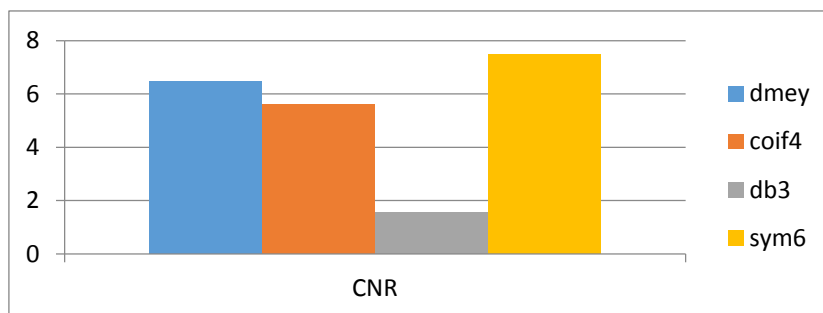
Фиг. 6. Максимални относителни грешки при $\gamma=1\%$



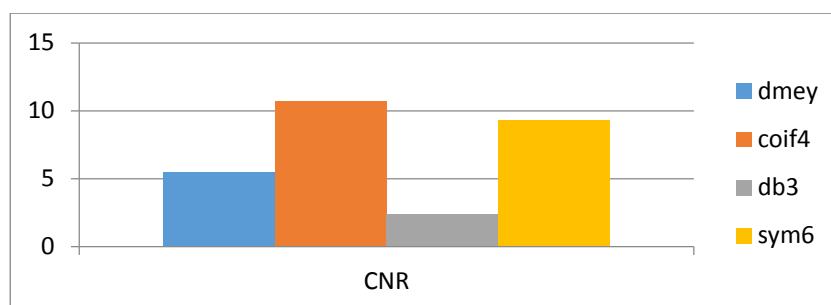
Фиг. 7. Максимални относителни грешки при $\gamma=5\%$



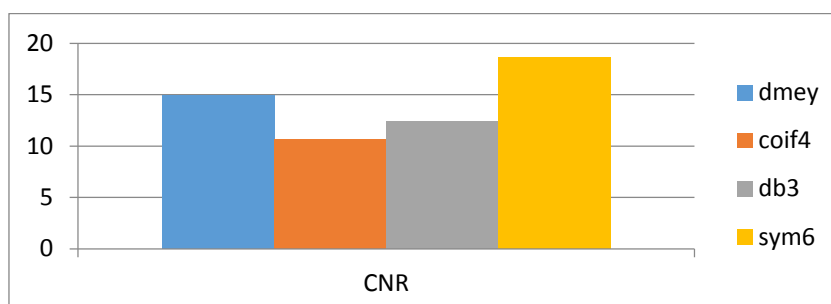
Фиг. 8. Максимални относителни грешки при $\gamma=10\%$



Фиг. 9. Коефициент на намаляване на шума при $\gamma=1\%$



Фиг. 10. Коефициент на намаляване на шума при $\gamma=5\%$



Фиг. 11. Коефициент на намаляване на шума при $\gamma=10\%$

Таблица 1. Стойности на максималната относителна грешка и коефициента на намаляване на шума при филтриране на зашумените сигнали с различни уейвлет функции

използвани уейвлети	dmey	coif4	db3	sym6
Отношение шум/сигнал $\gamma=1\%$				
$\delta_{\max}$ [%]	1,35	2,68	3,45	2,51
CNR	6,470226	5,600179	1,561464	7,487619
Отношение шум/сигнал $\gamma=5\%$				
$\delta_{\max}$ [%]	3,78	3,85	5,73	5,43
CNR	5,487622	10,73532	2,415217	9,335123
Отношение шум/сигнал $\gamma=10\%$				
$\delta_{\max}$ [%]	7,27	11,73	11,96	11,73
CNR	14,98455	10,62786	12,43002	18,62184

Заклучение

Филтрацията на зашумени сигнали посредством уейвлет преобразуване на сигнала има съществени предимства, които произтичат от самите свойства на уейвлет функциите.

Въз основа на изчислените максимални относителни грешки, получени при филтриране с различни често използвани уейвлет функции, се стига до извода, че за премахване на шума от апериодични сигнали се препоръчва да се използва дискретна Мейер вълничка. Предимството на този вид функция се наблюдава при филтрирането на всеки един от разгледаните зашумени сигнали, като най-ясно се откроява при силно зашумения сигнал с $\gamma=10\%$. Тези отлични резултати на дискретната Мейер вълничка спрямо останалите се дължат на факта, че тя съдържа филтър с крайна импулсна характеристика (FIR) с базирана апроксимация на Мейер уейвлет.

Твърдението, че Symlets и Coiflets вълничките са модификация на Daubechies представителите се доказва от факта, че максималните относителни грешки при използване на Daubechies функциите са сравнително по-големи спрямо тези на останалите два класа вълнички.

При използване на коефициента на намаляване на шума (CNR) като критерий за определяне на качеството на потиснатия шум се достига до извода, че в зависимост от нивото на шума е необходимо да се използват различни класове уейвлети. Сравнително най-добри резултати се наблюдават при използване на Symlets вълнички.

И по двата използвани критерия се стига до извода, че качеството на филтриране на сигналите е най-лошо при използване на уейвлетите на Daubechies.

Благодарности

Изследванията са осъществени благодарение на Научноизследователски проект в помощ на докторанти ПД 10/2016 “Изследване на алгоритми за идентификация на системи посредством уейвлет функции”.

Литература

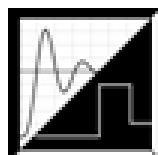
[1] Тодорова М., Р. Иванова, Изследване на приложенията на уейвлет функциите при идентификация на системи, Трета научна конференция с международно участие „Компютърни науки и технологии“, брой 2, стр. 44-54, ISSN 1312-3335, 2015.

- [2] Тодорова М., Приложение на изместените ортогонални полиноми при апроксимация на таблични данни, Годишна научна сесия с международно участие, ТУ – Варна, ISSN 1311-896X, Стр. 1261 – 1266, 2001.
- [3] Тодорова М., Изследване на шумоустойчивостта на алгоритми за идентификация на обекти с разпределени параметри, Известия на Съюза на учените – Варна, ISSN 1310-5833, Серия “Технически науки”, Стр. 12 – 15, 1’2002.
- [4] Тодорова М., Работоспособност на някои алгоритми за идентификация на обекти с разпределени параметри при наличие на рязко отличаващи се наблюдения, Известия на Съюза на учените – Варна, ISSN 1310-5833, Серия “Технически науки”, Стр. 8 – 11, 1’2002.
- [5] Todorova M., R. Ivanova, Research and signal analysis with different classes of wavelets, Signal processing, ICEST, Makedonia, 2016.
- [6] Todorova M., Research of the possibilities an dapplication of two dimensional orthogonal functions for dynamic distributed parameters systems identification, PhD Thesis, Varna, 2003.
- [7] Давидов А.В., Основы вейвлет- преобразования сигналов, 2008.
- [8] Doroslovacki M., H. Fan. Wavelet-B ased Linear System Modeling and Adaptive Filtering, IEEE Transactions on signal processing, 1996, vol. 44, № 5, pp. 1156- 1167.
- [9] Mallat S. A wavelet tour of signal processing, Second Edition, Academic Press, 1999, pp. 620.
- [10] Misitip M., Misiti Y., Oppenheim G., Poggi J., WaveletToolbox™ User’sGuide, 2014.

За контакти:

Доц. д-р инж. Марияна Г. Тодорова
катедра „Автоматизация на производството”
Технически университет - Варна
E-mail: mgtodorova@tu-varna.bg

Докторант инж. Ренета Данчева Първанова
катедра „Автоматизация на производството”
Технически университет - Варна
E-mail: renetaivanova1@gmail.com



СРАВНИТЕЛЕН АНАЛИЗ НА РЕКУРСИВНИ МЕТОДИ ЗА ОЦЕНЯВАНЕ НА ПАРАМЕТРИ В СИСТЕМА ЗА УПРАВЛЕНИЕ НА СКОРОСТТА НА ПОСТОЯННО ТОКОВ ДВИГАТЕЛ

Иван В. Григоров, Наско Р. Атанасов

Резюме: Адаптивното управление обхваща набор от методи, които осигуряват систематичен подход за автоматично регулиране в реално време. Рекурсивните методи за оценяване на параметри трябва да удовлетворяват изискванията за бързодействие към алгоритмите за идентификация в реално време. Това се обуславя от факта, че корекцията на модела след постъпване на данните от новото наблюдение, както и изработването на ново управляващо въздействие трябва да станат за един такт на дискретизация. В тази статия се прави анализ на надеждността на някои от по-често прилаганите рекурсивни методи за оценяване на параметри в адаптивни системи и техни вариации в системата за управление на скоростта на постоянно токов двигател.

Ключови думи: метод на най-малките квадрати, метод на инструменталната променлива, рекурсивни методи за оценяване на параметри, постоянно токов двигател.

Comparative analysis of more commonly used recursive methods for parameter estimation in DC motor control system

Ivan V. Grigorov, Nasko R. Atanasov

Abstract: The adaptive control covers a set of methods that provide a systematic approach for automatic control in real-time. Recursive methods for parameter estimation have to meet the requirements for identification algorithms in real time. This is determined from the fact that the adjustment of the model after the submission of new data from monitoring, and the development of new control action should be made in a single cycle of discretization. An analysis of the reliability of some of more commonly used recursive methods for parameter estimation in adaptive systems and their variations in DC motor control system will be made in this article.

Keywords: instrumental variable method, least squares method, recursive methods for parameter estimation, DC motor

1. Увод

Задължително условие за контрол на една система е разбирането на динамичните характеристики на обекта [3]. Рекурсивните методи за оценяване се използват при идентификация в реално време. По време на работата на обекта заедно с управлението може да продължи уточняването на модела на базата на получаваните нови данни. Това е от особено важно значение, когато характеристиките на обекта се изменят във времето. В адаптивните системи за управление характеристиките на обекта са неизвестни или се изменят във времето, а е възможно в обекта да се появи някаква неизправност. Методите за оценяване на параметри в реално време могат да помогнат за откриването на такива и тяхното по-лесно и бързо отстраняване. За целта ще се изследват някои от по-често използваните методи за оценяване на параметри в системата за управление на скоростта на постоянно токов двигател посредством ШИМ. Целта на изследването не е самото управление, а сравнителен анализ на методите за оценяване на параметри на модела и тяхната ефективност при подобно приложение.

2. Рекурсивни версии на метода на най-малките квадрати за оценяване на параметри

2.1. Рекурсивен претеглен метод на най-малките квадрати.

За да се получат рекурсивни оценки по метода на претеглените най-малките квадрати, се използва

$$\hat{\theta}_{N+1} = \hat{\theta}_N + \frac{C_N f_{N+1}}{\frac{1}{w_{N+1}} - f_{N+1}^T C_N f_{N+1}} (y_{N+1} - f_{N+1}^T \hat{\theta}_N) \quad (1)$$

Според (1) старата оценка $\hat{\theta}_N$ се коригира пропорционално на разликата $y_{N+1} - \hat{y}_{N+1}$ с векторен коефициент на пропорционалност

$$\Gamma_{N+1} = \frac{C_N f_{N+1}}{\frac{1}{w_{N+1}} - f_{N+1}^T C_N f_{N+1}} \quad (2)$$

Величината $\hat{y}_{N+1} = f_{N+1}^T \hat{\theta}_N$ е предсказаната стойност за y_{N+1} , изчислена въз основа на коефициентите от предишната итерация $\hat{\theta}_N$.

Използване на остатъци вместо грешки на предсказване

Във формула (2) участва грешката от предсказване

$$e_{N+1} = y_{N+1} - \hat{y}_{N+1}(\hat{\theta}_N) = y_{N+1} - f_{N+1}^T \hat{\theta}_N \quad (3)$$

За повишаване на точността на предсказване вместо e_{N+1} се използва остатъкът

$$r_{N+1} = y_{N+1} - f_{N+1}^T \hat{\theta}_{N+1} \quad (4)$$

От (2) и (3) следва

$$r_{N+1} - e_{N+1} = -f_{N+1}^T (\hat{\theta}_{N+1} - \hat{\theta}_N) \quad (5)$$

Ако от (2) се определи разликата $(\hat{\theta}_{N+1} - \hat{\theta}_N)$ и се замести в (5), се получава

$$r_{N+1} = \frac{\frac{1}{w_{N+1}} e_{N+1}}{\frac{1}{w_{N+1}} - f_{N+1}^T C_N f_{N+1}} \quad (6)$$

Стартиране на изчислителната процедура

За получаване на оценки по метода на претеглените най-малки квадрати са необходими $\hat{\theta}_N$ и C_N от предишни изчисления. Събират се $N \geq k$ наблюдения и по нерекурсивния метод на претеглените най-малки квадрати се намират начални оценки $\hat{\theta}_N$ и C_N . След това изчисленията продължават по (1) и (7)

$$C_{N+1} = C_N - \frac{C_N f_{N+1} f_{N+1}^T C_N}{\frac{1}{w_{N+1}} - f_{N+1}^T C_N f_{N+1}} \quad (7)$$

Алгоритъмът на рекурсивния претеглен метод на най-малките квадрати лежи в основата на редица други рекурсивни процедури. Той е следният:

- Събират се $N \geq k$ наблюдения и по нерекурсивния метод на претеглените най-малки квадрати се намират начални оценки $\hat{\theta}_N$ и C_N .
- Новите оценки се изчисляват по формула (1).
- Преизчислява се матрицата C_{N+1} по зависимост (7), за да се подготви за следващата итерация.
- С новите оценки $\hat{\theta}_{N+1}$ и C_{N+1} започва следващата итерация от т. 2 на алгоритъма.

2.2. Рекурсивен обикновен метод на най-малките квадрати.

Рекурсивният метод на най-малките квадрати може да се разглежда като частен случай на РПМНМК при $W = I$. Това означава всички тегла да се приемат равни на единица, което е възможно само ако $\rho = 1$. Поради това рекурсивният метод на най-малките квадрати (РМНК) може да се получи, като в процедурата от т.1.1. се положи навсякъде $\rho = 1$ [1, 3].

3. Рекурсивен метод на инструменталната променлива

Процедурата за получаването на оценки по рекурсивния метод на инструменталната променлива е подобна на тази по рекурсивния метод на най-малките квадрати.

Видът на инструменталния вектор v_{N+1} се избира от (8) или (8')

$$v_i^T = [-\hat{y}_{i-1} \ -\hat{y}_{i-2} \dots -\hat{y}_{i-n} \ u_{i-1} \ u_{i-2} \dots u_{i-m}] \quad (8)$$

$$v_i^T = [u_{i-1} \ u_{i-2} \dots u_{i-m}] \quad (8')$$

Алгоритъмът за получаване на оценки по рекурсивния метод на инструменталната променлива е следният:

- Оценяването започва по рекурсивния претеглен МНК. Това повишава устойчивостта на началните оценки. След натрупване на определен брой $N \geq k$ наблюдения се преминава към оценяване по рекурсивния метод на инструменталната променлива. Обикновено се избира N да е равно на 3 до 5 пъти броя на коефициентите.

- Оценките на коефициентите на $N+1$ -вата итерация се изчисляват по формулата

$$\hat{\theta}_{N+1} = \hat{\theta}_N + \frac{C_N v_{N+1}}{\rho + f_{N+1}^T C_N v_{N+1}} (y_{N+1} - f_{N+1}^T \hat{\theta}_N) \quad (9)$$

- Следващата итерация се подготвя, като се преизчисли C_{N+1} по формулата

$$C_{N+1} = \frac{1}{\rho} \left(C_N - \frac{C_N v_{N+1} f_{N+1}^T C_N}{\rho + f_{N+1}^T C_N v_{N+1}} \right) \quad (10)$$

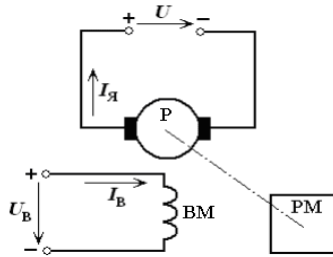
- Изчисленията започват от т. 2 с новите оценки. Ако не е необходимо претегляне на наблюденията, в (9) и (10) се полага $\rho = 1$ [1].

4. Система за управление на скоростта на постоянно токов двигател

Приложението на постоянно токовите двигатели е широко по отношение на автоматика, роботика и различни системи за управление. Подходящи са там, където е необходимо прецизно регулиране на момент и скорост, при подържане на момент при ниска и нулева скорост.

4.1. Постоянно токов двигател

Принципно устройство на двигатели за постоянен ток е показано на фигура 1, където РМ е работна машина, ВМ е възбудителна намотка, а Р е ротор.

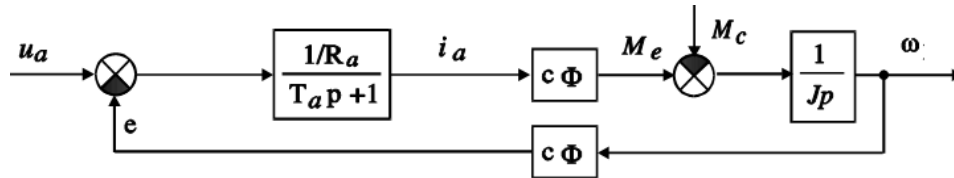


Фиг. 1. Принципно устройство на ПТД

Динамичните свойства на постоянно токовия и двигател са описани със системата

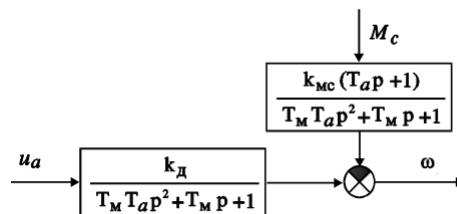
$$\begin{cases} u_a = R_a(T_a p + 1)i_a + c\Phi\omega \\ M_e = c\Phi i_a \\ M_e - M_c = Jp\omega \end{cases}, \quad (11)$$

представена чрез структурната схема (фигура 2)



Фиг. 2. Структурна схема на ПТД

която може да бъде преобразувана в (фигура 3):



Фиг. 3. Структурна схема на ПТД след преобразуване

където

$$\omega(p) = \frac{k_d}{T_m T_a p^2 + T_m p + 1} u_a(p) - \frac{k_{mc}(T_a p + 1)}{T_m T_a p^2 + T_m p + 1} M_c(p) \quad (12)$$

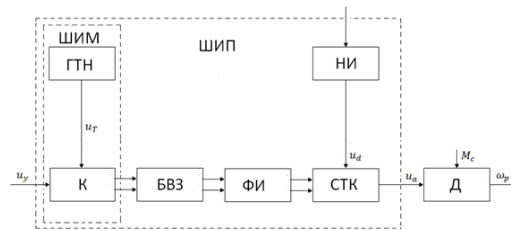
$$k_d = \frac{1}{c\Phi}; \quad k_{mc} = \frac{R_a}{(c\Phi)^2}; \quad T_a = \frac{L_a}{R_a}; \quad T_m = \frac{J R_a}{(c\Phi)^2} \quad (13)$$

4.2. Управление на скоростта на постоянно токов двигател посредством широчинно-импулсен преобразувател

Съвременните системи за управление на постоянно токови двигатели са с широчинни импулсни преобразователи, тъй като се характеризират с по-добри енергетични показатели и

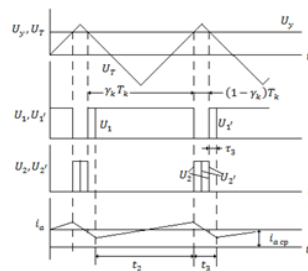
по-малки пулсации на тока и скоростта, което от своя страна води до намаляване на загубите и разширяване на диапазона на регулиране.

На блоковата схема на фигура 4 е представена принципна блок схема на ШИП-ПТД, където: ГНТ- генератор на трионообразно напрежение, К – компаратор, БВЗ – блок за времезадръжка, ФИ – формирова̀тел на импулсите, НИ – неуправляем изправител, СТК – силов транзисторен комутатор, Д – постоянно токов двигател.

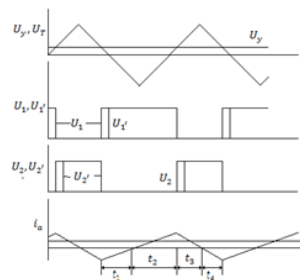


Фиг. 4. Принципна блок схема на ШИП-ПТД

На фигура 5а и фигура 5б е показан примерният вид на квазиустановен режим на работа при симетричен алгоритъм на комутация.



Фиг. 5а



Фиг. 5б

Математическото описание на процесите в системата ШИП-Д се затруднява от дискретния и нелинеен характер на силовия преобразовател. За изправителен анализ на процесите, както и за целите на синтеза, най-често се използва опростено математическо описание, неотчитащо посочените особености на преобразователя, докато за по-пълнен анализ е наложително отчитането на тези особености. За предавателната функция на ШИП по управляващия канал може да се запише:

$$W_n(p) = \frac{k_n}{T_n p + 1} \approx \frac{k_n}{\tau_{cp} p + 1} \approx \frac{\frac{u_y}{u_{аср}}}{\frac{T_k p}{2} + 1} \quad (14)$$

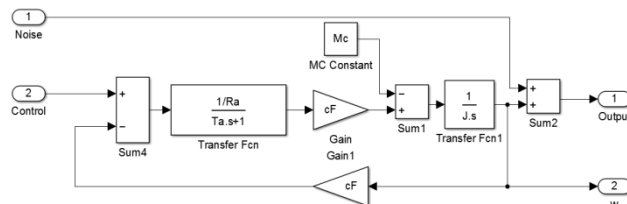
където k_n , T_n и τ_{cp} са съответно коефициент на предаване, еквивалентна времеконстанта и средностатистическо закъснение на преобразователя, обусловено от управляващото напрежение u_y .

5. Експериментални изследвания и резултати

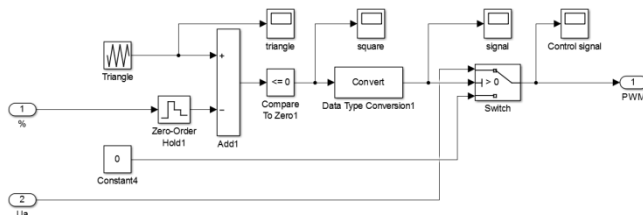
Изследванията тук се отнасят до работоспособността, възможностите и надеждността на описаните вече рекурсивни методи за оценяване на параметри.

За целта на изследването е генериран случаен входен сигнал, симулиращ зашумяване, с помощта на програмната среда MATLAB.

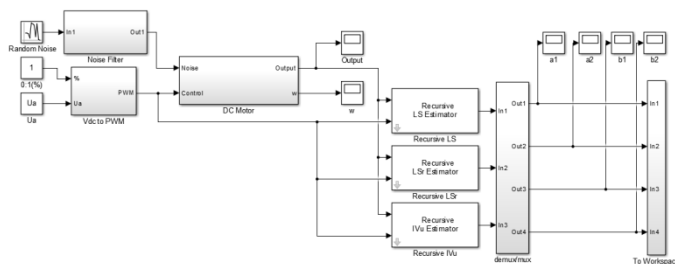
Изследванията са направени с използването на System Identification Toolbox в Matlab\Simulink. За тази цел са направени блокове за рекурсивно оценяване на параметри в реално време както следва: рекурсивен метод най-малките квадрати (RLS), рекурсивни най-малките квадрати с отчитане на остатъците вместо грешката на оценяване (RLSr) и рекурсивен метод на инструменталната променлива (IVu).



