Department of Computer Science
Series of Publications B
Report B-2012-3

Proceedings of AMICT 2010-2011 Advances in Methods of Information and Communication Technology

Jussi Kangasharju and Iurii A. Bogoiavlenskii (Editors)

University of Helsinki
Finland

Contact information

Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
FI-00014 University of Helsinki
Finland

Email address: postmaster@cs.helsinki.fi
URL: http://www.cs.helsinki.fi/
Telephone: +358 9 1911, telefax: +358 9 191 51120

Department of Computer Science Publications
Series of Publications B
Report B-2012-3
ISSN 1458-4786
ISBN 978-952-10-8565-9 (PDF)
Helsinki 2012

Contents

Preface **1**

Prof. Igor L. Bratchikov, Alexei A. Popov
Typewritten symbols recognition using Genetic Programming **5**
1 Introduction 6
2 Typewritten symbols recognition 7
3 Conclusion 13

Dr. Nikolai K. Krivulin
Algebraic Solutions to Scheduling Problems in Project Management **15**
1 Introduction 15
2 Definitions and Notation 16
3 Preliminary Results 18
4 Applications to Project Scheduling 21
5 Conclusion 28
6 Acknowledgments 28

Prof. Evsei V. Morozov, Alexander S. Rumyantsev
Moment properties and long-range dependence of queueing processes **31**
1 Introduction 31
2 Long range dependence in tandem 34
3 Conclusion 36
4 Acknowledgments 37

Matti Paksula
Introduction to store data in Redis, a persistent and fast key-value database **39**
1 Introduction 39
2 Architectural motivation 40
3 KVS vs RDBMS 40
4 Persisting objects in KVS 41
5 Key-value storages 42
6 Redis introduction 43
7 Implementation 44
8 Conclusion 47

Prof. A.V. Sokolov, A.V. Drac
The optimal implementation of n FIFO-queues in single-level memory **51**
1 Introduction 51
2 The problem 52
3 Consecutive allocation of the queues 53
4 Linked list presentation of queues 56
5 Comparison between consecutive and linked list presentations 63

Alexander S. Rumyantsev, Luybov V. Potakhina
Optimizing performance in heavy-tailed system: a case study **67**
1 Introduction 67
2 Choosing a service discipline 68
3 Choosing a server architecture 70
4 Choosing a task assignment policy 72
5 Conclusion 73
6 Acknowledgments 74

O. V. Lukashenko, R. S. Nekrasova, E. V. Morozov, M. Pagano
Some analytical aspects of regenerative simulation of fluid models **75**
1 Introduction 75
2 Regenerative approach 76
3 Numerical examples 77
4 Conclusion 80

Oleg V. Lukashenko, Mikhail Nasadkin
Simulation of the fluid system with long-range dependent input **81**
1 Introduction 81
2 Queue with long-range dependent input 82
3 Simulation 83
4 Conclusion 86

Tiina Niklander
How to avoid plagiarism? **87**
1 Introduction 87
2 Aspects of plagiarism 88
3 Copy detection tools 89
4 Turnitin 89

5 Avoiding unintentional plagiarism 91
6 Conclusion 92

R. S. Goricheva, O. V. Lukashenko, E. V. Morozov, M. Pagano
Regenerative simulation of the loss probability in the finite buffer queue with Brownian input **95**
1 Introduction 95
2 Queue with Brownian input 96
3 Regenerative method 96
4 Regenerative structure and simulation results 98
5 Conclusion 99

Preface

The Annual International Seminar "Advances in Methods of Information and Communication Technology" AMICT is a direct continuation of the "Finnish Data Processing Week at the Petrozavodsk State University". FDPW was originally started in 1994 as a guest lecturers' week, where the presentations came from the University of Helsinki. However, the frame of FDPW very soon developed to a research-oriented seminar with presentations from various Finnish, Karelian, and Russian computer science departments, research institutions, and even from industry. An interested reader finds a more detailed description of the FDPW history on the www page http://www.cs.karelia.ru/fdpw/index.php.en.

This eleventh volume of the Proceedings of the Annual Finnish Data Processing Week / Advances in Methods of Information and Communication Technology contains selected presentations from two seminars, AMICT 2010 organized 25.–26.5. 2010, and AMICT 2011 organized 27.–28.5. 2011.

In 2010 we also had the honor of having a long-standing collaborator of the seminar, Prof. Timo Alanko from University of Helsinki, attending the seminar as a special guest.

The two seminars combined consisted of 15 technical presentations, 4 keynote speeches, and a demo session in AMICT 2010 showing student projects with Maemo. Participants in the seminars came from University of Helsinki, Petrozavodsk State University, Saint Petersburg State University, and the Karelian Research Center of the Russian Academy of Science. Out of the seminar presentations, these proceedings contain 10 papers from the technical paper sessions.

I. L. Bratchikov and A. A. Popov use genetic programming in their article to recognize typewritten symbols. Genetic programming is a special case of genetic algorithms and has the potential to be used to optimize computer programs.

N. K. Krivulin's article focuses on scheduling problems in project management and proposes algebraic solutions. The solutions are given in a compact vector form and provide a basis for developing efficient algorithms.

E. V. Morozov and A. S. Rumyantsev study moment properties and long-range dependence of queueing processes. They empirically extend known results on long-range dependence to a 2-station tandem system.

M. Paksula presents an introduction into how Redis, a persistent and fast key-value database stores data. The article also provides a pattern to persist simple objects. This pattern focuses on consistency under race and failure conditions.