Фиг. 6. Блок схема на ПТД в Simulink



Фиг. 7. Блок схема на ШИП в Simulink

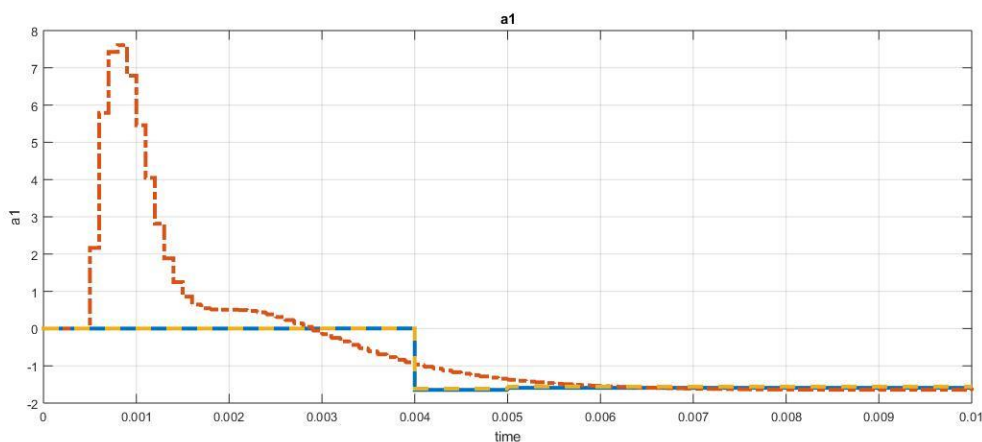


Фиг. 8. Блок схема на управление на ПТД с ШИП с рекурсивно оценяване в реално време в Simulink

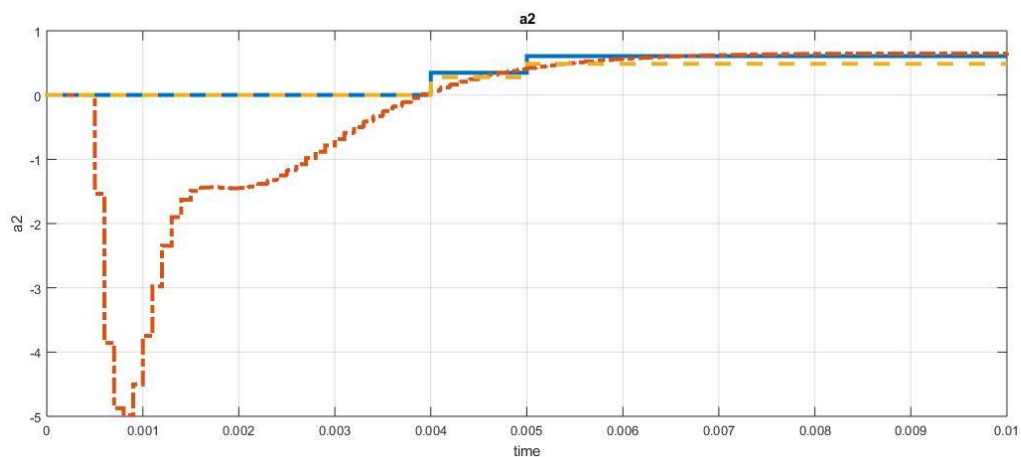
Чрез блока “0:1%” се задава импулсният сигнал от 0 до 100 % за регулиране на скоростта на постоянно токовия двигател.

Избира се модел на ПТД от втори ред с дискретна предавателна функция

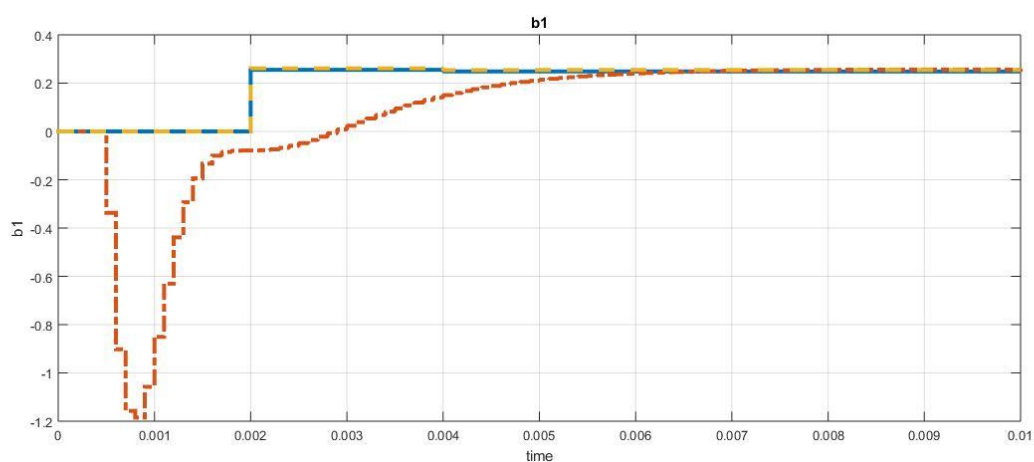
$$W(z) = \frac{0,2405z^{-1} + 0,1748z^{-2}}{1 - 1,591z^{-1} + 0,6065z^{-2}}$$



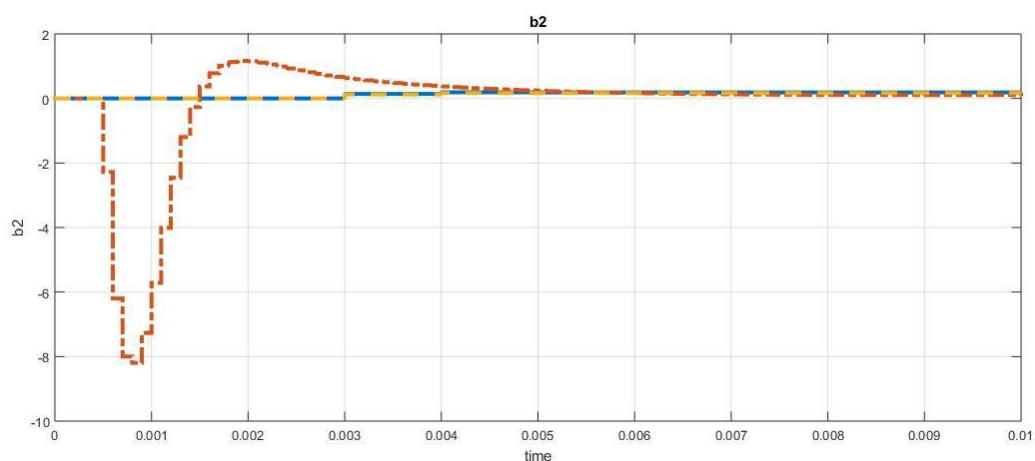
Фиг. 9. Оценяване на параметъра a1- RLS(c),RLSr(ч),IVu(ж)



Фиг. 10. Оценяване на параметъра a_2 - RLS(c),RLSr(ч),IVu(ж)



Фиг. 11. Оценяване на параметъра b_1 - RLS(c),RLSr(ч),IVu(ж)



Фиг. 12. Оценяване на параметъра b_2 - RLS(c), RLSr(ч), IVu(ж)

Реални стойности на параметрите	a1	a2	b1	b2
	-1.591	0.6065	0.2405	0.1748
Оценени стойности с RLS	-1.591	0.6066	0.2413	0.1744
Оценени стойности с RLSr	-1.592	0.6064	0.2401	0.1745
Оценени стойности с IVu	-1.590	0.6066	0.2415	0.1750

6. Анализ и изводи

- Анализираните рекурсивни методи за оценяване на параметри са едни от най-ефективни методи за оценяване на параметри, като те могат да бъдат модифицирани за по-добро бързодействие и използвани в по-сложни адаптивни системи за управление.
- При различен избор на теглата могат да се получат различни робастни оценки, които могат да бъдат нечувствителни към зашумяването.
- Рекурсивните методи за оценяване на параметри, разгледани в този доклад, могат да бъдат използвани за откриване на неизправности в САУ.

БЛАГОДАРНОСТИ

Изследванията, представени в този доклад, са проведени във връзка с проект НП-6 в Технически университет - Варна, финансиран от държавния бюджет за 2016 година

Литература

- [1]. Ангелов Н., Електротехника, ВВМУ „Н. Й. Вапцаров“-Варна,2006.
- [2]. Вучков. И.: Идентификация, ИК Юрамел, 1996
- [3]. Велев К.: Адаптивни системи, София, 1995
- [4]. Генов Г.: Идентификация на системи –теоретични основи, Варна,2007
- [5]. Мазаджиев Г., Компютърни технологии в анализа на системи и сигнали, 2003г.
- [6]. Маринова Д., Синтез на робастно управление за интелигентни структури, 2014
- [7]. Христова М., Кръстева А. : Методични указания за изследване на трифазен асинхронен двигател в програмната среда Matlab/Simulink,2012
- [8]. Николова Н., Грасиани Б., Изследване на робастни репетитивни системи за управление с вътрешен модел, 2015г.
- [9]. Agustin O., Oscar L, Francisco Q.: Identification of DC motor with parametric methods and artificial neural networks,2012
- [10]. Andonov A., Hubenova Z, Robust methods for control under undetermined criteria,2008
- [11]. Allan Aasbjerg Nielsen, Least Squares Adjustment: Linear and Nonlinear Weighted Regression Analysis, 2013
- [12]. Joshua D. Angrist and Alan B. Krueger:Instrumental Variables and the Search for Identification,2001
- [13]. Kama O., Mahanijah K., Nasirah M. and Norhayati H. : System Identification of Discrete Model for DC Motor Positioning
- [14]. Krneta R., Anti'c S., Stojanovi'c D. : Recursive Least Squares Method in Parameters Identification of DC Motors Models,2005
- [15]. Landau, I.D., Lozano R., M'Saad H., Kariirii A.: Adaptive Control Algorithms, Analysis and Applications, 2011
- [16]. Hassan, L. H Unknown input observers design for a class of nonlinear time-delay systems, 2011
- [17]. Mohamed M. Hassan, Amer A. Aly, and Asmaa F. Rashwan: Different identification methods with application to a DC motor

За контакти:

докторант, инж. Иван В. Григоров
катедра „Автоматизация на производството”
E-mail: ivan.grigorov.md@gmail.com
доцент д-р инж. Наско Р. Атанасов
катедра „Автоматизация на производството”
Технически университет-Варна
E-mail: nratanasov@tu-varna.bg

СЕКЦИЯ 4

СТУДЕНТИ И ДОКТОРАНТИ

Четвърта научна конференция с международно участие
"Компютърни науки и технологии"
30 септември – 1 октомври, 2016г.
Варна, България



Fourth Scientific International Conference
Computer Sciences and Engineering
30 September – 1 October, 2016
Varna, Bulgaria

SECTION 4

STUDENTS AND PhD STUDENTS

НЕРЕШИМИ И НЕРЕШЕНИ ЗАДАЧИ

Ирена Й. Георгиева

Резюме: Докладът разглежда темата за доказано нерешими и нерешени до този момент задачи. Най-обширно е представен проблемът със спирането и доказателството, че той е нерешим. Друг пример е последната теорема на Ферма. Като нерешени задачи са разгледани: алгоритъм на Колац, аликвотна последователност и проблемът $P=NP$.

Ключови думи: нерешими задачи, нерешени задачи, проблем със спирането, машина на Тюринг, последната теорема на Ферма, алгоритъм на Колац, аликвотна последователност, $P=NP$ проблем

Undecidable and Unsolved Problems

Irena Y. Georgieva

Abstract: The report reviews the undecidable problems and the unsolved yet problems. Most extensively is presented the halting problem and the proof that it's undecidable. Another example is Fermat's last theorem. As unsolved problems are reviewed: Collatz conjecture, aliquot sequence and $P=NP$ problem.

Keywords: undecidable problems, unsolved problems, halting problem, Turing machine, Fermat's last theorem, Collatz conjecture, aliquot sequence, $P=NP$ problem

Увод

На всеки програмист се случва да напише програма, която попада в безкраен цикъл. Как обаче можем да установим дали тя действително няма да спре или просто ѝ трябва повече време, за да върне резултат? Понякога отговорът е очевиден, но има случаи, в които не е толкова лесно да се прецени. Има задачи, за които не бихме предположили, че нямат решение. Има и такива, които остават нерешени.

Нерешими задачи

Универсална изчислителна машина

Универсална изчислителна машина е устройство, което е способно да намери решение на всеки проблем (който може да бъде разрешен), като на устройството са му осигурени достатъчно памет и време, за да завърши. Днес тези устройства наричаме машини на Тюринг.

Устройствата, които днес наричаме компютри, независимо дали под тази дума имаме предвид програмируем микроконтролер в микровълнова печка или суперкомпютърът Blue Gene, те всички са еквивалентни в смисъла на това какви проблеми могат да разрешават. За всички тях може да бъде доказано, че са еквивалентни на Машината на Тюринг. Разбира се, тези машини имат достъп до различно количество памет и работят с различна скорост.

Тюринг също така твърди, че проблем, който машината на Тюринг не може да разреши, не може да бъде разрешен по никакъв систематичен начин, нито от човек, нито от компютър. Тази теза остава недоказана! Въпреки това досега никой не е успял да опровергае тезата и много хора вярват в нейната истинност. За да се опровергае тезата, трябва човек или машина да разреши проблем, който е неразрешим от машина на Тюринг, което никой не е успял да направи до наши дни [1].

Проблем със спирането (Halting problem)

Алан Тюринг доказва, че действително има проблеми, които не могат да бъдат разрешени от универсална изчислителна машина. В теорията на изчислимостта, задачата за определяне дали дадена програма попада в безкраен цикъл, се нарича проблем за спирането или стоп-проблем (Halting problem). Състои се в това възможно ли е да се определи, ако е дадена произволна компютърна програма и входни данни за нея, дали програмата някога ще завърши изпълнението си и ще даде резултат или ще работи безкрайно дълго време.

През 1936 г. Алън Тюринг доказва, че алгоритъм, който дава отговор на тази задача за всички двойки програма – входни данни, не може да съществува. Основната идея в доказателството му е, че ако такъв алгоритъм бъде конструиран, той би противоречил на себе си, следователно той не е правилен. Ключова част от доказателството се оказва проблемът да се дефинира математически компютър и програма, което Тюринг прави с въвеждането на машината на Тюринг. Проблемът за спирането е нерешим за всяка машина на Тюринг. Това е една от първите решени задачи в Теорията на решенията.

Доказателството на Тюринг бива публикувано през май 1936 г., а доказателството на Алонсо Чърч за нерешимостта на проблема в ламбда смятането е от април 1936 г., като двамата работят независимо един от друг над този въпрос. По-късно са открити и много други нерешими проблеми. Типичен метод за доказване на нерешимостта на даден проблем е свеждането му до вече познат (доказано) нерешим проблем. След като никой алгоритъм не може да реши проблема, до който задачата е сведена, то никой алгоритъм не може да реши и по-общата задача.

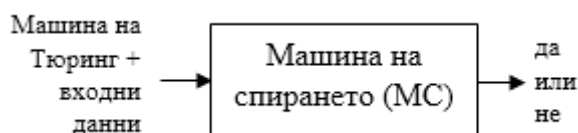
При някои алгоритми може да бъде доказано, че те ще спрат, каквито и да са входните данни, но всяко доказателство трябва да бъде разработено специално за конкретния алгоритъм. Не съществува механично, общовалидно решение, което да определи дали дадена машина на Тюринг ще спре. Все пак, могат да бъдат приложени някои евристични методи, които имат успех в голяма част от случаите. Това направление в информатиката се нарича автоматизиран анализ на приключването [2].

Доказателство на проблема със спирането

За да бъде разрешен този проблем, отговор (да или не) трябва да бъде даден за краен период от време, независимо от машината или входните данни. Очевидно някои машини никога не прекратяват изпълнението си. Например можем да напишем машина на Тюринг, която брой нагоре, започвайки от едно.

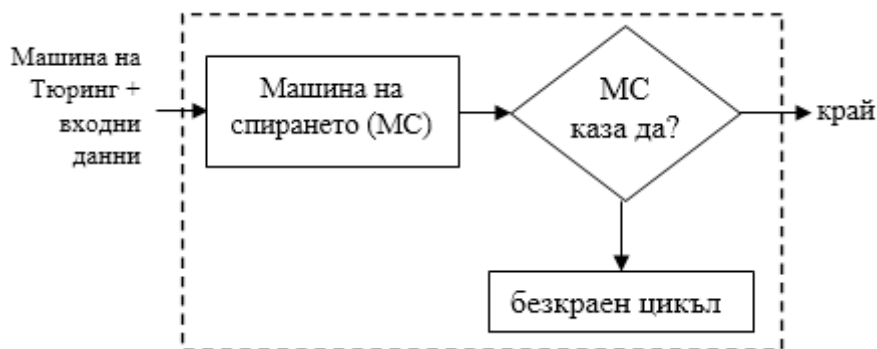
За да докажем по-лесно нерешимостта на проблема със спирането, ще го свържем с проблем, който вече ни е известен. Например Парадоксът на лъжеца. Той гласи: „Това изречение не е вярно“.

За да докажем, че няма машина на Тюринг, която може да разреши проблема със спирането, допускаме че такава машина съществува и после доказваме, че нейното съществуване е противоречиво. Нека наречем машината „Машина на Спирането“, или МС. МС е машина, която управлява друга машина и нейните входни данни, за да възпроизведе отговор (да или не) за краен период от време: или въпросната машина приключва за краен период от време (връща отговор „да“), или не (връща отговор „не“). Фигура 1 илюстрира действието на МС.



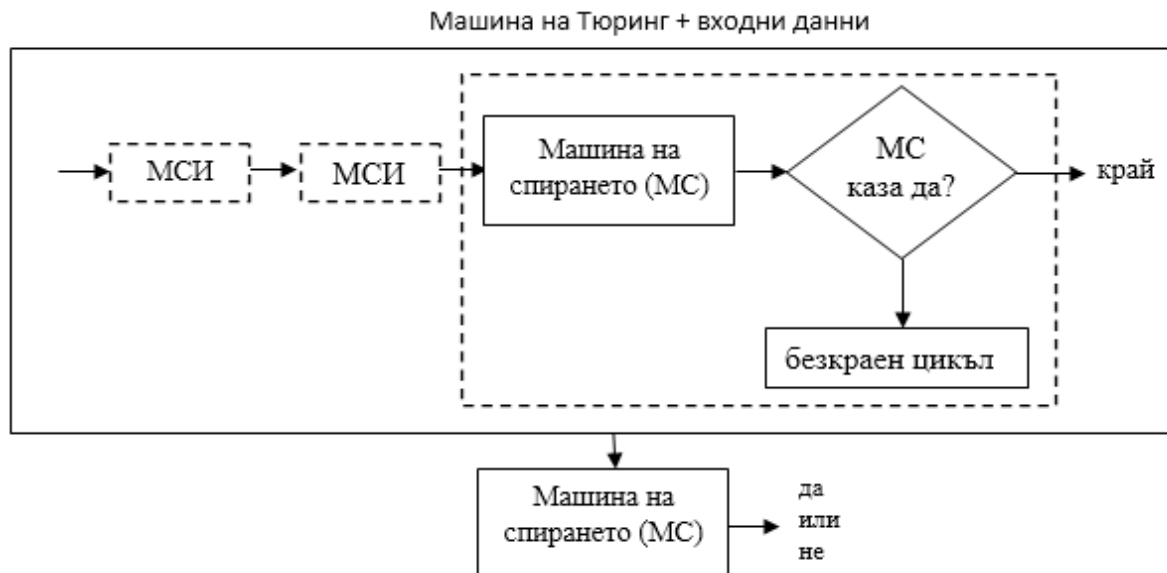
Фиг. 1. Машина на спирането

От МС създаваме втора машина, наречена МС Инвертор или МСИ. Тази машина инвертира отговора, даден от МС. На практика, ако МС отговори „да“, МСИ попада в безкраен цикъл; ако МС отговори „не“, МСИ спира (фигура 2).



Фиг. 2. МС Инвертор

Противоречието може да бъде видяно като попитаме МС дали МСИ прекратява изпълнението си, когато многократно му подаваме самия МСИ като входни данни (фигура 3). Многократно извикващият се МСИ, използващ себе си като входни данни, се използва като входни данни на МС. Докато двете машини действат при едни и същи входни данни, те трябва да връщат един и същи отговор: или машината на Тюринг прекратява изпълнението си при дадени входни данни, или не.



Фиг. 3. Проблем със спирането

Ако последната МС дава отговор „yes“ => машината, оперираща върху себе си, ще спре след определено време. Тогава копието от МС, намиращо се в МСИ, трябва също да дава отговор „yes“, но то дава отговор „no“.

Ако последната МС дава отговор „no“ => машината, оперираща върху себе си, ще попадне в безкраен цикъл. Тогава копието от МС, намиращо се в МСИ, трябва също да дава отговор „no“, но отговорът е „yes“.

От двете твърдения следва, че първоначалното твърдение, че МС съществува, е неправилно. Следователно няма машина на Тюринг, която може да разреши проблема със спирането [1].

Последна теорема на Ферма (Fermat's Last Theorem)

Уравнението

$$a^n + b^n = c^n \quad (1)$$

няма решение в положителни цели числа при $n > 2$. Това е едно от най-известните математически твърдения, наречено "Последна теорема на Ферма". Тя е обобщение на диофантовото уравнение

$$a^2 + b^2 = c^2, \quad (2)$$

известно и изследвано през древността. Свързано е с теоремата на Питагор и Питагоровите триъгълници. През 1996 г. официално е прието доказателството на Андрю Уайлс. Такава четворка естествени числа наистина не съществува. Доказателството е много сложно, съдържа се в 150 страници и проверката му е по силите на много малък брой математици [3]. За човек (и за компилатор), който не знае, че теоремата вече е доказана, ще бъде трудно да прецени дали програмата от фигура 4 ще намери решение и ще завърши или — не. Нещо повече, съществуват много подобни задачи, за които съответен теоретичен резултат не е известен и изобщо не се знае дали съответстващите програми ще зациклят [4].

```
void main() {
    unsigned a, b, c, i, x;
    for(x=3;;)
    {
        for(a=1; a<=x; a++)
            for(b=1; b<=x; b++)
                for(c=1; c<=x; c++)
                    for(i=3; i<=x; i++)
                        if((pow(a,i) + pow(b,i) == pow(c,i)) && a!=b && b!=c && a!=c)
                            cout << "a = " << a << "\nb = " << b << "\nc = " << c << endl;
        x++;
    }
}
```

Фиг. 4. Последната теорема на Ферма

Нерешени задачи

Алгоритъм на Колац (Collatz Conjecture)

Примерът от фигура 5 е известен програмен фрагмент. Дали програмата ще работи за всички възможни стойности на n ? Никой не знае отговора. При всички входни данни, при които е тестван алгоритъмът, програмата прекратява изпълнението си успешно [5].

```
while(n>1)
{
    if(n%2==1)
        n=3*n+1;
    else
        n/=2;
}
```

Фиг. 5. Алгоритъм на Колац

Това е т. нар. $3x + 1$ проблем или наричан още алгоритъм на Колац (Collatz conjecture, Collatz problem). Свързан е с поведението на итерациите на функцията, която на нечетните цели числа n дава стойност $3n + 1$, а на четните – стойност $n/2$ (3). Алгоритъмът на Колац твърди, че при започване с положително цяло число n , повторното итериране на тази функция все някога ще доведе до стойност 1.