The paper of A. V. Sokolov and A. V. Drac considers the optimal implementation of FIFO queues in single-level memory. Their solution is based on random walks into different areas of n-measured space.

A. S. Rumyantsev and L. V. Potakhina present a case study for optimizing performance in a heavy-tailed system. They consider three cases: choosing an optimal queueing policy, switching to a multi-server system, and choosing a task assignment policy. Their goal is to minimize effect of heavy tails on system performance.

O. V. Lukashenko et al. discuss analytical aspects of regenerative simulation of fluid models. Their approach combined regenerative methods with the so-called Delta-method.

Fluid systems are also the topic of another paper by O. V. Lukashenko and M. Nasadkin, where they simulate fluid systems with long-range dependent input. They focus on fractional Brownian input because its properties match that of the network traffic to a certain degree.

T. Niklander's article gives an overview of modern plagiarism detection tools and presents a more detailed overview of the system TurnItIn. The article also gives many helpful hints on how to avoid unintentional plagiarism.

R. S. Goricheva et al. present regenerative simulation of the loss probability in a finite buffer queue with Brownian input.

We want to express our sincere gratitude to the Rector of the Petrozavodsk State University, professor Anatoly V. Voronin, the President of the University, professor Victor Vasiljev, and to the vice-rectors Natalia S. Ruzanova, and Natalia V. Dorshakova, who all have provided both organizational and financial support to the seminars. We are grateful to professor Esko Ukkonen, the head of the Department of Computer Science of the University of Helsinki, for the support to the publication of these proceedings.

We also want to thank everyone involved in organizing the seminars. The burden of the event organization in Petrozavodsk was shared by Dmitry Korzun and Dmitry Chistyakov in 2010 and 2011 as well as by Natalia Kravchenkova in 2011. In Helsinki, the organization was handled by Tiina Niklander.

The technical editing of the proceedings was carefully done by Liang Wang and Tiina Niklander (University of Helsinki). This edition of the proceedings is the first to be published electronically in the University of Helsinki electronic publication series.

We also gratefully acknowledge the remarkable support we traditionally have had from the International Offices of our Universities personified by

Päivi Tauriainen and Ljudmila Kulikovskaya.

Dr. Iurii Bogoyavlenskiy, Head of the Department of Computer Science
of the Petrozavodsk State University

Dr. Jussi Kangasharju, Department of Computer Science
of the University of Helsinki

Typewritten symbols recognition using Genetic Programming

Prof. Igor L. Bratchikov, Alexei A. Popov

Faculty of Applied Mathematics and Control Processes,
Saint-Petersburg State University

Universitetskii prospekt 35, Petergof, Saint-Petersburg, 198504, Russia

E-mail: {braigor@yandex.ru, popov.lex@mail.ru}

Abstract

Genetic programming is a new technique in Artificial Intelligence based on the evolutionary algorithms and inspired by biological evolution. As a matter of fact, genetic programming is a special case of genetic algorithms, where each individual is a computer program. Therefore, this technique could be used to optimize a population of computer programs to solve the problem.

This report demonstrates how genetic programming can be used to solve the problem of optical character recognition, specifically typewritten symbols. At present, there are many approaches to solve this problem, but all of them have their own limitations.

The approach given in this report could be successful at learning, maintaining and upgrading rules for typewritten symbols recognition, particularly in disputable situations. Specific fitness functions, terminals and functions that satisfy the requirements of a main problem were considered.

This research presents the successful application of genetic programming to a difficult and topical task.

1 Introduction

Genetic programming (GP) is a new technique in Artificial Intelligence based on the evolutionary algorithms and inspired by biological evolution. Basically it is a special case of genetic algorithms, where each individual is a computer program (usually represented in memory as a tree structures). This technique could be used to optimize a population of computer programs to solve the problem.

This paper investigates the use of genetic programming for typewritten symbols recognition. The term 'recognition' means the mechanical or electronic translation of scanned images of printed or typewritten symbols into machine-encoded text. In practice, it is extremely hard to generate, maintain and upgrade the system that would be successful in solving the problem of the character recognition especially as such system would give the human-competitive results. Generally there is a rule set for each symbol that is presumably true only for that symbol. The main point is that any changes in a current rule set have to be tested on very large sets to insure that all examples of the symbol are accepted and all others (wrong ones) are rejected. Therefore it would be great to design and develop the system that could easily, fluently and correctly recognize any typewritten characters.

The main purpose of the research is to estimate the application of GP for the problem of typewritten symbols recognition.

The principle goals are the following:

- To determine the superiorities (advantages) and disadvantages of GP in comparison with the other approaches;
- To design and develop the terminals and functions, fitness measure, certain parameters for controlling the run, the termination criteria and method for designating the results of the run.

GP has been successfully applied to the simple character recognition problem [2, 3, 4, 5, 6]. John Koza evolved programs that could recognize an 'I' and a 'L' letters using Boolean templates and control code for moving the templates. This approach involved low-resolution characters, e.g. 4x6 or 5x5. Later on Andre extended this approach by using programs that were good for recognition of digits using co-evolved two-dimensional feature detectors [1, 2]. However, this approach involved low-resolution symbols as well. So both approaches were good at recognition the limited symbol sets only.

Therefore the general task is to define whether GP is a successful technique in solving the problem of character recognition or not. And in case of positive result, the principal goal is to apply such technique by designing and developing the GP-based system that could solve the problem of typewritten symbol recognition. Moreover, such a system can be regarded as the first stage of the development of a system for recognition of handwritten symbols.

Let us consider the main principles of GP, its specificity, features and benefits.

2 Typewritten symbols recognition

2.1 What is GP?

Let us continue by saying a few words about GP. Basically it is an automated method for creating a computer program from a high-level problem statement of a specific task. Genetic programming starts from a statement of "what needs to be done" and automatically creates a computer program to solve the problem. GP is a machine learning technique used to optimize a population of computer programs according to a fitness landscape designed to evaluate the program's ability to solve a given computational task [4].

GP evolves computer programs traditionally represented in memory as tree structures. Trees can be easily evaluated in a recursive manner. Every tree node has an operator function and every terminal node has an operand that makes mathematical expressions easy to evolve and evaluate. Thus traditionally GP favors the use of programming languages that naturally embody tree structures (for example, Lisp; other functional programming languages are also suitable).

The fact that genetic programming can evolve entities that are competitive with human-produced results suggests that genetic programming can be used as an automated invention machine to create new and useful patentable inventions.

2.2 How does it work?

One of the central challenges of computer science is to force a computer to do what needs to be done, without specifying how to do it. Genetic programming starts with a set of thousands of randomly created computer programs. This population of programs is progressively evolved over a series of generations. The evolutionary search uses the Darwinian principle

of natural selection (survival of the fittest) and analogs of various naturally occurring operations, including crossover (sexual recombination), mutation, gene duplication, gene deletion. Sometimes genetic programming also employs developmental processes by which an embryo grows into fully developed organism.

Therefore GP is a domain-independent method that genetically breeds a population of computer programs to solve a problem. Specifically, such technique iteratively transforms a population of computer programs into a new generation of programs by applying analogs of naturally occurring genetic operations. In addition, GP can automatically create a general solution to a problem in the form of a graphical structure whose nodes or edges represent components and where the parameter values of the components are specified by mathematical expressions containing free variables.

However to achieve the goal the user (human) has to specify the following preparatory steps:

1 the set of *terminals* (the independent variables of the problem) for each branch of the program;

2 the set of *functions* for each branch of the program;

3 the *fitness measure* (for implicitly or explicitly measuring the fitness of individuals in the population);

4 *certain parameters for controlling the run*;

5 *termination criterion* and *method for designating the result of the run.*

The figure below shows the aforesaid preparatory steps for the basic version of genetic programming. Those steps are the user-supplied input to the GP-system. The computer program represents the output of the genetic programming system.

Genetic programming iteratively transforms a population of computer programs into a new generation of the population by applying analogs of naturally occurring genetic operations. These operations are applied to some individuals selected from the population. They are stochastically selected to participate in the genetic operations based on their fitness (the third preparatory step). The iterative transformation of the population is executed inside the main generational loop of the run of genetic programming.

The first two preparatory steps specify the ingredients that are available to create the computer programs. A run of genetic programming is

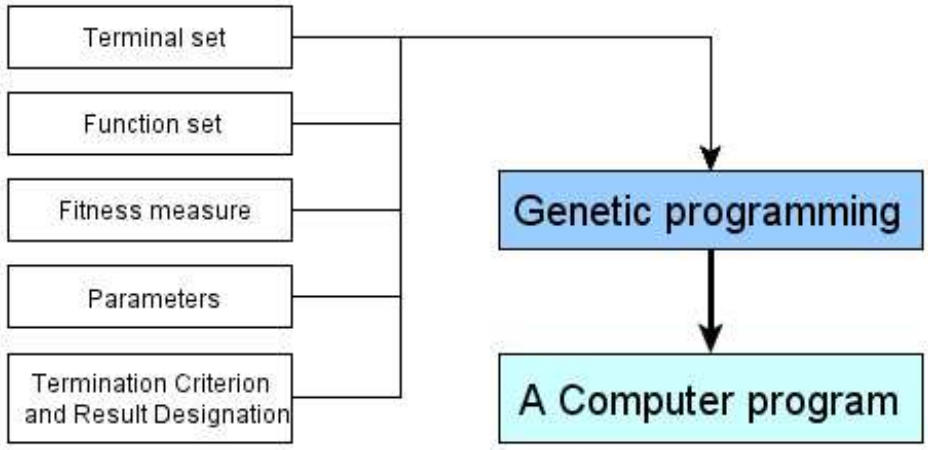


Figure 1: How does GP work

a competitive search among a diverse population of programs composed of the available functions and terminals.

The important thing to remember is that GP is a highly iterative process. It involves nested loops of procedures. The goal is to evolve successively better individuals with every generation. Theoretically we might want to run GP indefinitely, but due to limited computational power, we need to define some kind of termination criteria.

Generally the termination criteria has two parts, a successful fitness or a maximum number of generations. We provide GP with some number and say, if you evolve an individual with fitness better than or equal to that number, accept it and stop the run. This means that we have found a solution that is good enough. Alternatively, we want to stop GP from running too long if it is not progressing. We find that the probability of making further progress drops with number of generations passed, so we define some maximum number of generations. If we haven't succeeded by so many generations, it might be time to stop and try again.

The whole procedure can be described as follows:

1. Randomly create an initial population (generation 0) of individual computer programs composed of the available functions and terminals.