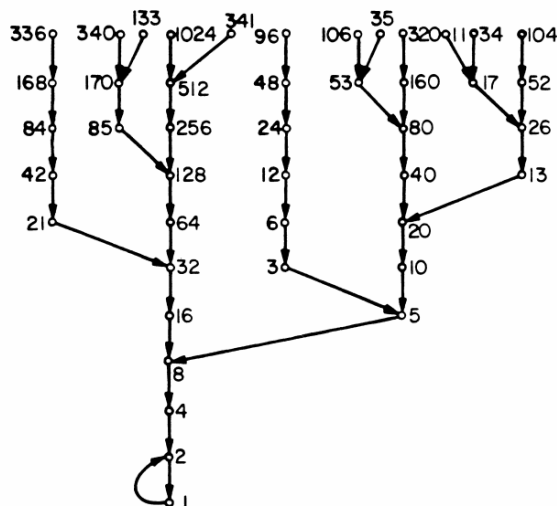
Например, ако започнем с

$n = 9: 9 \rightarrow 28 \rightarrow 14 \rightarrow 7 \rightarrow 22 \rightarrow 11 \rightarrow 34 \rightarrow 17 \rightarrow 52 \rightarrow 26 \rightarrow 13 \rightarrow 40 \rightarrow 20 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$.

Въпреки че е много лесно да се изкаже и разбере условието, но не е никак лесно да се докаже или опровергае твърдението. Този проблем все още не е разрешен. Той споделя свойствата си с други итерационни проблеми като аликуотните последователности, прословутите диофантови уравнения и последната теорема на Ферма.

$$T(n) = \begin{cases} \frac{3n+1}{2}, & \text{ако } n \equiv 1 \pmod{2} \\ \frac{n}{2}, & \text{ако } n \equiv 0 \pmod{2} \end{cases} \quad (3)$$

Един начин за разглеждането на $3x + 1$ проблема е свързан с насочен граф, чийто върхове са положителни цели числа и е насочен от n към $T(n)$ (фигура 6) [6].



Фиг. 6. Насочен граф, насочен от n към $T(n)$

Алгоритъмът на Колац е проверен със сигурност в диапазона $n < 2^{50} \approx 1,1 \times 10^{15}$ и завършва за краен период от време. Въпреки многобройните проведени изследвания и разсъждения, много въпроси остават. Известно е, че почти всички цели числа в крайна сметка достигат стойност 1 [7]. Проблемът обаче остава нерешен вече 80 години.

Аликуотна последователност (Aliquot Sequence)

В математиката аликуотна последователност е поредица от положителни цели числа, в които всеки член е равен на сумата от точните делители на предходния член. Ако поредицата достигне числото 1, тя свършва, тъй като сумата за 1 е 0. Аликуотна последователност, започваща с положително цяло число k , може да бъде дефинирана по следния начин:

$$s_0 = k \quad (4)$$

$$s_n = s(s_{n-1}) = \sigma_1(s_{n-1}) - s_{n-1} \text{ ако } s_{n-1} > 0 \quad (5)$$

$$\text{и } s(0) \text{ не е дефинирано.} \quad (6)$$

Например, аликуотна последователност на числото 20 е

$$\sigma_1(20) - 20 = 10 + 5 + 2 + 1 = 18 \quad (7)$$

$$\sigma_1(18) - 18 = 9 + 6 + 3 + 2 + 1 = 21 \quad (8)$$

$$\sigma_1(21) - 21 = 7 + 3 + 1 = 11 \quad (9)$$

$$\sigma_1(11) - 11 = 1 \quad (10)$$

$$\sigma_1(1) - 1 = 0 \quad (11)$$

Много аликутни последователности завършват на 0. Всички такива последователности задължително завършват с просто число, последвано от 1 (защото единственият точен делител на цяло число е 1), последвано от 0 (защото 1 няма точни делители). Има и няколко варианта, в които аликутна последователност може да не завърши никога [8].

- Аликутна последователност с период 1. Например числото 6:

$$\sigma_1(6) - 6 = 3 + 2 + 1 = 6 \quad (12)$$

⋮

- Аликутна последователност с период 2. Например числото 220:

$$\sigma_1(220) - 220 = 110 + 55 + 44 + 22 + 20 + 11 + 10 + 5 + 4 + 2 + 1 = 284 \quad (13)$$

$$\sigma_1(284) - 284 = 142 + 71 + 4 + 2 + 1 = 220 \quad (14)$$

⋮

- Аликутна последователност с период ≥ 3 . Например числото 1264460:

$$\sigma_1(1264460) - 1264460 = 632230 + 316115 + 252892 + 126446 + 74380 + 63223 + 37190 + 18595 + 14876 + 7438 + 3719 + 340 + 170 + 85 + 68 + 34 + 20 + 17 + 10 + 5 + 4 + 2 + 1 = 1547860 \quad (15)$$

$$\sigma_1(1547860) - 1547860 = 773930 + 386965 + 309572 + 154786 + 77393 + 8020 + 4010 + 3860 + 2005 + 1930 + 1604 + 965 + 802 + 772 + 401 + 386 + 193 + 20 + 10 + 5 + 4 + 2 + 1 = 1727636 \quad (16)$$

$$\sigma_1(1727636) - 1727636 = 863818 + 431909 + 3316 + 2084 + 1658 + 1042 + 829 + 521 + 4 + 2 + 1 = 1305184 \quad (17)$$

$$\sigma_1(1305184) - 1305184 = 652592 + 326296 + 163148 + 81574 + 40787 + 32 + 16 + 8 + 4 + 2 + 1 = 1264460 \quad (18)$$

⋮

Други. Например числото 95:

$$\sigma_1(95) - 95 = 19 + 5 + 1 = 25 \quad (19)$$

$$\sigma_1(25) - 25 = 5 + 1 = 6 \quad (20)$$

⋮

Проблемът $P=NP$

Проблемът $P = NP$ е много важен нерешен проблем в сферата на компютърните науки. Въпросът е дали всеки проблем, чието решение може да бъде потвърдено от компютър, може да бъде разрешено от компютър. Например вярно ли е, че сред числата -2, -3, 15, 14, 7, -10... има такива, чийто сбор е равен на 0? Отговорът е "да", защото $-2-3+15-10=0$ лесно се проверява с няколко сбороваия. Следва ли оттук, че е също толкова лесно да се подберат тези числа?

Отговорът на въпроса за равенството на класовете P и NP би установил дали проблеми, които могат да бъдат проверени за полиномиално време, могат да бъдат разрешени за

полиномиално време. Ако се окаже, че $P \neq NP$, това означава, че има проблеми в класа NP (като NP-пълните проблеми), които са по-трудни за решаване, отколкото за потвърждаване (не могат да бъдат разрешени за полиномиално време, но отговорът може да бъде потвърден за полиномиално време).

Освен в теорията на изчислимостта, доказателството би имало важно значение в сферата на математиката, криптографията, изкуствения интелект, философията, икономиката и много други.

Заклучение

Обикновено доказателството, че даден проблем е нерешим, слага край на десетилетия или столетия търсене на решение. Последната теорема на Ферма е формулирана около 1600 г., а нерешимостта ѝ е доказана през 1994 година. Често е по-трудно да се докаже, че нещо е невъзможно, отколкото обратното.

Неразрешени проблеми винаги е имало и най-вероятно винаги ще има. Както Paul Erdős казва по повод неразрешения $3x+1$ проблем: “Математиката все още не е готова за такива проблеми.”.

Със сигурност един ден проблемите, които затрудняват учените днес, ще бъдат разрешени, но дали на тяхно място няма да се появят други? Както е казал Karl Popper: „Всяко разрешаване на проблем повдига нови неразрешени проблеми.”.

Благодарности

Изказваме благодарности за подкрепата и съдействието по време на изработването на доклада на ас. инж. Нели Арабаджиева – Калчева.

Литература

- [1]. Lumetta, S. S. The Halting Problem. 2012, с. 1-2
- [2]. https://bg.wikipedia.org/wiki/Halting_проблем
- [3]. https://bg.wikipedia.org/wiki/Последна_теорема_на_Ферма
- [4]. Наков, П., П. Добриков. Програмиране++ Алгоритми. София, Software University, 2015, с. 358
- [5]. Shaffer, C. A. A Practical Introduction to Data Structures and Algorithm Analysis. Blacksburg, Virginia Tech, 2010, с. 587-588
- [6]. Lagarias, L. C. The $3x+1$ Problem and Its Generalizations. AT & T Bell Laboratories, Murray Hill, 1985, с. 3-5
- [7]. Lesieutre, J. On a Generalization of the Collatz Conjecture. Massachusetts Institute of Technology, Research Science Institute, 2007, с. 1
- [8]. https://en.wikipedia.org/wiki/Aliquot_sequence

За контакти:

Ирена Йорданова Георгиева
катедра „Компютърни науки и технологии“
Технически университет – Варна
E-mail: knownasirena@gmail.com

АЛГОРИТЪМ ЗА ПЪЛНО ТОПОЛОГИЧНО СОРТИРАНЕ НА АЦИКЛИЧЕН ГРАФ

Константин С. Георгиев

Резюме: Този доклад разглежда същността на топологичното сортиране при графите, както и неговото приложение при реални обстоятелства. Реализира се алгоритъм, който извежда списък с всички възможни случаи на топологично сортиране в даден граф, описан с матрица на съседство. Проверяват се резултатите от изпълнението на алгоритъма в програмна среда, като се анализира честотата на срещане на всеки от върховете в графа и след това се правят изводи спрямо разгледаната статистика.

Ключови думи: топологично сортиране, граф, алгоритъм

Algorithm for full topological sorting of an acyclic graph

Konstantin S. Georgiev

Abstract: This article views the essentials in topological sorting regarding graphs, as well as its appliances in real circumstances. An algorithm, which generates a list of all possible cases of topological sorting in a given graph, described with a matrix of adjacency, is put into effect. The results of the implemented algorithm are then tested in a programming environment through analyzing the frequency of occurrence in each of the nodes in the graph. After that, a conclusion is made based on the viewed statistics.

Keywords: topological sorting, graph, algorithm

1. Увод

Топологично сортиране на граф е операция, която може да се изпълнява само за ориентирани ациклични графи. Като резултат от тази операция се получава точно определена последователност, наречена топологичен ред. В този ред всички преки предшественици на даден връх трябва да се намират преди него или неговия номер. В един граф почти винаги съществуват няколко топологични реда.

Един топологичен ред може да се получи по следния алгоритъм (1):

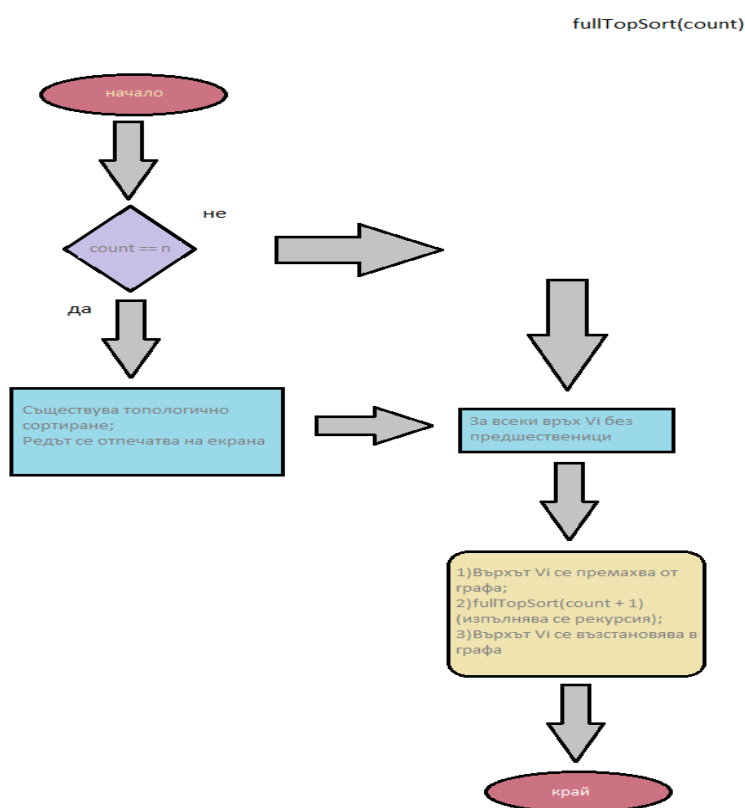
- 1) Ако графът няма върхове без предшественици, той е цикличен и не подлежи на топологично сортиране.
- 2) В произволен ред се записват върховете, които нямат предшественици и след това се изтриват.
- 3) Ако има останали неизтрети върхове, се преминава в стъпка 1), в противен случай топологичното сортиране е завършено [2].

Съществува и още един алгоритъм (2) за топологично сортиране на граф със същата сложност като предишния, но е в известна степен, обратен на него. Използва се функция за обхождане в дълбочина (Depth-First Search) за намиране на върхове на графа, които са без наследници и така сортирането се получава в обратен ред [3].

Топологичното сортиране намира широко приложение в случаи, когато трябва да се изграждат системи от данни, където подредбата е от важно значение. Като пример може да се даде един учебник по математика. При доказателството на една теорема се използват и други дефиниции, теореми и аксиоми. Учебникът може да се счита за топологично сортиран, само ако всяка теорема цитира информация, която е била дефинирана или доказана по-рано.

2.Алгоритъм и схема за изпълнение

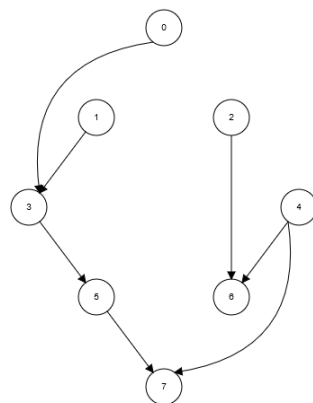
Задачата е да се направи пълно топологично сортиране на даден ацикличен граф. За да се осъществи това, е необходимо да се изгради множество от списъци, като във всеки списък се разположи един възможен случай на сортиране. Прилагайки алгоритъм (1) при търсене на топологично сортиране, на всяко ниво се избира връх без предшественици, а в случай, в който те са няколко, се избира произволен. От тази произволност на избора следват и различните решения на задачата. Така, когато е налице повече от една възможност, трябва последователно да се изпробват всички, до достигане на пълно изчерпване [1]. Нека тази функция да се нарече `fullTopSort()`. При разглеждане на всеки връх графът ще се модифицира, като първо този връх ще се изтрие и след това рекурсивно ще се извика същата функция. След връщането от рекурсията следва възстановяване на графа и съответно повтаряне на същия процес със следващия връх. Дъното на рекурсията се достига тогава, когато списъкът със сортиранията съдържа n на брой върхове, след което се отпечатва намереното решение. Функцията `fullTopSort()` ще бъде изпълнена по следната схема (фигура 1):



Фиг. 1. Блок-схема на функцията `fullTopSort()`

3. Задаване на граф и на неговата матрица на съседство

Нека за изпълнение на задачата се използва следният граф (фигура 2):



Фиг. 2. Зададен граф

Матрицата на съседство показва инцидентността на всеки от върховете в графа. Тя е необходима за неговото представяне в програмна среда чрез съхраняването на матрицата в двумерен масив.

	0	1	2	3	4	5	6	7
0				1				
1				1				
2							1	
3						1		
4							1	1
5								1
6								
7								

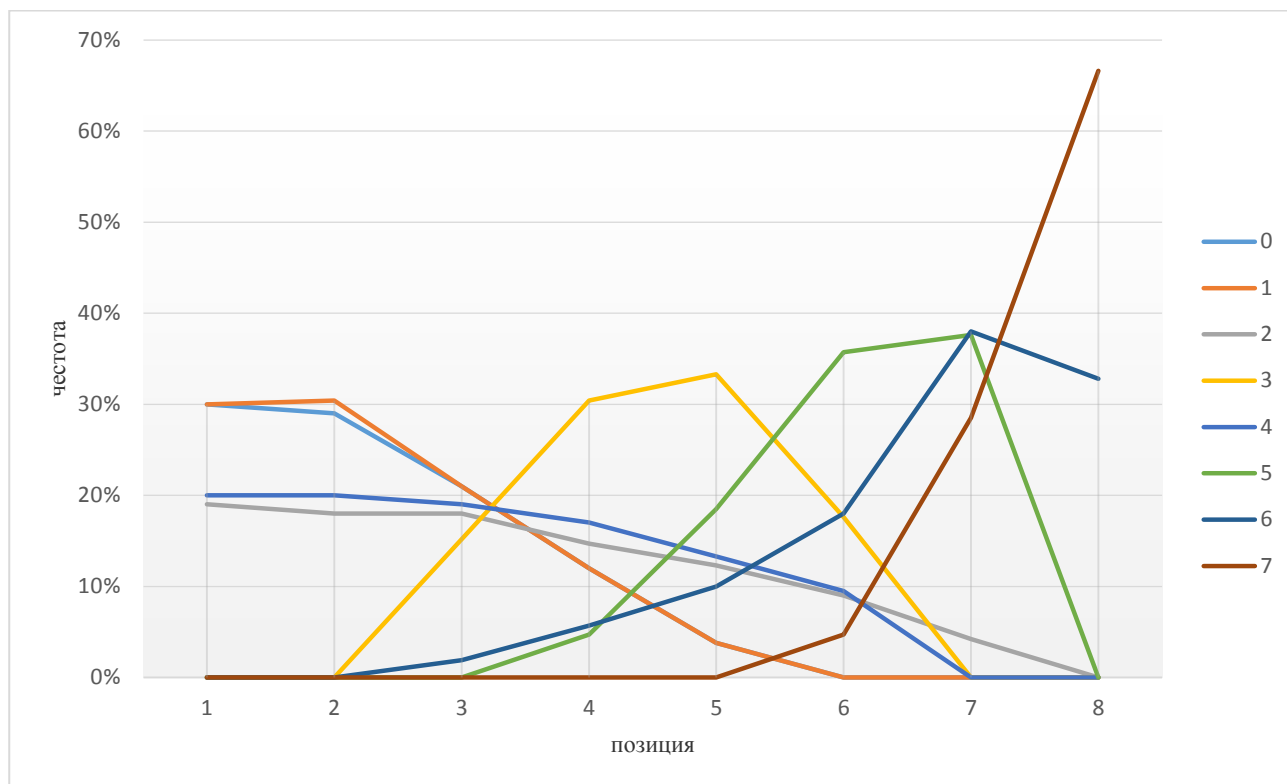
Фиг. 3. Матрица на съседство на зададеният граф

4. Резултати и анализ

Резултатите показват, че зададеният граф има 210 възможни различни топологични сортирания. Честотата на срещане на върховете в графа във всяка една позиция в списъците е следната (таблица 1):

Таблица 1. Статистика на върховете в графа

		Върхове в графа							
Позиция		0	1	2	3	4	5	6	7
	1	30%	30%	19%	0%	20%	0%	0%	0%
	2	29%	30.4%	18%	0%	20%	0%	0%	0%
	3	21%	21%	18%	15.2%	19%	0%	1.9%	0%
	4	12%	12%	14.7%	30.4%	17%	4.7%	5.7%	0%
	5	3.8%	3.8%	12.3%	33.3%	13.3%	18.5%	10%	0%
	6	0%	0%	9%	17.6%	9.5%	35.7%	18%	4.7%
	7	0%	0%	4.2%	0%	0%	37.6%	38%	28.5%
	8	0%	0%	0%	0%	0%	0%	32.8%	66.6%



Фиг. 4. Графика за честотата на срещане на върховете в графа (по абцисата x са разположени позициите на върховете, а по ординатата y – процентът честота)

От направените изследвания на списъците се вижда, че връх номер 2 (връх без предшественици) може да се разположи в най-много различни позиции при сортировката. Това се дължи на факта, че връхът 2 е съседен само на един връх (номер 6), който от своя страна е без наследници. Това означава, че връхът 2 е зависим единствено и само от този връх и не може да се намира след него в списъка. От друга страна, връхът без наследници - номер 7, който се намира в дъното на графа, може да се постави само в последните две позиции, като също така има и най-висок процент честота в една позиция спрямо другите върхове.

Графиката на честотите съответно показва, че:

- 1) При върховете без предшественици кривата е намаляваща;
- 2) При върховете без наследници кривата е нарастваща;
- 3) При върховете с предшественици и наследници кривата равномерно нараства и след това спада.

От тези данни могат да се направят следните изводи:

- 1) Връх, който няма предшественици, но има наследници, не може да присъства в края на списъка със сортиранията;
- 2) Връх, който няма наследници, но има предшественици, не може да присъства в началото на списъка;
- 3) Връх, който има наследници и предшественици, не може да присъства нито в началото, нито в края на списъка.

5. Заключение

Топологичното сортиране не е сред най-широко използваните алгоритми в програмирането. За сметка на това обаче, неговите правила относно разположението на един елемент в цяла система от различни или еднотипни такива, гарантират добра организация и

лесна навигация. Реализираният алгоритъм показва функционалността на тези правила като ги прилага при зададен граф.

Литература

- [1]. Наков, П., Добриков, П., Програмиране++Алгоритми, София, TopTeam Co., 2002, стр.320-324;
- [2]. <http://www.pg-vpeev.com/wp-content/uploads/2011/12/20-Graph.pdf>
- [3]. <http://home.cse.ust.hk/faculty/golin/COMP271Sp03/Notes/MyL08.pdf>
- [4]. <https://www.cs.usfca.edu/~galles/visualization/TopoSortDFS.html>

За контакти:

Константин С. Георгиев
катедра „Компютърни науки и технологии“
Технически университет – Варна
E-mail:dragonflarex@mail.bg



СИСТЕМА ЗА СИМУЛИРАНЕ НА БОТНЕТ DOS АТАКИ

Юлия А. Алексиева

Резюме: Ботнет е бързо разрастваща се заплаха за съвременната компютърна сигурност. IRC ботнет се използва за големи по мащаб DoS атаки. Съществуват различни техники за засичане на ботнет, но не и за превенция на тяхното действие. За разработването на качествена техника за цялостна защита срещу ботнет атаки и заразяване е нужна добра симулационна среда, която да обхваща качествено цялостната функционалност на ботнет DoS атаките. В доклада е представена реализирана система за симулиране на ботнет DoS атаки, която може да осъществява DoS атаки с различни параметри. Представени са две тестови постановки на системата и са анализирани резултатите.