2. Iteratively perform the following sub-steps (called a generation) on the population until the termination criterion is satisfied:

    a Execute each program in the population and ascertain its fitness

(explicitly or implicitly) using the problem's fitness measure.

b Select one or two individual program(s) from the population with a probability based on fitness (with reselection allowed) to participate in the genetic operations in (c).

c Create new individual program(s) for the population by applying the following genetic operations with specified probabilities:

I *Reproduction*: Copy the selected individual program to the new population.

II *Crossover*: Create new offspring program(s) for the new population by recombining randomly chosen parts from two selected programs.

III *Mutation*: Create one new offspring program for the new population by randomly mutating a randomly chosen part of one selected program.

IV *Architecture-altering operations*: Choose an architecture-altering operation from the available repertoire of such operations and create one new offspring program for the new population by applying the chosen architecture-altering operation to one selected program.

3. After the termination criterion is satisfied, the single best program in the population produced during the run (the best-so-far individual) is harvested and designated as the result of the run. If the run is successful, the result may be a solution (or approximate solution) to the problem.

2.3 Adding GP to the problem

Please see the flowchart of GP in Figure 2.

Now let's get back to our research, and especially to the character recognition problem. The data source was database of typewritten symbols. It consists of testing and training subsets.

The symbols are centered and represented by a matrix of black and white pixels (20 pixels wide and 30 pixels in height). The first step in the recognition of any symbol is to extract the boundary pixels from the bitmap. This can be done using a quick one-pass raster scan method [1] Each element in a list contains row and column information for a boundary pixel. The second step is to close holes that are small enough to be accounted for by noise. The outer boundary of the symbol is then split into four segments (left, right, bottom and top). Therefore, the maximum and minimum row

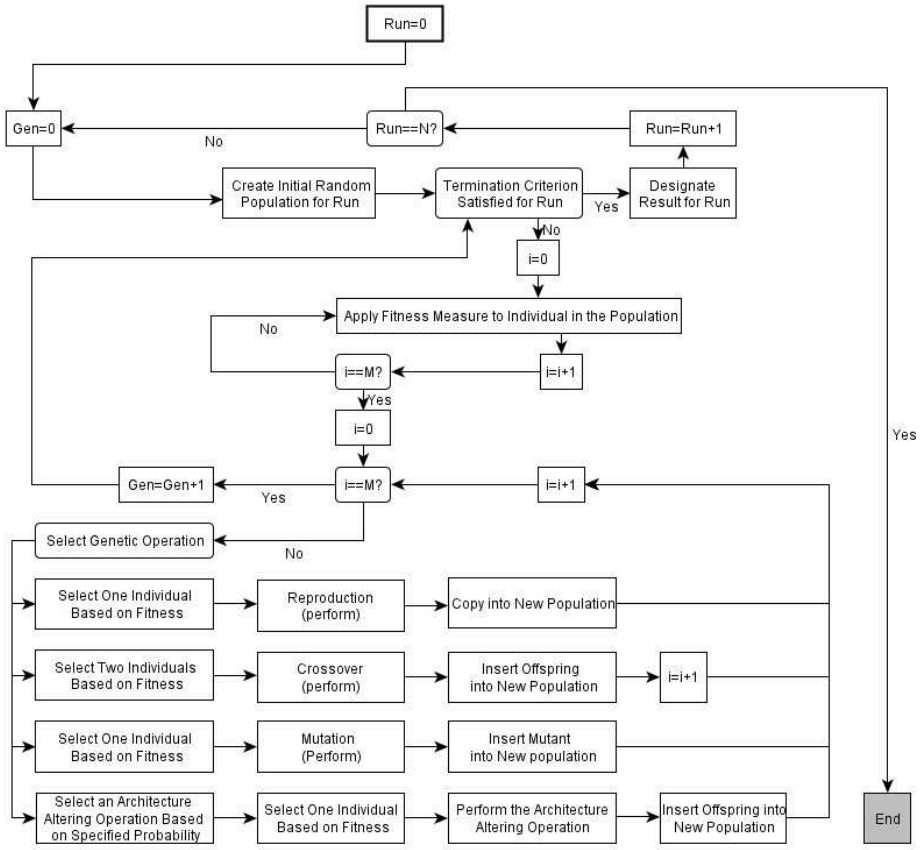


Figure 2: Typical GP scheme

and column are calculated for each hole, for each segment, and for the symbol as a whole. In addition, the number of pixels in each hole stored, and the segments are ranked according to their number of pixels.

It is assumed that solutions are always comparable and that, given a pair of them, we are always able to point the better one, unless they have the same value of the evaluation function.

For some hypothesis the evaluation function returns its accuracy of classification on the training set. Incomparability involves a partial order in the solution space and the possibility of existence of many best solutions at the same time. We can prevent the algorithm from losing good solutions by replacing the scalar evaluation function with a pairwise comparison of solutions.

Let's define formally the outranking relation between two solutions (hy-

potheses), given the sets of examples correctly classified by these hypotheses. Outranking means that first hypothesis is at least as good as a second one [5]. This condition has to hold separately and simultaneously for examples representing some decision classes. Therefore someone might ask an obvious question, how to select the best solutions?

Tournament selection scheme cannot work properly in solving this problem due to the fact, that the incomparability decreases the selection pressure, so some tournaments might remain undecided. Therefore we have to select some non-outranked solutions (hypotheses).