Ключови думи: ботнет, DoS атаки, TCP атака, UDP атака, ICMP атака, мрежова сигурност

Botnet DoS attack simulation system

Yuliya A. Aleksieva

Abstract: Botnet is a fast growing thread to the modern computer security. IRC botnet is used for large scale DoS attacks. There are different techniques to detect botnet attacks and infections, but they do not have the functionality to prevent malicious botnet activity. The development of a quality equipment for full protection from botnet attacks needs a good simulation environment that qualitatively meets the functionality of botnet DoS attacks. The paper presents a realised system for simulating botnet DoS attacks, which can create DoS attacks with various parameters. Performed are two performance experiences and the obtained results are analyzed.

Keywords: botnet, DoS attacks, TCP attack, UDP attack, ICMP attack, network security

1. Увод

Ботнет имат богата история, въпреки че представляват най-скоро появилата се Интернет заплаха, но за сметка на това и най-развиваща и усложняваща се и все по-широко разпространена. Във времето ботнет са претърпели развитие от прости генератори на спам до основно средство за масови криминални операции. Ботнет са използвани в кражби на самоличности и номера на кредитни карти, различни форми на шпионаж, Denial of Service (DoS - отказ на обслужване) атаки, атаки към несигурни продукти от новия дигитален свят – поставяйки в опасност модерното общество и развиващата се икономика.

Бот е машина, на която зловреден компютърен софтуер е бил инсталиран, като той може да се контролира самостоятелно и автоматично. Ботнет представлява група от ботове (съвкупност от заразени - зомби компютри), изпълняващи зловреден софтуер, бивайки контролирани от един хакер – общоизвестен като botmaster (главен бот), чрез централизирана инфраструктура Command&Control център на атаката [1].

Комбинацията на Ботнет със съвременни технологии като Internet Relay Chat (IRC), Hypertext Transfer Protocol (HTTP) и Peer-to-Peer (P2P) дава възможност да организират скритата си тактика в легални потребителски приложения, превръщайки засичането на ботнет атаките много по-сложно. IRC ботнет атаките са най-мощната ботнет заплаха, като към настоящата година атаките са нарастнали със 132% повече от миналата година[6].

Проблемът с нарастващите DoS IRC ботнет атаки касае съвременния компютърно-базиран бизнес, тъй като основното средство за вътрешна комуникация както в големи (Google, Yahoo, LinkedIn, Facebook), така и в по-малки бизнес корпорации (Six Apart, Last.fm)

се базира на използването на IRC клиенти и канали. Разменяната информация по тези канали е конфиденциална, което прави тези канали добра мишена за атакуване.

Съществуват техники за засичане на ботнет атаки и заразявания, но те нямат функционалността да предотвратят зловредното ботнет действие. Това ги превръща в недостатъчно ефективни средства за защита, тъй като не са достатъчно всеобхватни. За разработването на качествена техника за засичане и премахване на ботнет заразяване е нужна добра симулационна среда, която да е безопасна и напълно да отговаря на функционалността на ботнет DoS атаките. Наличието на такава симулационна среда улеснява разработването на техники за защита, те ще могат да се тестват, давайки реални резултати, също така ще може да се разработват качествени и ефективни средства за предпазване от ботнет атаки [1].

2. Анализ на съществуващи решения

Главна цел за съвременната мрежова сигурност е да бъдат създадени адекватни техники за откриването и евентуалното прекратяване на ботнет DoS заплахите. Това налага нуждата от подробното им познаване. Вникването в същността на ботнета и разбиране на дейността му ще доведе до ефикасна и разумна техника за засичане на ботнет. Налага се необходимостта от създаване на качествени специализирани приложения за симулации на ботнет DoS атаки.

Към момента съществуват разнородни средства за генериране на различни DoS flood атаки, но са тясно специализирани към изпълнението на определен вид атака без възможност за реализиране на останалите. Малко на брой са наличните средства, които генерират различни DoS атаки, а още по-малко тези, които генерират ботнет DoS атаки [3]. Ето някои от съществуващите генератори и техните основни характеристики:

1. LOIC (Low Orbit Ion Canon)

LOIC е един от най-популярните инструменти за DoS атака свободно достъпни в Интернет. Може да се използва само от един потребител да извърши DoS атака на малки сървъри. Изпълнява атака чрез UDP flood, TCP flood или HTTP заявки към сървър-жертва. Нужен е само IP адресът на сървър. Недостатъците на това средство са: не реализира DoS атака като ботнет, а от единичен IP адрес; не скрива IP адреса и той може лесно да бъде проследен.

2. XOIC

Средство, което извършва (D)DoS атаки към подаден IP адрес, като потребителят избира порт и протокол. Води се за по-мощно от LOIC. Има три нива на използване, като основното е за нормална DoS атака (TCP/HTTP/UDP/ICMP). Използва се за тестване на издръжливостта на уебсайтове. Недостатъкът в този случай е същият – реализира DoS атака от една машина, а не като ботнет. Също така средството не е симулатор, а реална система за атакуване на сигурността, което го прави опасно дори и за тестови цели.

3. DDOSIM

Генератор, който реализира HTTP flood и TCP flood като симулира наличието на няколко бот хоста със случайни IP адреси. Написан е на C++ и работи под Linux. Средството е с отворен код. Средството генерира атаки от една машина, като симулира, че действието се извършва от различни хостове, което не е реална ботнет симулация, т.к. се разполага само с ресурсите на машината, на която е стартиран, за генериране на атаки.

4. MAZEBOLT

Платформа за тестване – реализира flood атаки, ботнет атаки, атаки на приложно ниво. Средството се използва изцяло за тестване, но е платено. Симулира SYN flood, UDP flood и

ботнет атаки, различни от DoS. Средството е платено и съответно със затворен код, което ограничава възможностите за използването му и съответно внасяне на подобрения.

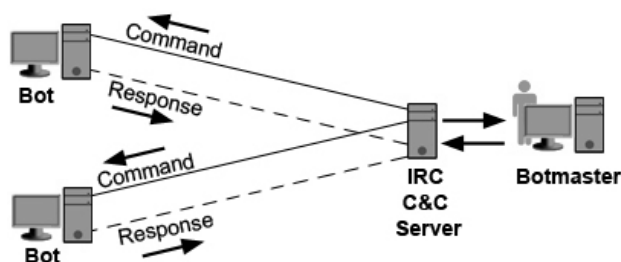
5. BoNeSi [7]

DDoS ботнет симулатор, който симулира трафик в затворена среда с цел проучване на ефекта от този тип атаки. Генерира ICMP, UDP и TCP flood атаки от ботнет с даден размер – различни IP адреси. Средството е с отворен код, но е изцяло конзолно. Като DDOSIM, генерира атаки от една машина, като симулира, че действието се извършва от различни хостове. Друг недостатък е и че приложението работи само под Linux операционна система.

Обобщеният анализ показва, че съществуващите решения за DoS симулатори са реализирани за частни случаи и определени атаки. Разгледаните генератори не съдържат пълен набор от възможности за атаки, имат системни ограничения и зависимости, не отговарят напълно на нуждите на анализаторите на мрежовата сигурност. Липсва разработен генератор на атаки, който да може да симулира реална DoS ботнет атака и съответно да могат да се генерират различни атаки към и от множество ботове.

3. IRC ботнет

IRC ботнет е най-разпространената ботнет атака, тъй като най-лесно може да се реализира и изключително трудно се засича наличието на зараза. IRC е чат система, която осигурява едно-към-едно и едно-към-много мигновени съобщения през Интернет [5]. Действието на IRC ботнета е показано на фигура 1.



Фиг. 1. Действие на IRC ботнет

IRC се използва като инфраструктурен C&C протокол, чрез който главният бот разпространява команди към своята бот армия. Бот мастърът се крие зад IRC сървъра и подава атаки към ботовете чрез IRC канали, в които собствениците на бот машините не знаят, че присъстват. Нападателят може да използва или публичен IRC сървър, или да изгради свой собствен сървър за целите на C&C. Ботовете са конфигурирани да се свържат обратно към C&C сървъра, очаквайки от който и да е конфигуриран порт да получат команди и да я изпълнят.

IRC ботнет се използва за реализиране на DoS атаки, тъй като бот мастърът лесно може да управлява атаките, скрит зад IRC технологията и трудно може да бъде засечен [2]. Тази атака е предпочитана поради факта, че е почти невъзможно да се разбере, че дадена машина е част от ботнет и съответно да бъде спряна, понеже IRC ботнет разчита на голям набор от зомби компютри, които да реализират малки части от атаката, но цялостно да се получи голямо натоварване. Възможностите на единичната бот машина не се използват максимално, извършвайки атаката и така не се натоварват ресурсите на бот машината. Целта на IRC ботнет е да има голяма армия от ботове, които едновременно да реализират атака към дадена

жертва. Така собствениците на бот машините няма как да разберат за наличие на зараза и да вземат мерки за изключване на бота от мрежата.

4. Реализация на системата за симулация

Представената система за симулиране на ботнет DoS атаки има архитектура с разпределен характер. Системата се състои от множество ботове, изпълняващи се на отделни машини и комуникиращи си през IRC канали.

При наличната система ботовете имат инсталиран скрипт, който осъществява тяхната бот същност. Той е един и същ за всяка машина и работи, докато ботът не бъде изключен. За да може да се симулира действието на IRC ботнет, всяка нужна машина трябва да има инсталиран този скрипт. Тоест, приема се, че ботнет мрежата е вече създадена – компютрите са били заразени. Системата може да действа с поне един свързан бот към мрежата до неограничен брой ботове. Скриптът се пуска локално на всяка машина и се контролира чрез IRC канала, към който се свързва. Този IRC канал представлява C&C центъра на ботнета. Управлението на ботнет DoS атаките става отдалечено чрез подаване на команди в зададения IRC канал. Всички ботове получават една и съща команда и всеки я изпълнява независимо от останалите с цел по-голяма ефективност на атаката. След изпълнение на атаката ботовете използват IRC протокола и се свързват към потребителя на канала, който е задал командата и му изпращат лично съобщение с отчет за извършената дейност. Така отчетът е видим само за атакувания, но не и за останалите потребители в канала.

За реализация на системата се използва IRC канал, специално създаден за тази цел. Този канал не се използва от легитимни потребители на IRC технологията. Освен това се използва локално инсталиран IRC сървър, а не публичен, тъй като използването на публичен IRC сървър би нарушило етикета за използване на Интернет и може да доведе до проблеми в реално използвани услуги. Локално инсталираният сървър не е достъпен за легитимни потребители на IRC технологията за комуникация, с което се запазва сигурността и затвореността на системата.

Скриптът, реализиращ бот функционалността на машините, се стартира на хостовете, участващи в изследването. Скриптът е програмиран така, че ботовете да се включват към IRC, използвайки IRC сървъра, който е инсталиран в мрежата, а не публичен. След като ботовете се свържат към сървъра, те се включват в зададения IRC канал. След това управлението на ботнета се осъществява на друга машина, където потребителят използва IRC клиент, за да се свърже с IRC сървъра и да се включи в канала, където ботовете слушат за команди.

Всеки бот има следните функционалности:

- UDP flood

Потребителят задава хоста, към който е насочена атаката, порта, към който да се извърши атаката и продължителността на атаката в секунди. Когато атаката се реализира, към главния бот се връща информация с това колко конекции са се осъществили от бота към атакуваната машина.

- TCP flood

Потребителят задава хоста, към който е насочена атаката, порта, към който да се извърши атаката, броя на връзките, които да се изпратят и дължината на данните, които да се изпратят. Когато атаката приключи, ботът изпраща на главния бот отчет за дължината на атаката в секунди.

- ICMP flood

Потребителят задава хоста, към който е насочена атаката, и броя пингове, които да се изпълнят. Атаката използва директно възможността за изпълняване на команди и изпълнява командата ping. Целта на атаката е да се изпратят голям брой ехо пакети към жертвата.

- Изпълнение на команди

Използва се вградената в скриптовия език команда за изпълнение на команди от командния ред. Потребителят трябва да подаде команда с нужните параметри за изпълнението ѝ и ботовете ще я изпълнят.

- Лог информация

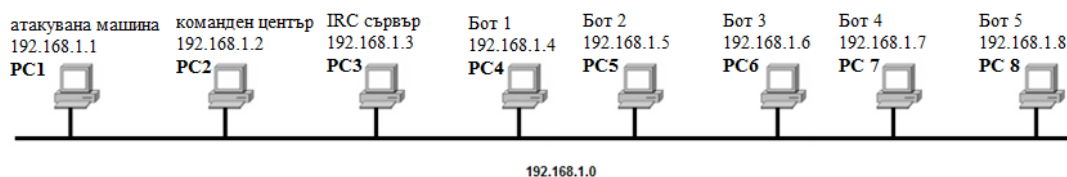
След всяко изпълнение на команда или атака ботовете съобщават статуса си на лично съобщение на потребителя. Всеки бот изпраща съобщение с отчет за това колко пакета е изпратил/ колко време е продължила атаката/ и какъв е резултатът от изпълнена команда.

Необходимостта от комбинативност по отношение на изпълнението на различните видове атаки изисква всеки вид атака да може да бъде изпълнявана автономно с максимални възможности за параметризация. За всяка атака е предвидена отделна функция, приемаща като аргумент структура, съдържаща в себе си различни аргументи, стойностите на които влияят върху характеристиките на изпълняваната атака.

5. Тестване на системата и резултати

Реализираната система е тествана чрез различни тестови постановки – различни мрежови инфраструктури (равнинна и маршрутизирана), различно разпределение на машините като роли в системата, различен брой заразени машини. В настоящия доклад са разгледани две тестови постановки на машините – в равнинна топология с различен брой заразени машини – 3 и 5, атакуващи една машина в мрежата. Целта на двете постановки е да покаже как минимална разлика в броя на ботовете се отразява върху мащаба на атаката. На фигура 2 е представено разпределението на машините при тестова постановка 2, реализирана с 5 бота, като при тестова постановка 1 разпределението е аналогично – без бот 4 и бот 5.

При равнинна топология ботовете, командният център, IRC сървърът и атакуваната машина са в една мрежа. Целта е да се определи влиянието на атаките и качеството на пропускателната способност на мрежата с един канал до атакуваната машина, когато я атакуват 3 бота и когато я атакуват 5 бота.



Фиг. 2. Равнинна топология с пет бота

Командният център и атакуваната машина са под управлението на Windows 7 Enterprise, а ботовете – под Slackware Linux 3.10. Експерименталната платформа е базирана на Intel Core2Duo T7700, работещ на 2.4 GHz тактова честота и 4GB RAM. За целта на тестовите постановки е използван UnrealIRCd IRC сървър с отворен код, който позволява персонална конфигурация. Това се налага поради факта, че използване на публичен IRC сървър би довело до конфликти и нарушава правилата за правилна комуникация в IRC средата. Като IRC клиент е използван клиентът Xchat.

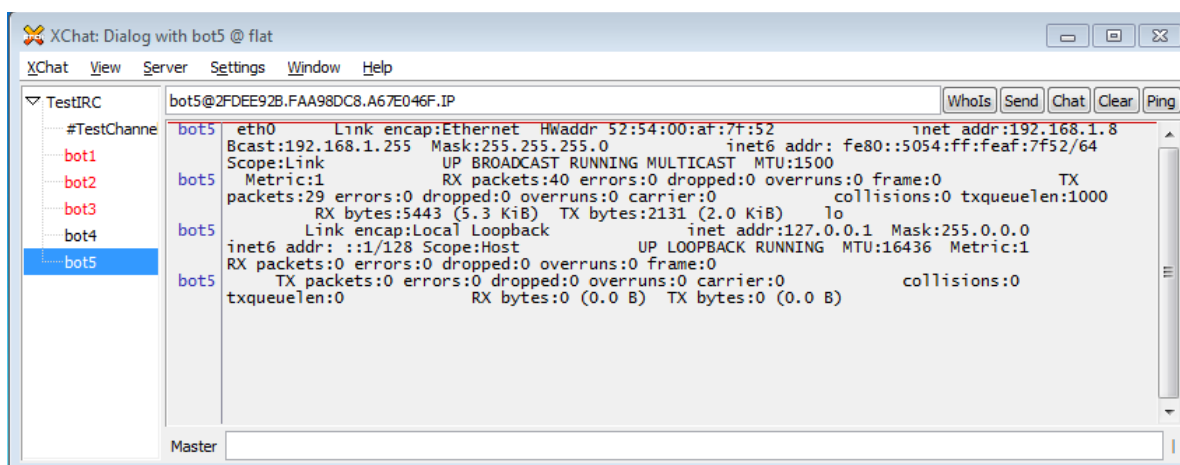
Тестовите са извършени в две насоки: за тестване на комуникацията между ботите и за реализация на DoS атаки. Ботите се стартират на съответните машини и при стартирането си се включват в тестовия канал, през който атакуващият бот задава командите.

Първата група тестове включва тестване на връзката между ботите – присъединяване в канала и изпращане на командата !say, последвана от текст. Всички ботове трябва да повторят текста и да го изведат на екрана на клиента. На фигура 3 са показани резултатите от теста с текст „Hello world!“.



Фиг. 3. Свързване на ботите в канала и изпълнение на команда !say Hello world!

Вторият тест проверява възможностите на ботите да изпълняват локално команди на операционната система чрез изпратена им от атакуващия бот команда !cmd, последвана от командата на ОС с нужните параметри за изпълнение. След изпълнението ботите изпращат резултата в персонално съобщение към атакуващия бот. Така резултатът е видим само за атакуващия бот, но не и за останалите потребители на канала. Резултатите от тест за изпълнение на команда на ОС ifconfig са показани на фигура 4.



Фиг. 4. Резултат от изпълнението на команда !cmd ifconfig от бот 5 към главния бот

Тази група тестове показват функционалността на ботите. Следва да се отбележи, че резултатите са аналогични и при двете тестови постановки.

Втората група тестове изследва функционалността на различните симуирани DoS атаки. Целта е да се определи ефикасността на симулационната среда за насищане на комуникационния канал с пакети. Атаките са осъществени по отношение на трите базови протокола.

Първият тип атака е атаката по TCP, изпълнена по порт 80 (HTTP), с обем 1000 пакета с дължина по 64 байта. На фигура 5 е представена извадка от отчета на Wireshark, стартиран на атакуващата машина. Атаката по ICMP включва изпълнение на определен брой команди ping

от командния ред на машината на всеки бот. На фигура 6 е представена извадка на Wireshark по време на ICMP атаката, като се вижда че и петте бота са свързани и изпращат Echo (ping) request. Атака по UDP към определен хост по порт 53 (DNS) с продължителност 120 сек. Пакетите съдържат символа 'X'.

No.	Time	Source	Destination	Protocol	Length	Info
6	0.00120700	192.168.1.4	192.168.1.1	TCP	74	33938-80 [SYN] Seq=0 win=4380 Len=0 MSS=1460 SACK_PERM=1 TSval=3419042 TSecr=0 WS=32
7	0.00121500	192.168.1.6	192.168.1.1	TCP	74	46869-80 [SYN] Seq=0 win=4380 Len=0 MSS=1460 SACK_PERM=1 TSval=3443211 TSecr=0 WS=32
8	0.00122000	192.168.1.7	192.168.1.1	TCP	74	53119-80 [SYN] Seq=0 win=4380 Len=0 MSS=1460 SACK_PERM=1 TSval=966029 TSecr=0 WS=32
9	0.00132700	192.168.1.5	192.168.1.1	TCP	74	36061-80 [SYN] Seq=0 win=4380 Len=0 MSS=1460 SACK_PERM=1 TSval=3276081 TSecr=0 WS=32
14	0.00162200	192.168.1.8	192.168.1.1	TCP	74	57067-80 [SYN] Seq=0 win=4380 Len=0 MSS=1460 SACK_PERM=1 TSval=547582 TSecr=0 WS=32
19	0.00176800	192.168.1.1	192.168.1.4	TCP	54	80->33938 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
20	0.00179000	192.168.1.1	192.168.1.5	TCP	54	80->36061 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
21	0.00181100	192.168.1.1	192.168.1.6	TCP	54	80->46869 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
23	0.00184400	192.168.1.1	192.168.1.7	TCP	54	80->53119 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
25	0.00202700	192.168.1.1	192.168.1.8	TCP	54	80->57067 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
26	0.00233700	192.168.1.4	192.168.1.1	TCP	74	33939-80 [SYN] Seq=0 win=4380 Len=0 MSS=1460 SACK_PERM=1 TSval=3419043 TSecr=0 WS=32
27	0.00238400	192.168.1.1	192.168.1.4	TCP	54	80->33939 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
28	0.00240900	192.168.1.6	192.168.1.1	TCP	74	46870-80 [SYN] Seq=0 win=4380 Len=0 MSS=1460 SACK_PERM=1 TSval=3443212 TSecr=0 WS=32
29	0.00244800	192.168.1.1	192.168.1.6	TCP	54	80->46870 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
30	0.00250000	192.168.1.7	192.168.1.1	TCP	74	53120-80 [SYN] Seq=0 win=4380 Len=0 MSS=1460 SACK_PERM=1 TSval=966031 TSecr=0 WS=32
31	0.00250700	192.168.1.5	192.168.1.1	TCP	74	36062-80 [SYN] Seq=0 win=4380 Len=0 MSS=1460 SACK_PERM=1 TSval=3276082 TSecr=0 WS=32
32	0.00254000	192.168.1.1	192.168.1.7	TCP	54	80->53120 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
33	0.00256500	192.168.1.1	192.168.1.5	TCP	54	80->36062 [RST, ACK] Seq=1 Ack=1 win=0 Len=0
34	0.00264200	192.168.1.8	192.168.1.1	TCP	74	57068-80 [SYN] Seq=0 win=4380 Len=0 MSS=1460 SACK_PERM=1 TSval=547583 TSecr=0 WS=32
35	0.00266800	192.168.1.1	192.168.1.8	TCP	54	80->57068 [RST, ACK] Seq=1 Ack=1 win=0 Len=0

Фиг. 5. Отчет от Wireshark за трафика минаващ през атакуваната машина 192.168.1.1 по време на TCP атаката

No.	Time	Source	Destination	Protocol	Length	Info
5	0.01580600	192.168.1.4	192.168.1.1	ICMP	98	Echo (ping) request id=0x071a, seq=1/256, ttl=64 (reply in 8)
8	0.01646000	192.168.1.1	192.168.1.4	ICMP	98	Echo (ping) reply id=0x071a, seq=1/256, ttl=128 (request in 5)
9	0.01888200	192.168.1.5	192.168.1.1	ICMP	98	Echo (ping) request id=0x0745, seq=1/256, ttl=64 (reply in 12)
12	0.01944000	192.168.1.1	192.168.1.5	ICMP	98	Echo (ping) reply id=0x0745, seq=1/256, ttl=128 (request in 9)
13	0.02055000	192.168.1.6	192.168.1.1	ICMP	98	Echo (ping) request id=0x06fd, seq=1/256, ttl=64 (reply in 16)
16	0.02108500	192.168.1.1	192.168.1.6	ICMP	98	Echo (ping) reply id=0x06fd, seq=1/256, ttl=128 (request in 13)
17	0.05761300	192.168.1.8	192.168.1.1	ICMP	98	Echo (ping) request id=0x070a, seq=1/256, ttl=64 (reply in 20)
20	0.05824600	192.168.1.1	192.168.1.8	ICMP	98	Echo (ping) reply id=0x070a, seq=1/256, ttl=128 (request in 17)
21	0.06818800	192.168.1.7	192.168.1.1	ICMP	98	Echo (ping) request id=0x070a, seq=1/256, ttl=64 (reply in 24)
24	0.06882900	192.168.1.1	192.168.1.7	ICMP	98	Echo (ping) reply id=0x070a, seq=1/256, ttl=128 (request in 21)
25	0.01736600	192.168.1.4	192.168.1.1	ICMP	98	Echo (ping) request id=0x071a, seq=2/512, ttl=64 (reply in 26)
26	0.01745900	192.168.1.1	192.168.1.4	ICMP	98	Echo (ping) reply id=0x071a, seq=2/512, ttl=128 (request in 25)
27	0.02025800	192.168.1.5	192.168.1.1	ICMP	98	Echo (ping) request id=0x0745, seq=2/512, ttl=64 (reply in 28)
28	0.02034200	192.168.1.1	192.168.1.5	ICMP	98	Echo (ping) reply id=0x0745, seq=2/512, ttl=128 (request in 27)
29	0.02175200	192.168.1.6	192.168.1.1	ICMP	98	Echo (ping) request id=0x06fd, seq=2/512, ttl=64 (reply in 30)
30	0.02187800	192.168.1.1	192.168.1.6	ICMP	98	Echo (ping) reply id=0x06fd, seq=2/512, ttl=128 (request in 29)
31	0.05907400	192.168.1.8	192.168.1.1	ICMP	98	Echo (ping) request id=0x070a, seq=2/512, ttl=64 (reply in 32)
32	0.05917600	192.168.1.1	192.168.1.8	ICMP	98	Echo (ping) reply id=0x070a, seq=2/512, ttl=128 (request in 31)
33	0.06956000	192.168.1.7	192.168.1.1	ICMP	98	Echo (ping) request id=0x070a, seq=2/512, ttl=64 (reply in 34)
34	0.06964700	192.168.1.1	192.168.1.7	ICMP	98	Echo (ping) reply id=0x070a, seq=2/512, ttl=128 (request in 33)

Фиг. 6. Отчет от Wireshark от атакуваната машина 192.168.1.1 за ICMP атаката

Информацията за резултатите от трафика за всяка атака се получава от различни отчети на WireShark, стартиран на атакуваната машина. Резултатите от двете постановки са обобщени в таблица 1, където са представени средните резултати от експериментите за двете постановки, получени на базата на обобщените отчети на анализатора.

Display Display filter: udp.port eq 53 Ignored packets: 0 (0,000%)						Display Display filter: tcp.port eq 80 Ignored packets: 0 (0,000%)					
Traffic	Captured	Displayed	Displayed %	Marked	Marked %	Traffic	Captured	Displayed	Displayed %	Marked	Marked %
Packets	600	248	41,333%	0	0,000%	Packets	10108	10059	99,515%	0	0,000%
Between first and last packet	227,178 sec	162,978 sec				Between first and last packet	93,433 sec	88,680 sec			
Avg. packets/sec	2,641	1,522				Avg. packets/sec	108,185	113,431			
Avg. packet size	885 bytes	586 bytes				Avg. packet size	64 bytes	64 bytes			
Bytes	531246	145328	27,356%	0	0,000%	Bytes	647466	644366	99,521%	0	0,000%
Avg. bytes/sec	2338,455	891,702				Avg. bytes/sec	6929,772	7266,234			
Avg. MBit/sec	0,019	0,007				Avg. MBit/sec	0,055	0,058			

Фиг. 7. Обобщен отчет от Wireshark за проведените TCP и UDP атаки

Таблица 1. Средни стойности, предоставени от отчета на Wireshark за тестова постановка 1 и 2

Атаки	пакети/секунда		размер на пакета		байтове/секунда	
	Тест 1	Тест 2	Тест 1	Тест 2	Тест 1	Тест 2
UDP атака	1,7 p/s	2,6 p/s	897 B	885 B	1550 B/s	2339 B/s
TCP атака	569 p/s	108 p/s	64 B	64 B	36407 B/s	6930 B/s
ICMP атака	3,4 p/s	5,6 p/s	98 B	97 B	335 B/s	544 B/s

В таблица 1 са представени средните резултати от отчета от Wireshark стартиран на атакуваната машина при постановка 1 – с три бота и при постановка 2 – с пет бота. При средния размер на пакетите няма големи разминавания в стойностите при двете постановки. При средните стойности на брой пакети в секунда и брой байтове в секунда стойностите се разминават значително – стойностите се покачват с около 40% при тест 2, дължащо се на факта, че в мрежата участват с 2 бота повече, които увеличават трафика по мрежата и съответно могат да нанесат повече щети към атакуваната машина.

Тестовите резултати са получени на базата на проведените тестови постановки и е възможно да се различават от резултатите, които биха били събрани на базата на тестове, проведени върху други машини, с различни параметри, операционни системи, нива на защита, среда за пренос на данни, натоварвания на мрежата.

5. Заключение

В настоящия доклад е представена реализирана система за симулиране на ботнет DoS атаки, която не просто да помогне за усъвършенстването и тестването на защитата на една компютърна система или мрежа, но и да даде възможност за извършването на стрес тестове в реално време и симулации, максимално доближаващи се до реалните условия, при които системата ще бъде принудена да функционира, в критични за нейната сигурност моменти.

Системата за симулация на IRC ботнет DoS атаки предоставя гъвкав инструментариум, с помощта на който могат да бъдат симулирани различен тип атаки с разнообразни параметри, както в реални, така и в лабораторни условия.

Цел на бъдеща разработка е разширение на функционалността на системата с още популярни видове атаки, за да се получи ефективна система за тестване на различни техники за засичане и техники за превенция, както и да се тества и сигурността на дадена мрежа.

Литература

- [1]. Elisan C. C., Malware, Rootkits & Botnets A beginner's guide, McGraw-Hill Education – Europe, ISBN 9780071792066, 2012, стр. 55-82
- [2]. Bégin, F., BYOB: Build Your Own Botnet and learn how to mitigate the threat posed by botnets, SANS Institute, The SANS Institute, 2011, стр. 1-4, 32-35
- [3]. Shankdhar, P., DoS Attacks and free DoS attacking tools, INFOSEC Institute, 2013.
- [4]. Belaton, B., A. H. R. Awadi, Multi-phase IRC Botnet and Botnet Behavior Detection Model. International Journal of Computer Applications (0975 – 8887), March 2013, Volume 66– No.15, стр. 2,10
- [5]. Waldecker, B., A Review on IRC Botnet Detection and Defence, Austria, St. Poelten University of Applied Sciences, 2013, стр. 1-9
- [6]. Алексиева, Ю., Критерии за генератори на ботнет DoS атаки, Годишник на ТУ-Варна, ЧНС, април 2016
- [7]. Bonesi, the DDoS Botnet Simulator, <https://code.google.com/p/bonesi/>

За контакти:

инж. Юлия А. Алексиева
кафедра „Компютърни науки и технологии“
Технически университет Варна
E-mail: iulia93@abv.bg

КАЧЕСТВО НА УСЛУГИТЕ В ОБЛАЧНИ ИЗЧИСЛЕНИЯ

Сияна Х. Вълчанова

Резюме: Облачните изчисления са доказано по-изгодният икономически модел за достъп до ИТ продукти, решения и услуги онлайн през Интернет в сравнение с традиционните локални решения. Облачните изчисления се характеризират като ново, четвърто поколение технологии за виртуализация на инфраструктурата. Качеството на услугите (Quality of Service – QoS) играе съществена роля за ефективното резервиране и предоставяне на ресурси. Актуален е въпросът за разработване на подходи и средства за повишаване на QoS в облачните среди, предоставящи изчислителни услуги. В настоящия доклад са представени някои аспекти на един от най-важните компоненти на QoS в облачните изчисления – наличността на услугите, както и изисквания към нейното подобряване.

Ключови думи: Облачни изчисления, Качество на услугите, Наличност

Quality of Services in Cloud Computing

Siana H. Valchanova

Abstract: Cloud computing has already proven itself to be the more profitable economic model of access to IT products, solutions and services online via the Internet for enterprises of different sizes and it also promises to become a serious competitive advantage. Cloud computing is characterized as a new, fourth-generation virtualization technology infrastructure. Because the Quality of Service (QoS) have an essential role in effective allocation and providing of resources, the important issue is the development of approaches and tools to increase the QoS to cloud environments. This paper presents some aspects of one of the most important components of the QoS for cloud computing - the availability of services and requirements for its improvement.

Keywords: Cloud Computing, Quality-of-Service, Availability

1. Въведение

Качеството на услугите (Quality of Service – QoS) играе съществена роля за ефективното резервиране и предоставяне на ресурси в разпределените системи за услуги, известни като „Изчисления в мрежа” (Grid Computing). Развитието на тези разпределени системи еволюира във развитието на нова парадигма – „Облачни изчисления” (Cloud Computing) [1, 3]. Този нов тип изчисления налага промяна по отношение на приложните и системните изисквания.

С популяризирането на облачните изчисления сред бизнеса и академичните общности рязко нараства както броят на предоставящите услуги доставчици, така и на интересующите се от тези услуги потребители. Доставчиците на услуги трябва да отговорят конкурентно на тази ситуация, предоставяйки услуги с високо качество, същевременно постигайки ниски производствени разходи. От друга страна, основната мотивация на потребителите да използват подобни услуги, отколкото техни собствени инфраструктури, е да постигнат високо качество на услугите на сравнително ниска цена. В този аспект постигането на баланс между QoS и цената е от взаимен интерес както за доставчиците, така и за потребителите.

От гледна точка на доставчиците постигането на този баланс се явява голямо предизвикателство поради все по-голямото разрастване на дейта центровете както по обем, така и по местоположение. Например, Google има 12 облачни дейта центрове на четири

континента. Управлението и поддържането на подобни дейта центрове изисква значителни разходи [8].

Разглеждането на баланса между QoS и цената е също важно и от гледна точка на потребителите. От една страна, за да е изгодно използването на различни услуги с различно качество и цена, потребителите трябва едновременно да използват услуги от различни доставчици [7], известни като *multi-cloud*. Това обаче води до нови предизвикателства, като подбор на правилните услуги, които ще удовлетворят потребителите в термините на QoS и цена [5]. От друга страна, независимо от очакванията на потребителите за определена производителност, доставчиците на облачни услуги не предлагат гаранции за подобна производителност. Поради динамичната натура на облачните среди, непредсказуемите претоварвания и откази, както и трудното оценяване на производителността за изискваните ресурси, актуален е въпросът за разработване на подходи и средства за повишаване на QoS в облачните среди, предоставящи изчислителни услуги [2].

В настоящия доклад са представени някои аспекти на един от най-важните компоненти на QoS в облачните изчисления – наличността на услугите, както и изисквания към нейното подобряване.

2. Cloud Computing

Облачни изчисления (Cloud Computing) е технология на разпределена обработка на данните, където компютърни ресурси и мощности се предоставят на потребителите като услуги (облачни услуги) при необходимост. Информация и потребителски приложения се съхраняват постоянно на мощни сървъри в Web (т.н. „облаци”) и са достъпни през интранет или Интернет. Така потребителите могат да използват различни приложения онлайн чрез широка гама крайни устройства (настолни компютри, лаптопи, смартфони, таблети), заплащайки само за действителното потребление. Потребителите получават достъп до мощни изчислителни и дискови ресурси, без да се интересуват за физическото разположение на тези ресурси, тяхната поддръжка и управление (фигура 1). Решенията при Cloud Computing се характеризират като *ново, четвърто поколение технологии за виртуализация на инфраструктурата*. Те се отличават с гъвкавост, мащабируемост, икономичност и производителност [1].



Фиг. 1. Достъпност и мобилност до облачни услуги от всяко място

2.1. Характеристики на Cloud Computing

Основните предимства на Cloud Computing се изразяват в следните аспекти:

Гъвкавост на инфраструктурата, която по-лесно може да се адаптира към бизнес потребностите на компанията. Доставчиците на облачни услуги са големи компании, които разполагат с възможности лесно да разширяват потребителската виртуална среда за сметка

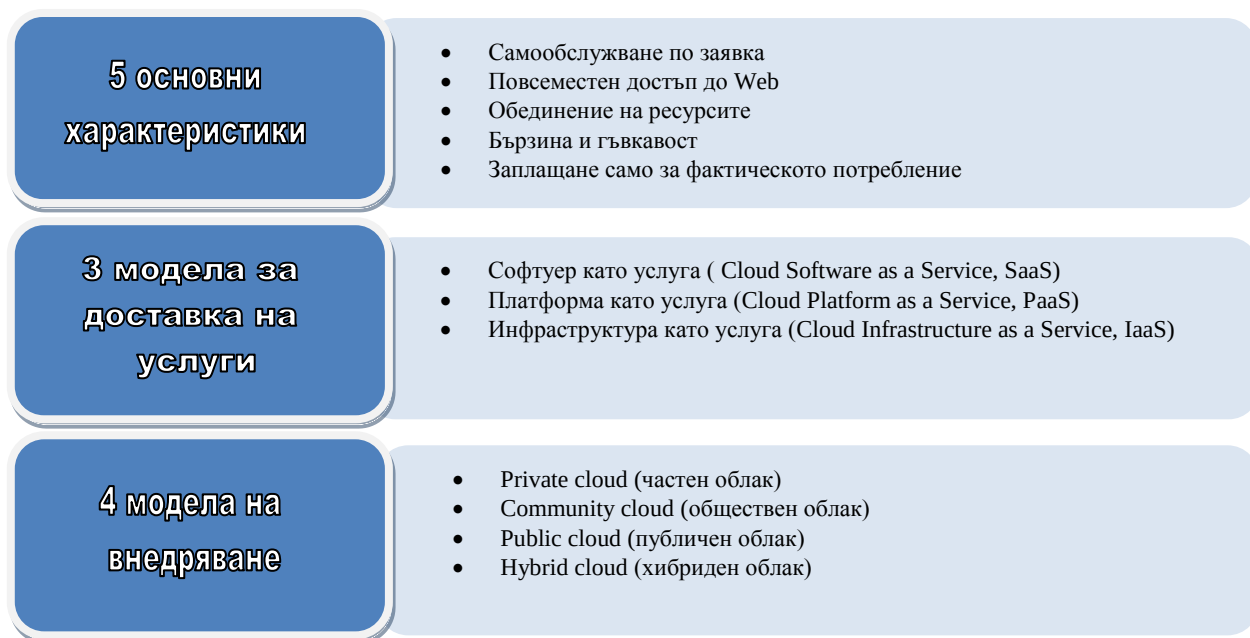
на собствената си виртуална инфраструктура, като дадат достъп до много производителни хардуерни и софтуерни платформи и много по-висока скорост за работа с тях. Освен това, Cloud Computing предлага нов *гъвкав модел на използване и заплащане на услугите*. Клиентите могат да използват точно толкова, колкото желаят, като имат възможност да повишават или понижават своите бизнес изисквания. Така добре познатият модел на фиксирани разходи за използването на дадена услуга се трансформира в *модел на променливи разходи*, което води до гъвкавост и защита при ситуации на съществена промяна на приходите или нуждите от ИТ услуги.

Икономия на средства и мащабируемост. Една от най-важните цели на Cloud Computing е предоставянето на по-евтини ресурси в сравнение с тези, които се поддържат и обслужват локално.

Мобилен достъп. Cloud Computing платформата е достъпна от всяка точка на света, тъй като е базирана върху географски разпределени центрове за обработка на данни, ресурси и платформи. Фирмени данни и документи са на разположение по всяко време и от всяко място при наличие на Интернет връзка и най-често Web браузър. Мениджърите и служителите могат да използват потребителски приложения и документи независимо от това дали са в офиса или извън него, чрез разнообразни средства и технологии за достъп до Интернет.

Висока производителност. Услугите на Cloud Computing се характеризират с мащабируемост, достъпност, производителност и безопасност, каквито не могат да осигурят отделните компании, разчитайки на собствените си ИТ ресурси. По този начин предприятията може да съсредоточат своите усилия върху най-съществените бизнес процеси, иновациите и стратегиите. Веднага трябва да се добави, че достъпността, надеждността и производителността на облачните услуги са силно зависими от инфраструктурата на доставчика и пропускателната способност на каналите за връзка, по които се движи трафикът до клиентите.

Моделът на Cloud Computing (фигура 2) може да се представи като съвкупност от пет основни характеристики, три модела на услуги, както и четири модели за внедряване [6].



Фиг. 2 Модел на Cloud Computing

2.2. Модели на услуги

Софтуер като услуга (SaaS) – определя модел на разпределяне на софтуер, при който приложенията се поддържат от доставчик и са налични през Интернет чрез Web браузър. Доставчиците поддържат инфраструктурата и платформите, върху които се изпълняват приложенията. SaaS предоставя потенциална възможност за бизнеса да използва софтуер на ниска цена при необходимост, отколкото закупуване на лицензи за всеки негов компютър [4].

Платформа като услуга (PaaS) – при този модел се предоставя достъп до операционна система и асоциирани услуги през Интернет без необходимост от изтегляне и инсталиране. Потребителят на услугата има възможност за изграждане на собствени приложения, работещи върху наетата среда, използвайки по принцип неограничени изчислителни ресурси.

Инфраструктура като услуга (IaaS) – предлага възможност за наемане на изчислителни мощности и съхранение на данни като стандартна услуга. На клиента се предоставя възможност за използване на сървъри, операционни системи, системи за съхранение на данни, мрежово оборудване и др.

2.3. Модели на предоставяне на ресурси

Днес дистанционното използване на облакови ресурси намира все по-голямо приложение както за рутинни операции на малки и средни предприятия, така и за големи изследователски проекти [3]. Съществуват следните видове облакови ресурси:

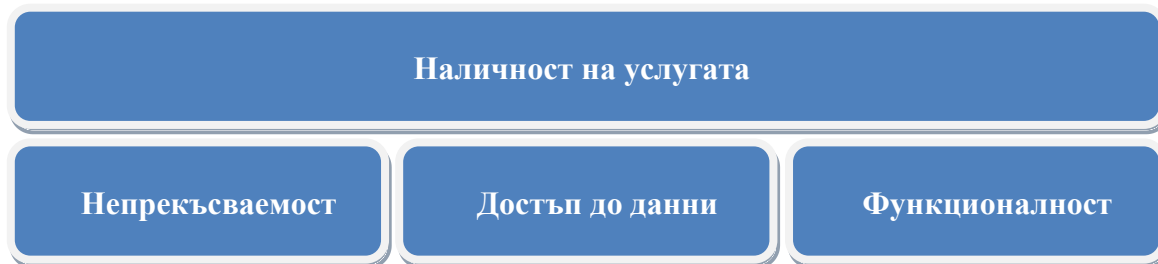
- Публичен облак (Public cloud) – достъпът до ресурсите се осъществява от обикновен потребител при условие на достъп до Интернет. Публичният облак осигурява гъвкав и икономичен начин за ползване на ИТ решения.
- Частен облак (Private cloud) – ресурсите са достъпни само за ограничен брой лица. Частният облак има по-високо ниво на сигурност в сравнение с публичния облак, тъй като е защитен от защитна стена и може да бъде достъпван единствено през вътрешна защитена мрежа. Процесите, услугите и информацията се управляват вътре в самата организация, което рефлектира в по-високата цена на този тип облаци.
- Обществен облак (Community cloud) – подобен е на публичния облак, с изключение на това, че ресурсите са достъпни на ограничен кръг организации, имащи еднакви потребности от гледна точка на информационните ресурси.
- Хибриден облак (Hybrid cloud) – облак, състоящ се от два и повече облака от различни видове, например публичен и частен.

3. Наличност на услугите в Cloud Computing

Качеството на услугите (QoS) дефинира нивото на производителността, надеждността и наличността, предоставяни от приложенията, платформите и инфраструктурите, където те се поддържат. Качествата на услугите са фундаментални за потребителите на облачните услуги, които очакват офериранияте от доставчиците характеристики, както и за самите доставчици, които се опитват да намерят баланс между качеството и себестойността на предоставяните от тях услуги. Намирането на оптимален баланс не е лесна задача, тъй като това изисква спазването на т.н. Service Level Agreement (SLA) договор между доставчиците и крайните потребители [5].

Една от най-важните характеристики на QoS е наличността (Availability) на услугите. Тя указва активността (Uptime) на системата, мрежата, хардуера или софтуера, които

съвместно предоставят определена услуга. Наличността може да се разглежда като съвкупност от няколко компонента (фигура 3).



Фиг. 3. Компоненти на наличността

Непрекъснатостта осигурява наличието на услугата за определен период от време без никакви прекъсвания. Същевременно, при наличие на инцидент, непрекъснатостта позволява рестартиране на услугата и предоставяне на повторен достъп до данните и функционалността на услугата в приемлив времеви диапазон. Това налага доставчикът да поддържа солидни средства за обработка на инциденти, за да гарантира спазването на SLA. Процесът на управление на инциденти касае също и потребителите, особено в случай на комерсиални услуги. Ако има загуба на наличност вследствие на хардуерен или софтуерен проблем или разпространените разпределени DoS атаки, крайният потребител трябва да бъде информиран не само за настъпилния проблем, но и за очакваното време на изчакване (downtime). След обработка на инцидента и възстановяването на услугата, той трябва да бъде известен, че може да рестартира услугата. Необходимо е да вземе предвид, че процесът на възстановяване от инцидент може да изисква определено време, така че пълното възстановяване на услугата да не може да се извърши на един път.

Функционалността описва какво крайният потребител може да получи от избраната услуга за определен период от време. Предоставянето на минимална функционалност е критично от гледна точка на бизнеса. Ако важна функционалност отсъства по време на фундаментална промяна, извършена от доставчика, това може да навреди на бизнеса на потребителя. В резултат е нарушаване на SLA. В подобни случаи е необходимо потребителят да има достатъчно време, през което функционалността ще е налична и тествана преди сключване на SLA.

Достъпът до данни трябва да осигурява надежден достъп както до данните, въвеждани от потребител, така и до тези, генерирани от услугата в резултат на самите потребителски данни. Осигуряването на адекватен достъп има голямо значение особено за данни, касаещи лична, финансова или здравна информация. Интересен проблем при достъпа до данни са случаите, когато потребителят формира своите решения на базата на получените от услугата данни. В подобни случаи доставчикът трябва да осигури приемливо време за отговор. В други случаи крайният потребител е възможно да не може да завърши транзакцията навреме, което би довело отново до загуби. Ето защо е наложително доставчикът да може да осигурява възстановяване на данни от подобен род.

За осигуряване на непрекъснатостта на услугите в Cloud Computing е необходимо наличието на средства за откриване на изключения, некоректно функциониране и понижаване качеството на услугите с цел бърза реакция преди настъпване на отказ на услуга. Това ще позволи на доставчиците на услуги да поддържат качеството на предоставяните от тях услуги. Актуален е въпросът за разработване на модели за анализ на качеството на услугите, както и средства за тяхната динамична количествена и качествена оценка.