The solutions (programs-candidates) performing image analysis and recognition are evaluated on a set of training cases (images), called fitness cases. Let us now consider some estimated values. Thus, the population size was 2000 (that's a quite enough indeed). The probability of mutation equals to 0.05 (it is a common or standard value). The value of maximal depth of a randomly generated tree (initialization): 3 or 4 (it depends in common case). Maximal number of generations: 100 (stopping condition; in some cases this value could be increased or decreased). Training set size equals to 200 (100 images per each class). We used the tournament selection which means that the selection works by selecting a number of individuals from the population at random, a tournament, and then selecting only the best of those individuals. Tournament sizes tend to be small relative to the population size. The ratio of tournament size to population size can be used as a measure of selective pressure. Note that a tournament size of 1 would be equivalent to selecting individuals at random and a tournament size equal to the population size would be equivalent to selecting the best individual at any given point. The selective pressure of tournament selection can be adjusted by means of the tournament size parameter, which makes it a more flexible selective procedure than fitness-proportional selection [4]. Therefore, the tournament selection works by creating a tight selective pressure on a small local group of individuals. Neither of these two selection procedures is better than the other for every problem. There are also a whole slew of other selection procedures that may or may not be based on either of these.

Because we only care about whether one individual is better than another, to save processing, we only need to consider standardized fitness for this selection procedure. Recall that better individuals have lower standard fitnesses.

3 Conclusion

In conclusion we would like to mention that GP has some evident superiority in comparison with the other approaches such statistics, neural networks and the other techniques, though it is not an ideal approach to solve the problem. In theory it could be successfully used simultaneously with the other methods in some disputable issues.

The main result obtained in the experiment is that the aforesaid search technique solves the problem in common cases. In addition the accuracy of classification on both testing and training sets was increased, although the results did deviate sometimes. Furthermore the results were false positive or negative at times. The system recognized the given symbols as a rule except some complicated cases. For example, the recognition of the digit '0' and letter 'O'; or the recognition of the digit '1' and letter 'l'. Another interesting fact is that the good solutions were commonly defended from discarding. Therefore we expect that the better results will be obtained after performing some modifications.

Although the research is made, its subject could be extended. There are some problems and aspects that were not consider in the current paper. First of all it will be great to make a deskewing and font normalization of the characters before their recognition. The second main task in perspective is to design and develop the recognition system (programming complex or toolbox). Once it is done it will be a big improvement to recognize not only the typewritten symbols but also handwritten characters. It will also be interesting to try to use in practice the approach given in this paper simultaneously with the other techniques such as the neural networks and the other techniques.

Bibliography

[1] Andre D., *A fast one pass raster-scan method for boundary extraction in binary images.* Canon Research Center Technical Report, Palo Alto, California, 1993.

[2] Andre D., *Learning and upgrading rules for an OCR system using Genetic Programming.* Stanford University and Canon Research Center of America, Stanford, 1994.

[3] Breunig M. M., *Location independent pattern recognition using Genetic Programming.* In Koza, J. R., editor, Genetic Algorithms and Genetic Programming at Stanford, Stanford Bookstore, Stanford University, 1995.

[4] Koza J. R., *Genetic Programming: on the programming of computers by means of natural selection.* MIT Press, Cambridge 1994.

[5] Krawiec K., *Genetic Programming using partial order of solutions for pattern recognition tasks.* Proceedings of II National Conference 'Computer Recognition Systems' KOSYR'2001, pp. 427–433.

[6] Poli R., *Genetic Programming for image analysis.* School of Computer Science, The University of Birmingham, 1996

Algebraic Solutions to Scheduling Problems in Project Management

Dr. Nikolai K. Krivulin

Faculty of Mathematics and Mechanics, St. Petersburg State University

Universitetsky Ave., 28, St. Petersburg, 198504, Russia

E-mail: {nkk@math.spbu.ru}

Abstract

We offer a computational approach to schedule development based on models and methods of idempotent algebra. The approach allows one to represent different types of precedence relationships among activities as linear vector equations written in terms of an idempotent semiring. As a result, many issues in project scheduling can be reduced to solving computational problems in the idempotent algebra setting, including linear equations and the eigenvalue-eigenvector problem. We give solutions to the problems in a compact vector form that provides a basis for the development of efficient computation algorithms and related software applications, including those intended for parallel implementation.

1 Introduction

The problem of scheduling a large-scale set of activities is a key issue in project management [1, 2]. There is a variety of project-scheduling techniques developed to deal with different aspects of the problem, ranging from the classical Critical Path Method and the Program Evaluation and Review Technique marked the beginning of the active research in the area in 1950s, to more recent methods of idempotent algebra [4, 5, 6, 7, 10].

We offer a new computational approach [10] to schedule development based on implementation of models and methods of idempotent algebra. The approach allows one to represent different types of precedence relationships among activities as linear vector equations written in terms of an idempotent semiring. As a result, many issues in project scheduling can be reduced to solving computational problems in the idempotent algebra setting, including linear equations and the eigenvalue-eigenvector problem. We give solutions to the problems in a compact vector form that provides

a basis for the development of efficient computation algorithms and related software applications, including those intended for parallel implementation.

The rest of the paper is organized as follows. We start with algebraic definitions and notation, and then outline basic results that underlie subsequent applications of idempotent algebra. Furthermore, examples of actual problems in project scheduling are considered. We show how to formulate the problems in an algebraic setting, and give related algebraic solutions. Finally, concluding remarks are given to summarize the results.

2 Definitions and Notation

We start with a brief overview of basic concepts, terms and symbols in idempotent algebra. Further details can be found in [4, 5, 6, 7, 10].

2.1 Idempotent Semiring

Consider a set $\mathbb{X}$ that is equipped with two operations $\oplus$ and $\otimes$ referred to as addition and multiplication, and has neutral elements $\mathbb{0}$ and $\mathbb{1}$ called zero and identity. We suppose that $\langle \mathbb{X}, \mathbb{0}, \mathbb{1}, \oplus, \otimes \rangle$ is a commutative semiring, where addition is idempotent and multiplication is invertible. Such a semiring is usually called idempotent semifield.

Let us define $\mathbb{X}_+ = \mathbb{X} \setminus \{\mathbb{0}\}$. Each $x \in \mathbb{X}_+$ is assumed to have its inverse x^{-1}. For any $x \in \mathbb{X}_+$ and integer $p > 0$, the power is defined in the ordinary way,

$$x^0 = \mathbb{1}, \quad x^p = x^{p-1} \otimes x = x \otimes x^{p-1}, \quad x^{-p} = (x^{-1})^p, \quad \mathbb{0}^p = \mathbb{0}.$$

In what follows, the multiplication sign $\otimes$ is omitted as is usual in conventional algebra. The notation of power is thought of as defined in terms of idempotent algebra. However, in the expressions that represent exponents, we use ordinary arithmetic operations.

Since the addition is idempotent, it induces a partial order $\leq$ on $\mathbb{X}$ according to the rule: $x \leq y$ if and only if $x \oplus y = y$. The relation symbols are understood below in the sense of this partial order. According to the order, it holds that has $x \geq \mathbb{0}$ for any $x \in \mathbb{X}$.

As a classical example of idempotent semirings (semifields), one can consider the semiring

$$\mathbb{R}_{\max,+} = \langle \mathbb{R} \cup \{-\infty\}, -\infty, 0, \max, + \rangle.$$

The semiring has the neutral elements $\mathbb{0} = -\infty$ and $\mathbb{1} = 0$. For each $x \in \mathbb{R}$, there exists its inverse x^{-1}, which is equal to $-x$ in ordinary arithmetics.

For any $x, y \in \mathbb{R}$, the power x^y is equivalent to the arithmetic product xy. The order induced by idempotent addition coincides with the natural linear order on $\mathbb{R}$.

The semiring $\mathbb{R}_{\max,+}$ provides the basis for the development of algebraic solutions to project scheduling problems in subsequent sections.

2.2 Matrix Algebra

Now consider matrices with elements in $\mathbb{X}$. The set of all matrices of size $m \times n$ is denoted by $\mathbb{X}^{m\times n}$.

The matrix with all entries equal to zero is the null matrix denoted by $\mathbb{0}$. A matrix is called regular if it has at least one nonzero element in every row.

For any scalar $x \in \mathbb{X}$ and matrices

$$A = (a_{ij}) \in \mathbb{X}^{m\times n}, \qquad B = (b_{ij}) \in \mathbb{X}^{m\times n}, \qquad C = (c_{ij}) \in \mathbb{X}^{n\times l}$$

matrix addition and multiplication, as well as multiplication by scalars are defined in the usual way with the expressions

$$\{A \oplus B\}_{ij} = a_{ij} \oplus b_{ij}, \qquad \{BC\}_{ij} = \bigoplus_{k=1}^{n} b_{ik}c_{kj}, \qquad \{xA\}_{ij} = xa_{ij}.$$

A square matrix is called diagonal if all its off-diagonal entries are zero, and triangular if its entries above (below) the diagonal are zero. The matrix $I = \mathrm{diag}(\mathbb{1}, \ldots, \mathbb{1})$ is referred to as identity matrix.

A matrix A is irreducible if and only if it cannot be put in a block triangular form by simultaneous permutations of rows and columns.

For any square matrix A and integer $p > 0$, the power is defined as usual,

$$A^0 = I, \qquad A^p = A^{p-1}A = AA^{p-1}.$$

For a square matrix $A = (a_{ij}) \in \mathbb{X}^{n\times n}$, its trace is given by

$$\mathrm{tr}\, A = \bigoplus_{i=1}^{n} a_{ii}.$$

Let $A = (a_{ij}) \in \mathbb{X}^{m\times n}$ be a regular matrix. The pseudo-inverse matrix of A is defined as $A^- = (a_{ij}^-) \in \mathbb{X}^{n\times m}$, where $a_{ij}^- = a_{ji}$ if $a_{ji} \neq \mathbb{0}$, and $a_{ij} = \mathbb{0}$ otherwise.

Finally, consider the set $\mathbb{X}^n$ of all column vectors with elements in $\mathbb{X}$. The vector with all elements equal to zero is called null vector and denoted by $\mathbb{0}$.

For any column vector $\boldsymbol{x} = (x_1, \ldots, x_n)^T \neq \mathbb{0}$, one can define a row vector $\boldsymbol{x}^- = (x_1^-, \ldots, x_n^-)$ with elements $x_i^- = x_i$ if $x_i \neq \mathbb{0}$, and $x_i = \mathbb{0}$ otherwise, $i = 1, \ldots, n$.

We define the distance between any two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{X}_+^n$ with a metric

$$\rho(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{y}^- \boldsymbol{x} \oplus \boldsymbol{x}^- \boldsymbol{y}.$$

When $\boldsymbol{y} = \boldsymbol{x}$ we have the minimum distance $\rho(\boldsymbol{x}, \boldsymbol{x}) = \mathbb{1}$.