4. Заключение

С повишаване на нуждите на потребителите, бизнеса и операторите от услуги и ресурси като еволюционен етап от развитието на компютърните технологии и приложения се появяват и системите за Cloud Computin. Тези системи предоставят един нов подход, който води до нова стратегия, включваща в себе си пълно преосмисляне на ролята на информационните технологии в организациите. Качеството на услугите е фундаментално за потребителите на облачните услуги, които очакват офериранияте от доставчиците характеристики, както и за самите доставчици, които се опитват да намерят баланс между качеството и себестойността на предоставяните от тях услуги.

В настоящия доклад са представени някои аспекти на качеството на услугите в облачните изчисления, като е акцентирано върху една от най-важните им характеристики – наличността на услугите. Цел на бъдеща работа е разработване на модели за анализ на качеството на услугите, както и средства за тяхната динамична количествена и качествена оценка.

Литература

- [1]. Armbrust M, A. Fox, R. Griffith, at all. A View of Cloud Computing. Commun. of ACM 2010, pp.50–58.
- [2]. Danilo A., G. Casale. Quality-of-service in cloud computing: modeling techniques and their applications. Journal of Internet Services and Applications, 2014, pp.5-11.
- [3]. Erl T., Cloud Computing: Concepts, Technology & Architecture. Prentice Hall, 2013, p.528.
- [4]. Kavis M. Architecting the Cloud: Design Decisions for Cloud Computing Service Models (SaaS, PaaS, and IaaS).Wiley, 2014, pp.224.
- [5]. LinlinW., K. Saurabh. SLA-Based Resource Allocation for Software as a Service provider in Cloud Computing Environments. In Int. Symp. On Cluster, Cloud and Grid Computing, 2011, pp.195-204.
- [6]. NIST Cloud Computing Reference Architecture. <http://www.nist.gov>.
- [7]. Vahid D. QoS-Aware and Semantic-Based Service Coordination for Multi-Cloud Environment. PhD Thesis. University of Melbourne, 2013.
- [8]. Quinghua Lu, X. Xiwei. Incorporating Uncertainty into in-Cloud Application Deployment Decisions for Availability. In Int. Conf. In Cloud Computing, 2013, pp.454-461.

За контакти:

инж. Сияна Х. Вълчанова, докторант
катедра „Компютърни науки и технологии”
Технически университет - Варна
E-mail: siqna_valchanova@abv.bg

АНАЛИЗ И ИЗСЛЕДВАНЕ НА АЛГОРИТМИ ЗА НАМИРАНЕ НА ПЪТ В СРЕДА

Никола В. Василев, Милена Н. Карова

Резюме: Докладът има за цел да дискутира и сравни три различни проблема за ориентиране на робот в статична среда. Роботът се приема като една точка в двумерен масив. За постигане на целта се използват модифицирани генетични алгоритми. Във всички случаи генерацията на път се получава на случаен принцип. Изпълнението на алгоритмите е напълно динамично.

Ключови думи: Алгоритми, Програмиране, Намиране на път, Генетични алгоритми.

Analysis and research of pathfinding algorithms

Nikola V. Vasilev, Milena N. Karova

Abstract: The idea of the article is to discuss and compare three different problem of path planning for robot in static area. Considering the case of constrained environments where the robot is represented as one point and area like array. For that, we used an approach based on models of modified genetic algorithms. A population of paths is obtained firstly using a random distribution strategy. The performance of the proposed algorithms is fully dynamic with increasing complexity.

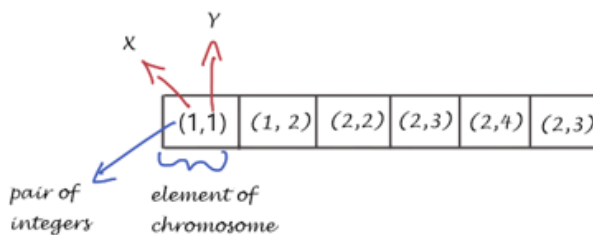
Keywords: Algorithms, Programming, Pathfinding, Genetic algorithms

1. Въведение

Целта на алгоритмите е да се намери маршрут, който да съдържа само движения, които са позволени. Трябва да се отбележи че роботът вижда само на разстояние една стъпка пред него в четирите посоки. Алгоритмите работят със статична среда. Входната и изходната точка се задава от потребителя. Използван е обектно-ориентиран модел за създаването на алгоритмите.

2. Използвани структури

За реализация на алгоритмите са използвани главно динамични масиви. Средата се представя като двумерен масив, а пътят (хромозома), като масив от точки, които са чифт от целочислени стойности (фигура 1).



Фиг. 1. Структура на хромозома

3. Генерация на средата

Темата на доклада не е насочена да дискутира алгоритми за генерация на среда (лабиринт). Използва се стандартен DFS алгоритъм за създаване на „перфектен лабиринт“

(който има само един възможен път от входната до изходната точка) и проста функция за изчистване на препятствия за анализ на резултатите.

4. Генетично разпределение

Методът, по който се генерира всяка една хромозома, е базиран на случайно генериране на цели числа в диапазона от 1 до 4, които се разбират като четири посоки (напред, назад, нагоре, надолу). Параметър на метода е размерът на популацията. В началото се инициализират две точки – текуща, която е началната, и временна, както и индекс за текущата хромозома. Разпределението се извършва, докато текущата точка не стигне целевата. При всяко итериране се пази точката преди преместване, която е равна на текущата, генерира се посока, и ако е възможно движение, се записва в хромозомата, иначе се възстановява с координатите на предходната. Този цикъл продължава, докато координатите на текущата точка не са равни на целевата.

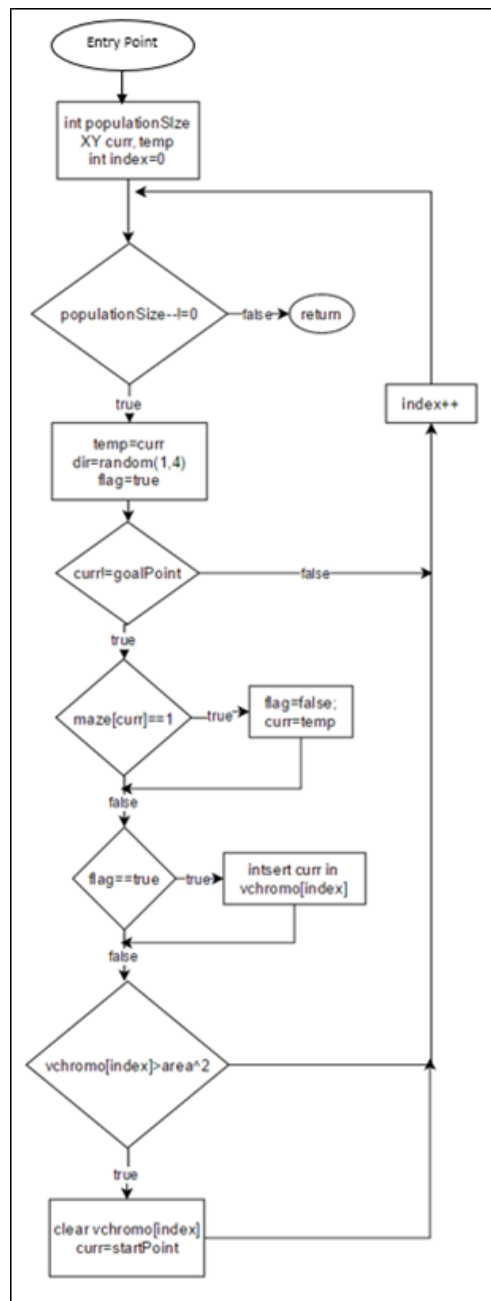
В алгоритъма има и включена проверка - след края на всяка добавена точка в хромозомата се проверява дали дължината на хромозомата е по-малка от квадрата на площта. Ако тя е по-голяма, се изчиства. Основни разлики тук са, че при генерацията се извършва и валидацията, т.е. проверката за принадлежаща стена в хромозомата се извършва при самото генериране. Генерацията на хромозомите е разпаралелена, понеже данните за генерация на всяка хромозома са само за самата нея и няма опасност от кохерентност на данните. Алгоритъмът е представен на фигура 2.

5. Фитнес метод

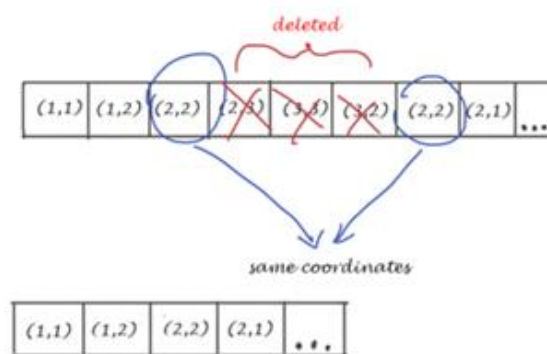
Целта на метода е да се намерят най-кратките N пътища от всичките хромозоми. В началото се инициализират два вектора, които пазят размерите и индексите на всяка хромозома. Векторът с размерите се сортира и размерът на вектора се сравнява с размера на всяка хромозома. Ако са идентични, се добавя индексът във вектора за индекси, разменят се желаните хромозоми, а всички останали се изчистват. Препоръчително е отделянето да се извършва след генерацията, а не след модификацията, защото времето за изпълнение се удължава до десет пъти.

6. Метод за коригиране

Целта на този метод е да се премахне въртенето на робота. Например, ако координатите са $(1,1) \rightarrow (2, 1) \rightarrow (3, 1) \rightarrow (2,1) \rightarrow (1,1)$, да останат само $(1, 1)$. Подобен пример е даден на фигура 3. Решението е просто, намират се дублираните координати и се премахват всички елементи между тях. Алгоритъмът е паралелен, т.е. всяка една хромозома се поправя на отделна нишка. Броят на нишките се определя хардуерно и използва всички налични процесорни нишки.



Фиг. 2. Блок схема на генетичното разпределение



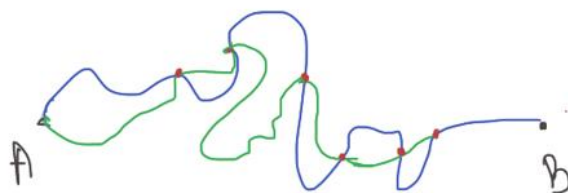
Фиг. 3. Пример на коригиращия метод

7. Намиране на път с връщане назад (Backtracking)

Методът наподобява генетичния, но разликата е, че имаме вектор от checkpoints. Checkpoint е всяка точка (двойка от координати x, y), която има повече от един път, т.е. не е заградена от препятствия. Алгоритъмът е добър за случая „перфектен лабиринт“. Съдържа в себе си и проверка за провалените опити за намиране на пътя, и ако са повече от пет, се изчиства векторът и се започва отначало.

8. Crossover

Този метод има за цел да намери най-краткия път измежду 2 хромозоми. Пример - на фигури 4 и 5.

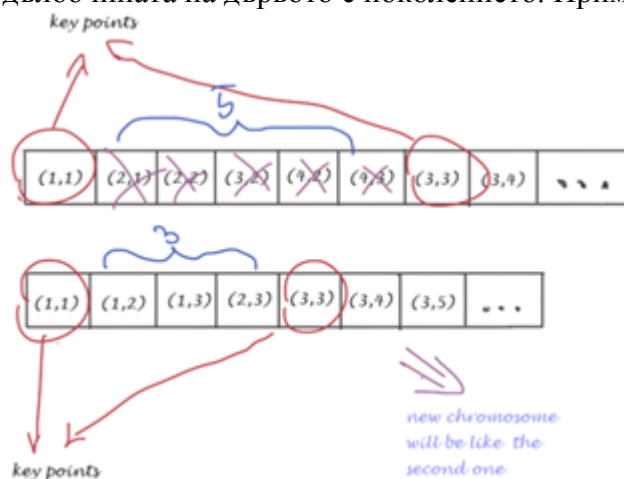


Фиг. 4. Кръстосване на два пътя



Фиг. 5. Резултат от кръстосването

Основната идея тук е сечение на два вектора и вземане на по-малката дължина от двете сечения. Алгоритъмът намира добро приложение, ако се използва при построяване на бинарно дърво, където дълбочината на дървото е поколението. Пример - на фигура 6.



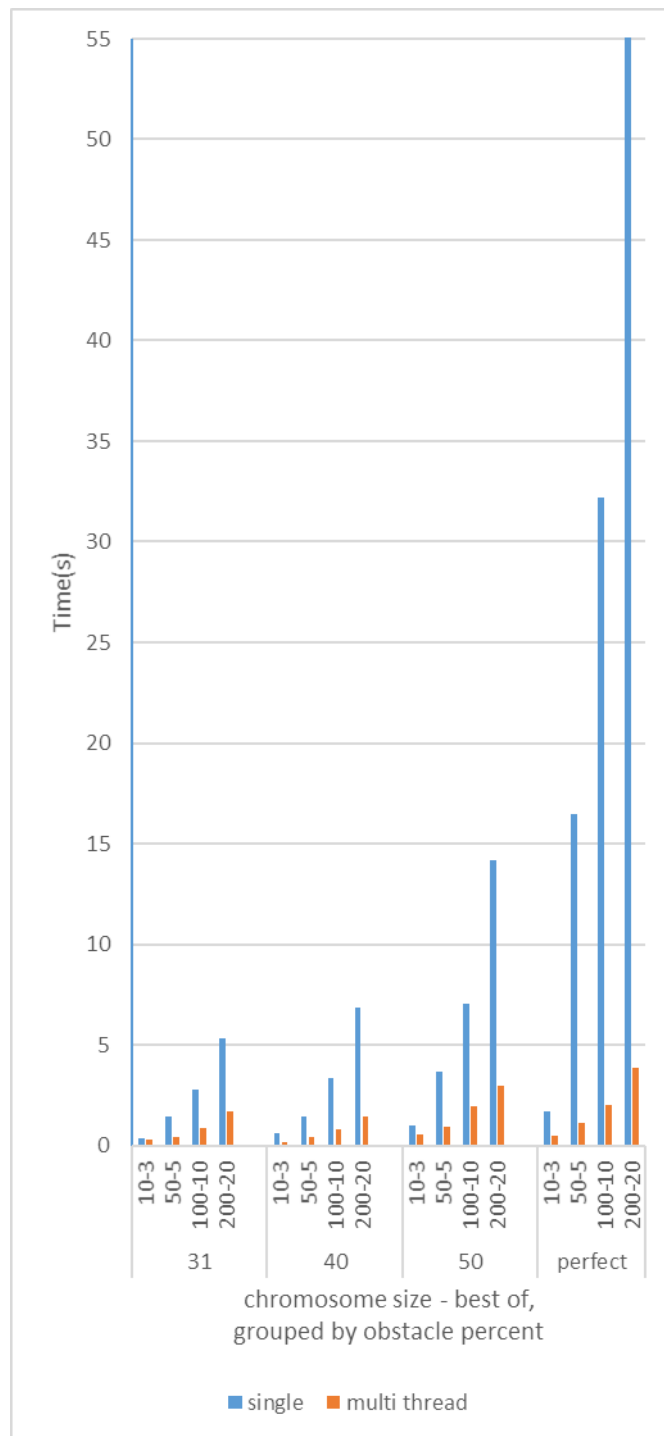
Фиг. 6. Кръстосване на две хромозоми

9. Тестове и анализ

size	obstacle%	backtracking		genetic	
		time	size	time	size
11x11	50	0.004	2	0.003	100
11x11	50	0.003	2	0.005	100
11x11	55	0.003	2	0.004	100
11x11	55	0.003	2	0.004	100
11x11p	60	0.005	2	0.004	100
11x11p	60	0.002	2	0.004	100
21x11	40	0.012	2	0.034	100
21x11	40	0.017	2	0.028	100
21x11	40	0.012	2	0.032	100
21x11	45	0.013	2	0.033	100
21x11	45	0.009	2	0.029	100
21x11	45	0.011	2	0.035	100
21x11	50	0.012	2	0.039	100
21x11	50	0.01	2	0.007	100
21x11	50	0.004	2	0.011	100
21x11	55	0.002	2	0.007	100
21x11p	55	0.002	2	0.008	100
21x11p	56	0.004	2	0.006	100
21x21	40	0.006	2	0.022	100
21x21	40	0.034	2	0.022	100
21x21	40	0.127	2	0.015	100
21x21	45	0.002	2	0.025	100
21x21	45	0.003	2	0.025	100
21x21	45	0.002	2	0.023	100
21x21	50	0.003	2	0.024	100
21x21	50	0.004	2	0.021	100
21x21	50	0.002	2	0.031	100

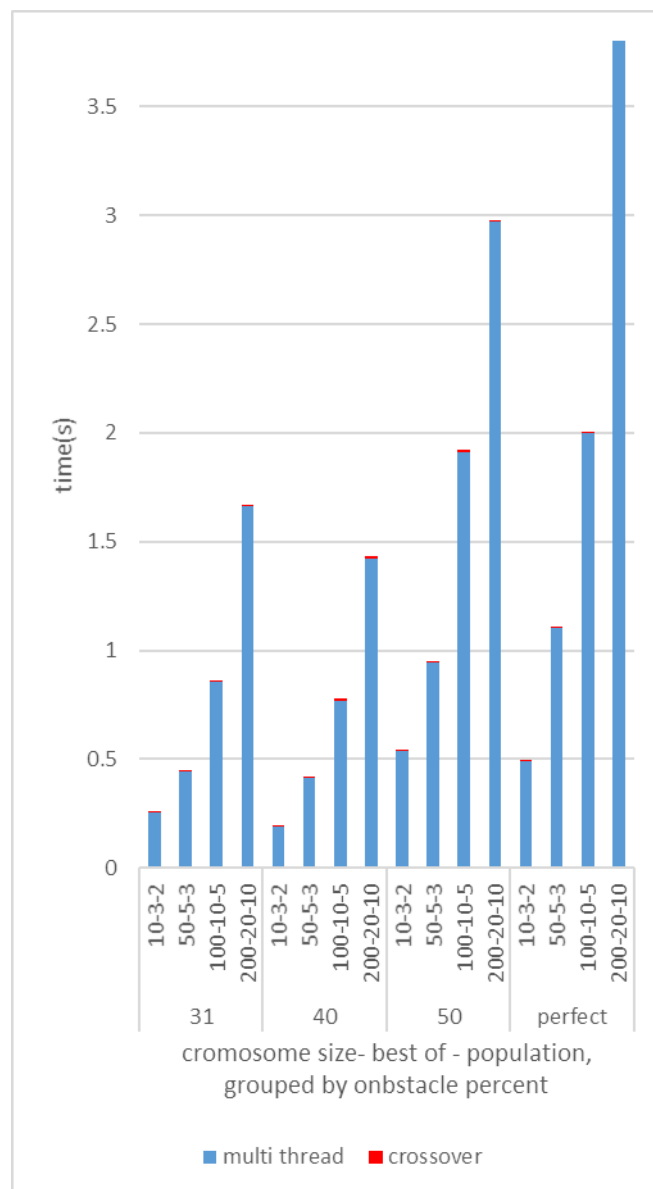
Фиг. 7. Сравнение между генетично разпределение и с backtracking

От направите тестове, резултатите от които са показани на фигури 7, 8 и 9, се вижда, че при нарастване на заетостта сложността на алгоритъма се увеличава експоненциално при една нишка и линейно при паралелен алгоритъм. При backtrack алгоритъма обаче, времето намалява логаритмично.



Фиг. 8. Сравнение между генетично разпределение върху една/осем нишки на среда с размер 35x25

От фигура 9 се вижда, че времето за кръстосване на решенията не оказва голямо влияние върху цялостния резултат. При поколение по-голямо от 5, се получава кратък път, който в сравнение с масовия алгоритъм за намиране на най-кратък път - A*, е 5 стъпки по-малък, но времето е 5 пъти по-голямо.



Фиг. 9. Сравнение между генетично разпределение и кръстосване между най добрите решения , върху осем нишки на среда с размер 35x25

10. Заключение

Генетичните алгоритми са бързи, но не винаги дават добър резултат. Бектрекингът работи най-добре в случая „перфектен лабиринт“, а генетичният - във всякаква друга среда. Колкото е по голяма популацията, толкова по-добро и качествено е решението. За най-добро решение се използва и кръстосване на решенията. В случая колкото повече хромозоми съществуват, толкова повече поколения ще се образуват.

Литература

- [1]. Преслав Наков, Панайот Добриков. Програмиране==++Алгоритми.
- [2]. John Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, Michigan. 1975. ISBN 0-262-58111-6.

- [3]. Larry J. Eshelman, The CHC Adaptive Search Algorithm: How to Have Safe Search When Engaging in Nontraditional Genetic Recombination, in Gregory J. E. Rawlins editor, Proceedings of the First Workshop on Foundations of Genetic Algorithms. pages 265-283.
- [4]. Morgan Kaufmann, 1991. ISBN 1-55860-170-8.
- [5]. Tomasz D. Gwiazda, Genetic Algorithms Reference Vol.1 Crossover for single-objective numerical optimization problems, Tomasz Gwiazda, Lomianki, 2006.

За контакти:

Никола Веселинов Василев
катедра „КНТ”
ТУ Варна
E-mail: tutzrzt@gmail.com

доц. д-р инж. Милена Николова Карова
катедра „КНТ”
ТУ Варна
E-mail: milena.karova@gmail.com



СЪВРЕМЕННИ СОФТУЕРНИ ПАКЕТИ ЗА ОБРАБОТКА НА ДАННИ В ПСИХОЛОГИЯТА. SPSS И R - СИЛНИ СТРАНИ И ОГРАНИЧЕНИЯ

Славена А. Лазарова, Вяра Н. Колева - Ефремова

Резюме: Този доклад съдържа преглед на съвременните пакети за обработка на статистически данни, които се използват в социалните науки. Прегледът покрива някои от най-широко използваните софтуерни продукти, сравнение на тяхната употреба според сферата на дейност, покритието на статистическите анализи, предимствата и недостатъците за потребителите. Представяме обзор на значимостта на статистиката в социалните науки и трудностите пред студентите по психология. Приемаме, че има две алтернативи пред тях по отношение на софтуера, който ще използват – SPSS и R, и предлагаме сравнителен анализ на тези два продукта, с цел да улесним процеса на избор на най-подходящия за тях софтуер.

Ключови думи: SPSS, R, изследователски методи, статистика, софтуер, психология

Contemporary Software Packages for Statistical Analytics in Psychology. SPSS and R – Prospects and Constraints

Slavena A. Lazarova, Vyara N. Koleva - Efremova

Abstract: This report contains a review on the contemporary statistics software packages, which are used in social science research. The review covers some of the most widely spread software products, a comparison of their usage in different fields, their statistical analytics coverage, advantages and disadvantages of the user experience. We have presented an overview of the importance of the statistics in social science studies and the difficulties experienced by psychology students. We assume that there are two alternatives to them, in regards to the software they will use – SPSS and R, and we provide a comparison of these two products in order to ease the users in the process of finding the most suitable one for them.