In the semiring $\mathbb{R}_{\max,+}^n$, the metric takes the form

$$\rho(\boldsymbol{x}, \boldsymbol{y}) = \max_i |x_i - y_i|,$$

and thus coincides with the Chebyshev metric.

3 Preliminary Results

Now we outline some basic results from [8, 9, 10] that underlie subsequent applications of idempotent algebra to project scheduling.

3.1 The Equation $A\boldsymbol{x} = \boldsymbol{d}$

Suppose a matrix $A \in \mathbb{X}^{m \times n}$ and a vector $\boldsymbol{d} \in \mathbb{X}^m$ are given. Let $\boldsymbol{x} \in \mathbb{X}^n$ be an unknown vector. We examine the equation

$$A\boldsymbol{x} = \boldsymbol{d}, \tag{3.1}$$

and the inequality

$$A\boldsymbol{x} \leq \boldsymbol{d}, \tag{3.2}$$

A solution $\boldsymbol{x}_0$ to equation (3.1) or inequality (3.2) is called the maximum solution if $\boldsymbol{x}_0 \geq \boldsymbol{x}$ for all solutions $\boldsymbol{x}$ of (3.1) or (3.2).

We present a solution to (3.1) based on analysis of distance between vectors in $\boldsymbol{X}^n$. To simplify further formulae, we use the notation

$$\Delta = (A(\boldsymbol{d}^- A)^-)^- \boldsymbol{d}.$$

Lemma 1. *Suppose $A \in \mathbb{X}^{m \times n}$ is a regular matrix, and $\boldsymbol{d} \in \mathbb{X}_+^m$ is a vector without zero components. Then it holds that*

$$\min_{\boldsymbol{x} \in \mathbb{X}_+^n} \rho(A\boldsymbol{x}, \boldsymbol{d}) = \Delta^{1/2},$$

where the minimum is achieved at $\boldsymbol{x}_0 = \Delta^{1/2}(\boldsymbol{d}^- A)^-$.

Lemma 2. *Under the same conditions as in Lemma 1 it holds that*

$$\min_{A\boldsymbol{x}\le\boldsymbol{d}} \rho(A\boldsymbol{x},\boldsymbol{d}) = \min_{A\boldsymbol{x}\ge\boldsymbol{d}} \rho(A\boldsymbol{x},\boldsymbol{d}) = \Delta,$$

where the first minimum is achieved at $\boldsymbol{x}_1 = (\boldsymbol{d}^- A)^-$*, and the second at* $\boldsymbol{x}_2 = \Delta(\boldsymbol{d}^- A)^-$*.*

As a consequence of Lemma 1 and 2, we get the following result.

Theorem 1. *A solution of equation* (3.1) *exists if and only if* $\Delta = \mathbb{1}$*. If solvable, the equation has the maximum solution given by*

$$\boldsymbol{x} = (\boldsymbol{d}^- A)^-.$$

Suppose that $\Delta > \mathbb{1}$. In this case equation (3.1) has no solution. However, we can define a pseudo-solution to (3.1) as a solution of the equation

$$A\boldsymbol{x} = \Delta^{1/2} A(\boldsymbol{d}^- A)^-,$$

which is always exists and takes the form $\boldsymbol{x}_0 = \Delta^{1/2}(\boldsymbol{d}^- A)^-$. It follows from Lemma 1 that the pseudo-solution yields the minimum deviation between vectors $\boldsymbol{y} = A\boldsymbol{x}$ and the vector $\boldsymbol{d}$ in the sense of the metric ρ.

Consider the problem of finding two vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ that provide the minimum deviation between both sides of (3.1), while satisfying the respective inequalities $A\boldsymbol{x} \le \boldsymbol{d}$ and $A\boldsymbol{x} \ge \boldsymbol{d}$. As it is easy to see from Lemma 2, these vectors are given by

$$\boldsymbol{x}_1 = (\boldsymbol{d}^- A)^-, \qquad \boldsymbol{x}_2 = \Delta(\boldsymbol{d}^- A)^-.$$

The next statement is another consequence of the above results.

Lemma 3. *For any regular matrix* A *and vector* $\boldsymbol{d}$ *without zero components, the solution to* (3.2) *is given by the inequality*

$$\boldsymbol{x} \le (\boldsymbol{d}^- A)^-.$$

A solution to equation (3.1) with an arbitrary matrix A and a vector d is considered in [10]

3.2 The Equation $A\boldsymbol{x} \oplus \boldsymbol{b} = \boldsymbol{x}$

Suppose a matrix $A \in \mathbb{X}^{n \times n}$ and a vector $\boldsymbol{b} \in \mathbb{X}^n$ are given, and $\boldsymbol{x} \in \mathbb{X}^n$ is an unknown vector. Consider the equation

$$A\boldsymbol{x} \oplus \boldsymbol{b} = \boldsymbol{x}, \tag{3.3}$$

and the inequality

$$A\boldsymbol{x} \oplus \boldsymbol{b} \leq \boldsymbol{x}, \tag{3.4}$$

A solution to equation (3.3) is proposed based on application of a function $\mathrm{Tr}\, A$ that takes each square matrix to a scalar and plays the role of the determinant in conventional linear algebra. The function is given by

$$\mathrm{Tr}\, A = \bigoplus_{m=1}^{n} \mathrm{tr}\, A^m$$

and exploited to examine whether the equation has a unique solution, many solutions, or no solution.