Keywords: SPSS, R, research methods, statistics, software, Psychology

1. Въведение

Традиционно голяма част от образованието по психология се заема от изследователските методи и статистиката, защото те са важна основа за тези от студентите, които избират да разработят научни трудове, дипломни работи, дисертации или дори да продължат кариерата си в областта на академичните изследвания.

Неразделна част от изследователските методи се явява и софтуерът за обработка на статистическите данни. Изборът на подходящ продукт е от съществена важност и съвсем не е лесно решение. Но как да го изберем? Често статистиката и математиката сами по себе си са достатъчно плашещи за изучаващите и завършилите психология и мотивацията за по-сериозно проучване на възможните алтернативи и изучаването на софтуера и всички негови възможности изобщо липсва. Много от потребителите се спират на даден продукт без предварителна подготовка относно това, с което им предстои да се сблъскат. Често основен фактор при избора са изискванията на техните преподаватели и научни ръководители.

Традиционно в социалните науки се използва SPSS, но все повече форуми, дискусии, публикации и дори книги акцентират на неговите ограничения и на възможните алтернативи и в частност на софтуера R.

Целта на този доклад е да се направи по-задълбочено проучване на двата продукта (SPSS и R), за да се изведат техните силни и слаби страни и да се улесни изборът, в зависимост от потребителските нужди.

2. Съвременни софтуерни пакети за обработка на статистически данни в психологията

Wikipedia дава дълъг списък от близо 60 софтуерни пакета за статистически анализ на данни [1]. След широко проучване на форуми, уебстраници [1, 2, 3, 4, 5], статии и трудове [6, 7, 8] по въпроса кой е най-добрият и най-често употребяван софтуерен пакет, се стигна до извода, че хората предимно се насочват към това, което използват техни колеги или приятели или този продукт, с който са се запознали в университета. Това важи в пълна сила и за употребата им в психологическите изследвания. Считаме, че от определящо значение за масовата употреба на даден софтуерен пакет са предпочитанията на академичната общност и в частност университетските преподаватели по предмети, свързани с изследователските методи.

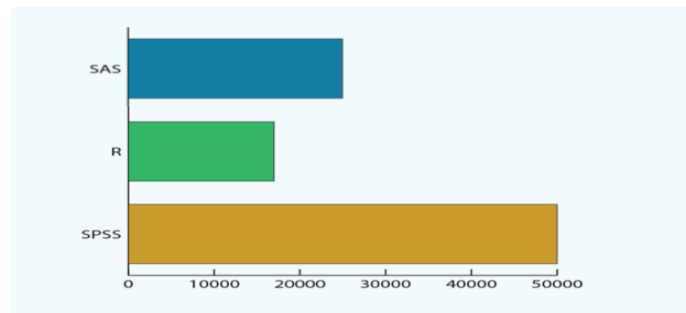
Тъй като образованието по психология е с научно-практическа насоченост, изучаването на изследователски методи и статистиката са съществена част от него. Въпреки това, по много причини, темите в изследователски методи са предизвикателни за преподаване. Студентите имат разнообразна математическа, статистическа и изчислителна подготовка; студентите често нямат мотивация и не отдават голямо значение на статистиката [7].

Както при много други университети, изследователски методи в психологията в Deakin University (Мелбърн, Австралия), където преподава Англим, до голяма степен се преподава с помощта на софтуер, наречен SPSS. Този софтуер обикновено се представя на студентите като програма с меню, която се използва за анализ на данни, даваща възможност за стандартна манипулация на данни, анализи и графична визуализация. Докато SPSS е относително лесен за употреба при стандартни анализи, що се отнася до преподаването му той има няколко недостатъка. По-конкретно, той е много скъп и съответно трудно достъпен продукт. По този начин не може да се очаква студентите да имат достъп до него, както от дома си, за да се упражняват и използват в проектите си, така и в бъдещите си работни места. В допълнение, докато SPSS е лесен продукт за извършване на стандартни анализи, той не е много гъвкав и това го прави труден за използване при по-нови видове анализи [7]. Поради тези причини, много преподаватели търсят алтернативен статистически софтуер за преподаване на изследователски методи [7, 6].

И така, макар и приоритетно в социалните науки да се използва софтуерният пакет SPSS (Software Package for Social Science), той далеч не е единствената възможност за обработка на данни в психологическите изследвания - използват се още SAS, Stata, R, Malab, Excel.

Считаме, че е най-подходящо да се направи добър анализ на силните и слабите страни на софтуера за обработка на статистически данни SPSS, чрез сравнението му с поне още един такъв продукт. Но кой от до тук споменатите да изберем?

На фигура 1 са представени броевете научни трудове, използващи един от трите най-често спрягани софтуерни пакети от търсене в Google Scholar. Тук ясно се вижда приоритетното използване на SPSS за академични изследвания, но делът на R и SAS също е значителен.



Фиг. 1. Броевете научни трудове, използващи SPSS

Брендън Оконър (Brendon O'Connor) - асистент в Техническият университет в Масачузетс, преподаващ „Компютъризиране в социалните науки“, извежда кратка сравнителна таблица между петте най-използвани софтуерни пакета, от които лесно могат да се видят основните различия между тези три продукта (виж Таблица 1. Сравнителна характеристика на R, SAS, SPSS).

Таблица 1. Сравнителна характеристика на R, SAS, SPSS [6]

Име	Предимства	Недостатъци	Свободен достъп?	Основни потребители
R	Поддържа библиотеки; визуализация	Стръмна крива на обучение	Да	Финанси; Статистика
SAS	Голяма база данни	Скъп; остарял програмен език	Не	Бизнес; Правителство
SPSS	Лесни статистически анализи	Много скъп	Не	Наука

От предоставената таблица става ясно защо, въпреки че SPSS е специализиран за социалните науки, софтуерът R все повече е спряган като негов основен конкурент – той има огромното предимство да е свободно достъпен [7, 9, 6] и се очертава като жизнеспособна алтернатива на SPSS. R е безплатен, така че той е свободен за студенти и работещи. Той разполага с широк спектър от статистически функционалности. Въпреки тези предимства, R предоставя няколко предизвикателства за интегрирането му в психологията. Анализите в R обикновено се осъществяват с помощта на скриптове. Така много често не е ясно за потребителя как да стартира някои от анализите [7].

3. Образованието по статистика и ролята на софтуера

Налице е значителна литература относно образованието по статистика и ролята на статистическият софтуер в него. Тисковская и Ланкастър [10] предоставя един преглед на предизвикателствата в областта на образованието по статистика. Тяхното проучване е структурирано около преподаването и ученето, статистическата грамотност и статистиката като професия. От особено значение за статистиката на преподаване в областта на психологията, те очертават няколко проблема и предоставят съответните препратки към статистическата литература. С препратки, взети от техния доклад, тези проблеми са: невъзможност да се приложи математиката към проблемите от реалния свят (например, Garfield, 1995); проблеми с мотивацията и тревожността при студентите, свързани с обучението им по математика и статистика (например, Gal & Ginsburg, 1994); присъщи трудности при студентите за разбиране теорията на вероятностите и статистиката (например, Garfield & Ben-Zvi, 2008); необходимостта от разработване на статистическа грамотност, която се превръща в ежедневие (напр, Gal, 2002) и необходимостта от разработване на инструменти за оценка на статистическата грамотност.

Проблемите при образованието по статистика са особено засилени, когато говорим за следващите психология. Макар и приоритетна част от програмата им да е заета от изследователски методи, затрудненията при възприемането им идват по две причини. Първо, студентите по психология рядко постъпват в университета със солидни математически познания. Второ, студентите по психология преследват различна професионална реализация. Не всички имат амбиции за реализация в психологическите изследвания. Голяма част от студентите търсят практическа реализация (например в клиничната психология, организационната психология и т. н.), а друга част изобщо не се реализират по специалността си.

Тук идва въпросът дали всички изучаващи психология имат еднаква необходимост и мотивация за работа с пакетите за обработка на статистически данни и дилемата пред преподавателя да избере между продукт, който ще е лесно-разбираем за всички студенти или такъв, който ще даде повече възможности на студентите, които имат амбиции за реализация в изследванията. Например, пакетът SPSS е по-ограничен от R, но неговите ограничения могат да направят стандартните анализи по-лесни за студентите [7].

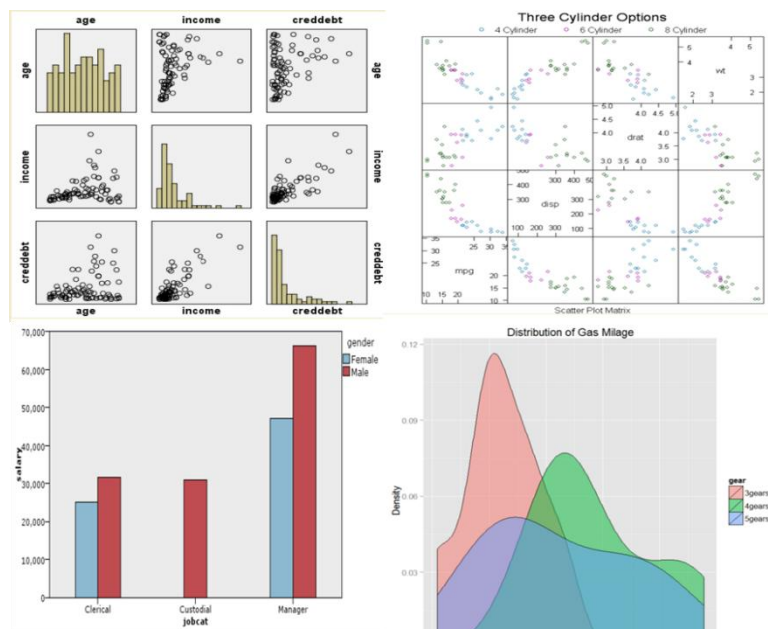
4. Сравнение на SPSS и R. Силни и слаби страни

При избора на софтуер за обработка на статистически данни трябва да се вземат под внимание следните показатели [8, 11, р. 19]: леснота на въвеждане на данни, възможност за импортиране на данни в различни формати; интерактивни възможности; динамична връзка между данните, графични и числени анализи; лесен за употреба за специфични аудитории; достъпност, преносимост; времето, което е необходимо за изучаването му; до каква степен софтуерът ще бъде използван в процеса.

По отношение на достъпност и преносимост, цената е основното предимство на R, който е напълно безплатен. За разлика от него SPSS е сравнително скъп продукт. Цената на потребителски лиценз за студенти за една година е 200 долара, цената на професионалния лиценз е 2000 долара, а цената, която плащат университетите, също не е малка [12]. Задължителното използване на SPSS в обучението създава проблеми по отношение на финансовите възможности на студентите и дори може да ги тласне към пристъпване към търсене на пиратски версии на програмата.

От гледна точка на функционалността:

- И двата продукта могат да бъдат инсталирани на Windows, Mackintosh и Linux операционни системи.
- И двата софтуера поддържат основните анализи, но R предлага много по-голям избор (виж таблица 2)
- SPSS има няколко инструмента, които R не поддържа, включително инструмент за въвеждане на данни, графичен интерфейс с менюта и система за управление на таблици и графики [7] (виж фигура 3).
- R позволява всички анализи да се случват „на едно място“ и следователно слага край на прехвърлянето на файлове от Excel към SPSS, от LISREL към SPSS, минаването през .csv файлове за трансфер на данни и др. [9].
- Веднъж заредени данните в R, те остават на разположение до края на сесията и не е нужно да се отварят нови прозорци с данни [9].
- R предлага по-добра визуализация на графики и чартове (виж фигура 2)



Фиг. 2. Визуализация на графики и чартове на SPSS (ляво) и R (дясно) [5]

По отношение на своята *гъвкавост* и SPSS и R имат възможност за извършване на гъвкави анализи. Въпреки това, R предоставя много по-лесно постепенното надграждане на стандартните анализи. Той е изключително подходящ за разработването за напреднали изследователи, които могат да се възползват от възможностите му за увеличаване на статистическата сложност [7]. Предвид тенденцията дизайните на изследванията (особено в експерименталната и когнитивната психология) да стават все по-многофакторни, специфични и изискващи "Бийсетовска статистика", тяхното изчисляване отдавна не е по силите на SPSS [9]. Освен това R е изключително гъвкав по отношение на отворения му код. Всеки може да пише библиотеки, както и да ги добавя на поддържащите R сървъри. Към октомври 2015г. пакетите са вече 7352 [13].

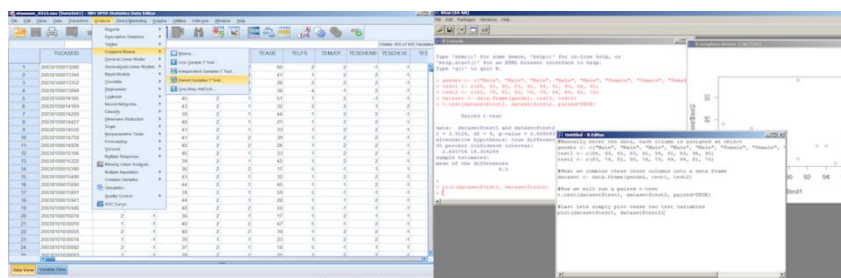
От гледна точка на *приложимост* R и SPSS са доста различни:

- R предполага задълбочени познания по статистика.
- SPSS има интерфейс, който прилича повече на стандартните Windows-базирани програми (виж фигура 3).
- R е език за програмиране с по-малко последователен мисловен модел на стандартните програми за Windows (виж фигура 3).
- R има по-стръмна крива на изучаване (*learning curve*), докато при SPSS тя е сравнително полегата (виж фигура 4). Тъй като R е с отворен код, това създава трудности на потребителите, които трябва да изучават неговия синтаксис.
- SPSS предоставя по-лесни опции от R за следното: именуване на променливите, модифициране на мета данни, импортиране на данни, интерактивни чартове, стандартни анализи и др.

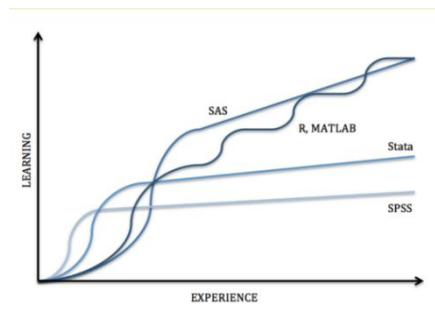
При избора между SPSS и R е добре да се има предвид мисловния модел (*mental model*), необходим за работа с двата софтуера. И двата мисловни модела изискват различни умения, които трябва да се тренират, за да е успешен преходът от студентския мисловен модел и този, необходим за боравене със софтуера [7].

Таблица 2. Сравнителна таблица между софтуерните продукти R и SPSS

TYPE OF STATISTICAL ANALYSIS	R	SPSS	TYPE OF STATISTICAL ANALYSIS	R	SPSS	TYPE OF STATISTICAL ANALYSIS	R	SPSS
Nonparametric Tests	Yes	Yes	Markov Chain Monte Carlo	Yes		Diffusions	Yes	
T-test	Yes	Yes	Bootstrap & Jackknife	Yes		Counting Processes	Yes	
ANOVA & MANOVA	Yes	Yes	EM Algorithm	Yes		Filtering	Yes	
ANCOVA & MANCOVA	Yes	Yes	Missing Data Imputation	Yes	Yes	Instrumental Variables	Yes	
Linear Regression	Yes	Yes	Outlier Diagnostics	Yes	Yes	Simultaneous Equations	Yes	
Generalized Least Squares	Yes	Yes	Robust Estimation	Yes		Splines	Yes	
Ridge Regression	Yes		Longitudinal (Panel) Data	Yes	Limit ed	Nonparametric Smoothing Methods	Yes	
Lasso	Yes		Survival Analysis	Yes	Yes	Extreme Value Theory	Yes	
Generalized Linear Models	Yes	Yes	Path Analysis	Yes		Variance Stabilization	Yes	
Mixed Effects Models	Yes	Yes	Propensity Score Matching	Yes		Cluster Analysis	Yes	Yes
Logistic Regression	Yes	Yes	Stratified Samples (Survey Data)	Yes	Yes	Neural Networks	Yes	Limit ed
Nonlinear Regression	Yes		Experimental Design	Yes		Classification & Regression Trees	Yes	Limit ed
Discriminant Analysis	Yes	Yes	Quality Control	Yes	Yes	Boosting Classification & Regression Trees	Yes	
Nearest Neighbor	Yes	Yes	Reliability Theory	Yes	Yes	Random Forests	Yes	
Factor & Principal Components Analysis	Yes	Yes	Univariate Time Series	Yes	Limit ed	Support Vector Machines	Yes	
Copula Models	Yes		Multivariate Time Series	Yes		Signal Processing	Yes	
Cross-Validation	Yes		Markov Chains	Yes		Wavelet Analysis	Yes	
Bayesian Statistics	Yes		Hidden Markov Models	Yes		ROC Curves	Yes	Yes
Monte Carlo, Classic Methods	Yes	Limit ed	Stochastic Volatility Models	Yes	Limit ed	Optimization	Yes	



Фиг. 3. Интерфейс на SPSS (от ляво) и R (от дясно)



Фиг. 4. Крива на изучаване (learning curve) при различните софтуерни пакети [5]

SPSS мисловният модел се базира на набор от данни. Типичният работен процес протича както следва:

- 1) импортиране или създаване на данни;
- 2) дефиниране на мета данни(meta data);
- 3) избор на анализи от меню;
- 4) други възможности за избор в самите анализи чрез диалогови прозорци;
- 5) произвеждат се много изходни данни;
- 6) чрез инструкции се улеснява тълкуването на изходните данни;
- 7) изходните данни могат да бъдат копирани и използвани в Word документи или друг тип файлове за създаване на финален отчет.

За обикновения потребител не се препоръчват персонализирането на статистически функции или ползването на изходни резултати като входящи данни за други анализи. По този начин, като цяло системата направлява потребителя в анализа.

За разлика от SPSS, в R потребителят направлява софтуера. Поради тази причина работния процес при R е следният:

- [1].настройка на необработени данни в друга програма;
- [2].импортиране на данни - най-често биват различни набори от данни, мета набор от данни и други обекти данни (вектори, таблични данни);
- [3].преобразуване на данните, използвайки набор от команди;
- [4].извършване на анализа - изпълнението на командите може да наложи търсене в Google или учебник, разбирането на аргументите в командите може да бъде улеснено от документация на софтуера или онлайн инструменти;
- [5].тъй като изходните резултати са минимални, често потребителят трябва да поиска конкретни данни чрез допълнителни команди;
- [6].много от стандартните възможности на SPSS изискват потребителски команди в R, но средно напреднал потребител може лесно да създаде множество команди, несъществуващи в SPSS; в R е далеч по-лесно извличането на част от изходните данни и ползването им като входни данни за други или следващи функции и анализи;
- [7].макар че крайните резултати могат да бъдат вградени в Word или Excel документи, софтуерът насърчава потребителите да участват в разнообразни работни процеси, които изтъкват възможностите на софтуера за възпроизведими научни изследвания.

R и SPSS също могат да бъдат сравнени по отношение на съществуващите ресурси. Има много онлайн и печатни ресурси както за R [14, 15, 16, 17], така и за SPSS [18, 19]. Съществуват и материали за конкретното приложение на R в психологическите изследвания, но те са далеч по-малко, отколкото материалите за SPSS [20].

5. Заключение

След направената съпоставка на двата софтуерни пакета за обработка на статистически данни, може да се направи заключението, че SPSS е подходящ за работа чрез менюта и лесен

интерфейс за стандартни анализи и е подходящ за голяма част от проучванията по психология, особено по време на следването в университета. R е по-подходящ за изследователи, които имат нужда от по-голям набор от статистически анализи, имат по-сериозни познания по програмиране и по-добра подготовка по статистика като цяло. Освен това, R е подходящ само за потребители, които го използват често, защото има по-дълъг период на обучение и много повече команди за запомняне. В този ред на мисли, SPSS е по-подходящ за потребители, които имат нужда от обработка на данни от време на време и особено подходящ за студентите по психология в бакалавърска и дори магистърска степен. Освен това SPSS е много по-широко прилаган в психологическите изследвания и е възможно изследвания с R да не бъдат подкрепени от научни ръководители или други членове на изследователски екипи. Имплементирането на R в учебната програма също би било предизвикателство и е препоръчително то да бъде насочено към по-високи степени на висшето образование (магистри, докторанти). Редица университетски преподаватели вече преосмислят варианти за реализирането на тази идея [7, 6], но основната академична общност все още се придържа към по-лесния за употреба SPSS. И все пак, нарастващата нужда от комплексни и многофакторни анализи има своята бъдеща алтернатива в лицето на този софтуер R, при това напълно безплатна.

Литература

- [1]. "Comparison of statistic packages," 17 10 2015. [Online]. Available: http://psychology.wikia.com/wiki/Comparison_of_statistical_packages.
- [2]. M. Remon, "Comparison of statistical software," 15 8 2014. [Online]. Available: <http://www.datasciencecentral.com/profiles/blogs/comparison-of-statistical-software>.
- [3]. „Statistical Language Wars,“ 03 06 2014. [Онлайн]. Available: <http://datacamp.wpengine.com/wp-content/uploads/2014/05/infograph.png>.
- [4]. ProStatServices, „A Review of the Top Five Statistical Software Systems,“ 2012. [Онлайн]. Available: <http://www.prostatservices.com/statistical-consulting/articles-of-interest/a-review-of-the-top-five-statistical-software-systems>.
- [5]. „Which Statistical Software to Use,“ 2015. [Онлайн]. Available: <https://sites.google.com/a/nyu.edu/statistical-software-guide/home>.
- [6]. B. O'connor, „Comparison of data analysis packages: R, Matlab, SciPy, Excel, SAS, SPSS, Stata,“ 23 02 2009. [Онлайн]. Available: <http://brenocon.com/blog/2009/02/comparison-of-data-analysis-packages-r-matlab-sciPy-excel-sas-spss-stata/>.
- [7]. J. Anglim, „Evaluating the Potential Incorporation of R into Research Methods Education in Psychology,“ 17 7 2013. [Онлайн]. Available: <http://jeromyanglim.blogspot.bg/2013/07/evaluating-potential-incorporation-of-r.html>.
- [8]. B. Chance, D. Ben-Zvi, J. Garfield и E. Medina, „The Role of Technology in Improving Student Learning of Statistics. Technology Innovations in Statistics Education,“ 2007. [Онлайн]. Available: [url: http://www.escholarship.org/uc/item/8sd2t4rr](http://www.escholarship.org/uc/item/8sd2t4rr).
- [9]. Г. Янков, „WISC-IV: Експлораторни и конфирмационни факторни анализи чрез R“, Седми научен конгрес по Психология - 31.10 - 02.11.2014, София, 2014.
- [10]. S. Tishkovskaya и G. Lancaster, „Statistical education in the 21st century: a review of challenges, teaching innovations and strategies for reform,“ *Journal of Statistics Education*, pp. 1-55, 2012.
- [11]. C. Franklin and J. Garfield, "The GAISE (Guidelines for Assessment and Instruction in Statistics Education) project: Developing statistics education guidelines for pre K-12 and college courses. In G. Burrill (Ed.), 2006 NCTM Yearbook: Thinking and reasoning with data and chance," National Council of Teachers of Mathematics, Reston, VA, 2006.
- [12]. „SPSS Statistics,“ 2015. [Онлайн].

Available: <http://www-01.ibm.com/software/analytics/spss/products/statistics/buy-now.html>.

[13]. „CRAN,“ 2015. [Онлайн]. Available: <http://cran.us.r-project.org/>.

[14]. J. Chambers, Software for Data analysis: programming with R, New York: Springer, 2008.

[15]. „R Bloggers,“ 2015. [Онлайн]. Available: www.r0bloggers.com.

[16]. M. Crawley, The R Book, John Wiley & Sons, 2012.

[17]. R. Kabacoff, R in Action, Manning Publications Co., 2011.

[18]. Field, Discovering statistics using SPSS for Window: Advanced techniques for the beginners, London: Sage Publications, 2000.

[19]. M. George, SPSS intermediate statistics use and interpretation, Netlibrary Inc., 2004.

[20]. N. Brace и R. Snelgar, SPSS for Psychology: a guide to data analysis using SPSS for Windows, L. Erlbaum Associates, 2000.

За контакти:

Славена А. Лазарова
катедра „Социални и правни науки”
Технически университет - Варна
E-mail: Slavena_lazarova@abv.bg

Вяра Н. Колева-Ефремова
катедра „Компютърни науки и технологии”
Технически университет - Варна
E-mail: viarakoleva@yahoo.com



IPv6 ADDRESS FROM VIN NUMBER

Radosław Wróbel, Hristo Valchanov

Abstract: The main aim of this paper is to propose an algorithm that allows conversion Vehicle Identification Number, which is used in “real” world to unequivocal identification of the vehicle, into IPv6. This conversion will create unique IPv6 number that will be assigned to the vehicle. Such solution will allow to unequivocal identification of the vehicle in “virtual” world as well. The algorithm will be implemented in the application (built for this paper) called VINIP. It makes the algorithm useful and additionally, correctness of it could be checked. Indirect goal is the investigation of the subjects related to IPv6 and vehicle identification.

Keywords: IPv6, VIN.

IPv6 адрес от VIN номер

Радослав Вробел, Христо Вълчанов

Резюме: Основната цел на този доклад е да предложи алгоритъм, който позволява конвертиране на идентификационния номер на превозното средство (VIN), който се използва в "реалния" свят за недвусмислена идентификация на превозното средство, в IPv6. Това преобразуване ще създаде уникален IPv6 адрес, който ще бъде присвоен на превозното средство. Такова решение ще позволи недвусмислената му идентификация във "виртуалния" свят. Алгоритъмът ще се реализира в приложение (създадено за целта) и наречено VINIP. Това прави алгоритъма полезен и допълнително осигурява възможност за проверка на коректността му. Непряка цел е изследването на темите, свързани с IPv6 и идентификация на превозното средство.

Ключови думи: IPv6, VIN.

Introduction

The main aim of this paper is to propose an algorithm that allows conversion Vehicle Identification Number, which is used in “real” world to unequivocal identification of the vehicle, into IPv6. This conversion will create unique IPv6 number that will be assigned to the vehicle. Such solution will allow to unequivocal identification of the vehicle in “virtual” world as well. The algorithm will be implemented in the application (built for this paper) called VINIP. It makes the algorithm useful and additionally, correctness of it could be checked. Indirect goal is the investigation of the subjects related to IPv6 and vehicle identification.

Internet Protocol version 6 – known as IPv6. Is the future of data transmission via the Internet. The protocol is a successor of well know, very common and broadly used Internet Protocol version 4 – IPv4 [2 – 4]. The purpose why IPv4 must be replaced by newer version is trivial – Ipv4 is running out of network addresses. With constantly increased demands of internet features, need of IP addresses has to be fulfilled.

Internet Protocol

IPv4 was developed in the beginning of 1970s. [1 - 4]. Initially was used to communication between academics and government researches in the United States. Probably even IPv4 inventors did not realize how rapid and broad will be internet usage. Web had grown quickly far beyond mailing and web pages support. More and more IP addresses were needed. IPv4 has finite number of addresses that is equal 232 (4 294 967 296) [8, 9]. In that time this number was consider as sufficient. Reality turned out different and scientists were forced to develop IPv6. Computer’s era

had begun. Companies such as Microsoft, Apple, IBM started to race with each other which companies will produce faster, smaller and more powerful computer. What is more they started to produce computer-similar devices such as laptops, netbooks and tablets. In the meantime mobilephones became internet connected. Every of these devices needed its private, unique IP address. Hence demands on IP addresses was growing equally with amounts of electronic devices that were able to connect to the internet. That is how IPv6 was born.

The one of the main difference between IPv4 and IPv6 is length of the address. IPv6 address format has 128 bits so 2¹²⁸. Unbelievable amount. It can be consider as infinite number.

Differences between IPv4 and IPv6 are [12, 13, 17, 19]:

- Extended address space – from 32 bits in IPv4 to 128 bits in IPv6
- Autoconfiguration – network prefix can be taken by booting device from an IPv6 router on its link. Thanks to this prefix information device can be configured for one or more valid global IP addresses by using private random number that allow to build a unique IP address or its MAC identifier can be used as well. In IPv4 technology, unique IP address needs to be assigned to every device by using Dynamic Host Configuration Protocol (DHCP) or by manual configuration. In other words it means that for example if new device with its own IP address would be added to private DHCP server in IPv4 technology, the whole server would need to be reconfigured. Thanks to IPv6 it is done automatically.

- Simplification of header format – fixed header length in contrast to IPv4. 40 bytes (two times 16 bytes for the Source and Destination addresses and 8 bytes for header information).

- Improved support for options and extensions – extension header introduced in IPv6 that allows faster processing.

In IPv6 two types of host configuration are distinguish [1, 14, 21, 27]:

- Stateless Address Auto configuration – base element of auto configuration system
- Stateful Address Auto configuration – provide advanced control on assigned addresses.

Dynamic Host Configuration Protocol (DHCP) is the protocol that is responsible for addresses allocation and management. The stateful configuration is the enlargement of stateless configuration and works only together

For first time, IETF organization distinguished 3 types of addresses IPv6:

- Link-local address is used during communication between the neighbor nodes on the same link. Such address is valid only within the local network segment connected with the interface. Link-local address starts with FE80. First bits are set to 1111 1110 10 in order to identify such addresses. Link-local address is automatically configured and is required for Neighbor Discovery process [10, 15, 16]. The address is not routable. Hence there is no possibility to forward that address outside the link.

- Site-local address is not reachable from other sites. Hence it is prohibited to forward site-local traffic outside the site by routers. Site-Local addresses are used in case there is no direct, routed connection to the IPv6 internet for example by intranet. In such case intranet can use site-local address not interfering with global address [1 – 8, 15].

- Global IPv6 addresses are globally reachable in the Internet [1-4, 8, 9, 11, 21, 27].

Interface ID is unique within a subnet prefix. It identifies interface on a link. It is not recommended to assign the same ID to different nodes on a link. There is an exception from that if interface ID is directly taken from interface's link-layer address. In such case if multiple interfaces are attached to different subnets there is possibility to use the same interface ID for these interfaces.

Vehicle Identification Number

A vehicle marking system was introduced in early 80' last century [23]. Because a big increase in car usage there was a need to invent one broadly used system for car marking. The National Highway Traffic Safety Administration established demand that 17- characters number,

currently known as vehicle identification number, has to be assigned to every car produced in this time and in the future. Now the VIN is commonly use almost all over the world. Every car produced in last 30 years is marked by VIN. There is no two the same VIN in all over the world hence a car can be clearly identify only by that number. Considering a car as a human being, VIN would be a DNA. Every information that is needed to unequivocal car and engine, that the car is equipped with, identification is incorporated in this number. VIN may consist only capital letters (without polish ones and I, O, Q) and Arabic numbers 0 – 9. Vehicle Identification number can be divided into three parts [22, 23]:

- World Manufacturing Identifier (WMI). International manufacturer code. WMI is represented by first three signs in VIN. According to ISO 3780/PN-92/S-02062 norm WMI number is allocated to the vehicle by the proper national organization in the country where the vehicle's manufacturer has his headquarters.

- Vehicle Descriptor Section. VDS consists of six signs which describes vehicle's features.

- Vehicle Indicator Section (VIS). VIS consists of eight characters which describe particular model of the car.

VIN CONVERSION TO IPV6

Nowadays vehicles are equipped with more and more devices to increase ride quality and improve maintenance of the car. Such process is becoming more common every day – proportional to the technology developing. Cars manufacturers are racing with themselves in vehicle equipping with more sophisticated devices. Simultaneously with IT technology spreading and popularity and usefulness of internet there is an option to connect all cars accessory to the internet. It creates a lot of possibility thank to them vehicles become more modern and self-sufficient. Internet connection allows to current, in real time, monitoring of every factor such as fuel level, ABS system state, temperature of engine, vehicle's elements wear and many more, without vehicle leaving. Every parameters may be available on the control panel fixed in the car. What is more such solution may improve safety on the road by restraining car's speed if detect any abnormalities in car operation or exceeded wear of any element in the car. Sensors will be fitted on the road as well hence communication between these sensors and sensors in the vehicle may prevent car accidents. Entertainment – another advantage of IPv6 usage.

Problem determination

In binary notation every character from hexadecimal notation is represented as 4 bits (e.g. 0 hex = 0000 bin; F hex = 1111 bin).

The characters that can be used in VIN number are: 0–9 = 30–39 hex (ASCII), A–Z = 41–5A (without O, I, Q, ASCII) [18].

It is visible from above that all characters that are used to VIN creation, in hexadecimal notation, consist of 2 signs. Hence it is known that every character that is used in VIN is represented as 8 bits in binary notation.

Because of that there is no possibility to convert the whole VIN into IPv6. VIN consists of 17 characters. 17 multiple by 8 is equal 136- at least such bits are needed to reflect the whole VIN. Not saying about prefix that needs another free bits. So straight VIN conversion into IPv6 is not possible.

After closer analysis of the problem and VIN as well it has turned out that to fully determine vehicle, characters from 9 to 17 are enough. These 9 characters require 72 free bits. Thanks to this 56 bits remain for prefix and special addresses.

An application project

Presented scheme (fig. 1) will be used as a pattern for algorithm creation. The diagram does not get into details. Procedures and functions are not taken into account in the diagram. The ones that will be unusual and worth to mentioned will be described in further part of the thesis.

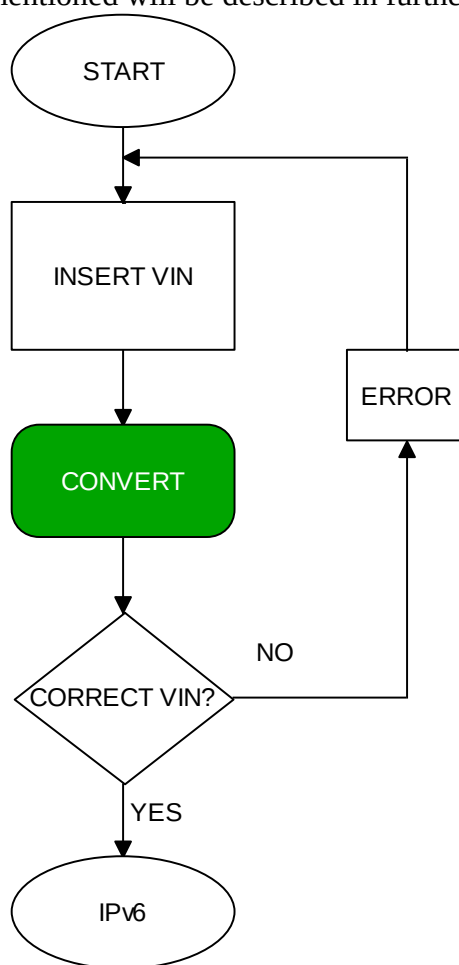


Fig. 1. An application blocks diagram

Having information captured above, such as assumptions and blocks diagram a converter “VINIP” was invented. The converter has been made using C# programming language. Applications written in C# are desired for Windows operation system [20].

The conversion operation – main part of the VINIP application is presented in Listing 1 on Fig.2. There are some likely mistakes, if made, may have influence on the conversion and cause wrong IPv6 representation. In order to prevent that VINIP is equipped with two protections. First of them checks correctness of the VIN (Fig.3, Fig.4).

```
for (int i = 0; i < ASCII.Length; i++)
{
    Hex += String.Format("{0:X}", Convert.ToInt32(ASCII[i]));

    if (i % 2 == 0 && (i + 1) < ASCII.Length)
        Hex += ":";
}
```

Fig. 2. Main part of VINIP program (C#)

```
private bool CheckVINAddress(string VIN)
```

Fig. 3. Checking of proper VIN no (C#)

```
return Regex.IsMatch(VIN, @"^[a-zA-Z0-9]{16}[a-zA-Z0-9]${1}") &&  
Regex.IsMatch(VIN, @"^[^ioqIOQ]{16}[^ioqIOQ]${1}");
```

Fig. 4. Checking of forbidden VIN characters (C#)

Clicking Convert button starts the program working and the VIN conversion. In case of proper VIN, below appears VIN converted into IPv6. Such scenario is presented on fig.5.

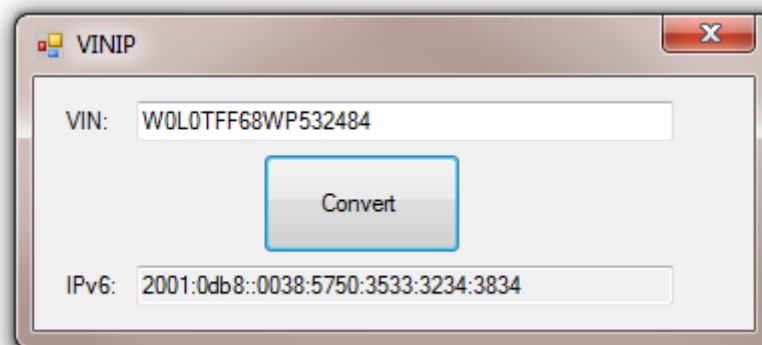


Fig. 5. Usage of the application

Заклучение

It is important to know that is it only prototype. What is more quite simply Internet Engineering Task Force (IETF) has been facing such problem for couple of years and till now it still remains unresolved [24]. Presented program can be solution of the problem. Indeed, when it comes about functionality the application meets IETF requirements and convert VIN into IPv6. But it is not enough so that recognize VINIP application as a functional.

There is much more problems that are not solved, such as for example security in the Internet (eve if IPsec is mandatory in IPv6 [3, 5, 26]) . It is highly prioritize problem and has to be meet if one is talking about global spreading of the application.

Production cost of such application is equal zero. All necessary tools needed to an application creation are provided by cooperation with TU Varna.

References

- [1]. Bires J., Esa L. A., IPv6 Primer – a concise and practical guide for IT professionals and students, Bires J. Esa L. 2015.
- [2]. Cicileo G., Gagliano R., IPv6 for all, ISOC.Ar 2009.
- [3]. Coffeen T., IPv6 Address Planning, Designing an address plan for the future, O'Reilly Media 2014.
- [4]. Comer E. Douglas, Interworking with TCP/IP Volume One (6th edition, Pearson 2013.
- [5]. Dalal M., IPv6: Powering the next generation article.
- [6]. Davies J., Understanding IPv6 3rd edition, Microsoft Press 2012.

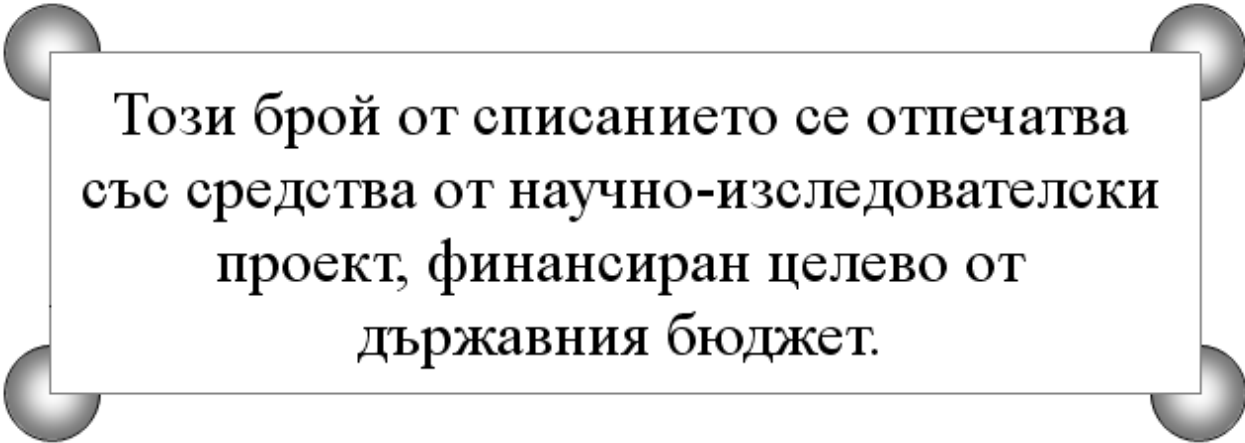
- [7]. Diaz J. Lazaro, Unraveling the mysteries of IPv4 & IPv6, CreateSpace Independent Publishing Platform 2014.
- [8]. Fall R. Kevin, Stevens W. Richard, TCP/IP Illustrated, Volume 1: The Protocols (2nd Edition), Addison-Wesley Professional 2011.
- [9]. Graziani R., IPv6 Fundamentals: A Straightforward Approach to Understanding IPv6, Cisco Press 2012.
- [10]. Hagen S. IPv6 Essentials 3rd edition, Published by O'Reilly Media 2014.
- [11]. Hughes E. Lawrence, The second internet- Reinventing Computer Networking with IPv6, Publisher: InfoWeapons.
- [12]. Loshin P., IPv6 Theory, protocol and practice 2nd edition, Morgan Kaufmann 2003.
- [13]. Lear IPv6, www.tutorialspoint.com.
- [14]. RFC2460 Internet Protocol, Version 6 (IPv6) Specification.
- [15]. RFC3587 IPv6 Global Unicast Address Format.
- [16]. RFC4291 IP Version 6 Addressing Architecture.
- [17]. <http://ipv6.com>.
- [18]. http://nss.et.put.poznan.pl/study/projekty/sieci_komputerowe/ipv6/html/adresacja.htm.
- [19]. [https://technet.microsoft.com/pl-pl/library/cc736439\(v=ws.10\).aspx](https://technet.microsoft.com/pl-pl/library/cc736439(v=ws.10).aspx).
- [20]. http://4programmers.net/C_sharp.
- [21]. <http://www.tutorialspoint.com>.
- [22]. <http://www.autocheck.com>.
- [23]. <http://www.carfax.com/company/vehicle-identification-numbers-vins>.
- [24]. <https://tools.ietf.org/html/draft-imadali-its-vinipv6-viid-00>.
- [25]. <http://www.techietek.com/tag/ipv6>.
- [26]. <http://docs.oracle.com/cd/E19253-01/816-4554/ipv6-overview-170/index.html>.
- [27]. <http://www.carfax.com>.

For contacts:

PhD Eng. Radoslaw Wrobel
Wroclawska Politechnika
E-mail: wrobel@os.pl

ИЗИСКВАНИЯ ЗА ОФОРМЯНЕ НА СТАТИИТЕ ЗА СПИСАНИЕ "КОМПЮТЪРНИ НАУКИ И ТЕХНОЛОГИИ"

- I. Статиите се представят разпечатани в два екземпляра (оригинал и копие) в размер до 6 страници, формат А4 на адрес: Технически университет – Варна, ФИТА, ул. „Студентска” 1, 9010 Варна, както и в електронен вид на имейл адреси: ned.nikolov@tu-varna.bg или yulka.petkova@tu-varna.bg.
- II. Текстът на статията трябва да включва: УВОД (поставяне на задачата), ИЗЛОЖЕНИЕ (изпълнение на задачата), ЗАКЛЮЧЕНИЕ (получени резултати), БЛАГОДАРНОСТИ към сътрудниците, които не са съавтори на ръкописа (ако има такива), ЛИТЕРАТУРА и информация за контакти, включваща: научно звание и степен, име, организация, поделение (катедра), e-mail адрес.
- III. Всички математически формули трябва да са написани ясно и четливо (препоръчва се използване на Microsoft Equation).
- IV. Текстът трябва да бъде въведен във файл във формат WinWord 2000/2003 с шрифт Times New Roman. Форматирането трябва да бъде както следва:
 1. Размер на листа - А4, полета: ляво - 20мм, дясно - 20мм, горно - 15мм, долно - 35мм, Header 12.5мм, Footer 12.5мм (1.25см).
 2. Заглавие на български език - размер на шрифта 16, удебелен, главни букви.
 3. Един празен ред - размер на шрифта 14, нормален.
 4. Имена на авторите - име, инициали на презиме, фамилия, без звания и научни степени - размер на шрифта 14, нормален.
 5. Два празни реда - размер на шрифта 14, нормален.
 6. Резюме и ключови думи на български език, до 8 реда - размер на шрифта 11, нормален.
 7. Заглавие на английски език - размер на шрифта 12, удебелен.
 8. Един празен ред - размер на шрифта 11, нормален.
 9. Имена на авторите на английски език - размер на шрифта 11, нормален.
 10. Един празен ред - размер на шрифта 11, нормален.
 11. Резюме и ключови думи на английски език, до 8 реда - размер на шрифта 11, нормален.
 12. Основните раздели на статията (Увод, Изложение, Заключение, Благодарности, Литература) се формират в едноколонен текст както следва:
 - a. Наименование на раздел или на подраздел - размер на шрифта 12, удебелен, центриран, един празен ред преди наименованието и един празен ред след него - размер на шрифта 12, нормален;
 - b. Текст - размер на шрифта 12, нормален, отстъп на първи ред на параграф – 10 мм; разстояние от параграф до съседните (Before и After) за целия текст – 0.
 - c. Цитиране на литературен източник - номер на източника от списъка в квадратни скоби;
 - d. Текстът на формулите се позиционира в средата на реда. Номерация на формулите - дясно подравнена, в кръгли скоби.
 - e. Фигури - центрирани, разположение спрямо текста: "Layout: In line with text". Номер и наименование на фигурата - размер на шрифта 11, нормален, центриран. Отстояние от съседните параграфи – 6 pt.
 - f. Литература – всеки литературен източник се представя с: номер в квадратни скоби и точка, списък на авторите (първият автор започва с фамилия, останалите – с име), заглавие, издателство, град, година на издаване, страници.
 - g. За контакти: научно звание и степен, име, презиме (инициали), фамилия, организация, поделение (катедра), e-mail адрес, с шрифт 11, дясно подравнено.Образец за форматиране можете да изтеглите от адрес <http://cs.tu-varna.bg/> - Списание КНТ, Spisanie_Obrazec.zip.



Този брой от списанието се отпечатва
със средства от научно-изследователски
проект, финансиран целево от
държавния бюджет.