For any $A \in \mathbb{X}^{n \times n}$, we define matrices A^+ and $A^\times$ as follows

$$A^+ = I \oplus A \oplus \cdots \oplus A^{n-1}, \qquad A^\times = AA^+ = A \oplus \cdots \oplus A^n.$$

Let $\boldsymbol{a}_i^+$ be column i of A^+, and $a_{ii}^\times$ be entry (i, i) of $A^\times$. Provided that $\mathrm{Tr}\, A = \mathbb{1}$, we define the matrix $A^* = (\boldsymbol{a}_i^*)$ with the columns

$$\boldsymbol{a}_i^* = \begin{cases} \boldsymbol{a}_i^+, & \text{if } a_{ii}^\times = \mathbb{1}, \\ \mathbb{0}, & \text{otherwise.} \end{cases}$$

If $\mathrm{Tr}\, A \neq \mathbb{1}$, then we take $A^* = \mathbb{0}$.

The solution to equation (3.3) is given by the following result.

Theorem 2. *Let $\boldsymbol{x}$ be the solution of equation* (3.3) *with an irreducible matrix A. Then the following statements hold:*

1) *if $\mathrm{Tr}\, A < \mathbb{1}$, then there exists a unique solution $\boldsymbol{x} = A^+\boldsymbol{b}$;*
2) *if $\mathrm{Tr}\, A = \mathbb{1}$, then $\boldsymbol{x} = A^+\boldsymbol{b} \oplus A^*\boldsymbol{v}$ for all $\boldsymbol{v} \in \mathbb{X}^n$;*
3) *if $\mathrm{Tr}\, A > \mathbb{1}$, then with the condition $\boldsymbol{b} = \mathbb{0}$, there exists only the solution $\boldsymbol{x} = \mathbb{0}$, whereas with $\boldsymbol{b} \neq \mathbb{0}$, there is no solution.*

Lemma 4. *Let $\boldsymbol{x}$ be the solution of inequality* (3.4) *with an irreducible matrix A. Then the following statements hold:*

1) *if $\mathrm{Tr}\, A \leq \mathbb{1}$, then $\boldsymbol{x} = A^+(\boldsymbol{b} \oplus \boldsymbol{v})$ for all $\boldsymbol{v} \in \mathbb{X}^n$;*
2) *if $\mathrm{Tr}\, A > \mathbb{1}$, then with the condition $\boldsymbol{b} = \mathbb{0}$, there exists only the solution $\boldsymbol{x} = \mathbb{0}$, whereas with $\boldsymbol{b} \neq \mathbb{0}$, there is no solution.*

Related results for the case of arbitrary matrices can be found in [8, 10].

3.3 Eigenvalues and Eigenvectors

A scalar λ is an eigenvector of a square matrix $A \in \mathbb{X}^{n\times n}$ if there is a vector $\boldsymbol{x} \in \mathbb{X}^n \setminus \{\mathbb{0}\}$ such that

$$A\boldsymbol{x} = \lambda\boldsymbol{x}.$$

The maximum eigenvalue is called spectral radius of A and given by

$$\varrho = \bigoplus_{m=1}^{n} \mathrm{tr}^{1/m}(A^m).$$

The eigenvector corresponding to ϱ takes the form

$$\boldsymbol{x} = A_\varrho^* \boldsymbol{v},$$

where $A_\varrho = \varrho^{-1}A$, and $\boldsymbol{v}$ is any vector.

Lemma 5. *For any irreducible matrix A with the spectral radius ϱ, it holds that*

$$\min_{\boldsymbol{x}\in\mathbb{X}_+^n} \rho(A\boldsymbol{x}, \boldsymbol{x}) = \varrho \oplus \varrho^{-1},$$

where the minimum is achieved at any eigenvector $\boldsymbol{x}$ corresponding to ϱ.

The case of arbitrary matrices is considered in [9, 10].

4 Applications to Project Scheduling

In this section we show how to apply the algebraic results presented above to solve scheduling problems under various constraints (for further details on the schedule development in project management see, e.g., [1, 2]).

As the underlying idempotent semiring, we use $\mathbb{R}_{\max,+}$ in all examples under discussion.

4.1 Precedence Relations of the Start-to-Finish Type

Consider a project that involves n activities. Activity dependencies are assumed the form of Start-to-Finish relations that do not allow an activity to complete until some time after initiation of other activities. The scheduling problem of interest is to find initiation time for all activities subject to given constraints on their completion time.

For each activity $i = 1, \ldots, n$, denote by x_i its initiation time, and by y_i its completion time. Let d_i be a due date, and a_{ij} a minimum possible time lag between initiation of activity $j = 1, \ldots, n$ and completion of i.

Given a_{ij} and d_i, the completion time of activity i must satisfy the relations

$$y_i = d_i, \qquad x_j + a_{ij} \le y_i, \quad j = 1, \ldots, n,$$

where if a_{ij} is not actually given for some j, it is assumed to be $\mathbb{0} = -\infty$.

The relations can be combined into one equation in the initiation times

$$\max(x_1 + a_{i1}, \ldots, x_n + a_{in}) = d_i.$$

By replacing the ordinary operations with those in $\mathbb{R}_{\max,+}$ in all equations, we get

$$a_{i1} \otimes x_1 \oplus \cdots \oplus a_{in} \otimes x_n = d_i, \quad i = 1, \ldots, n.$$

For simplicity, we drop the multiplication symbol $\otimes$, and write

$$a_{i1} x_1 \oplus \cdots \oplus a_{in} x_n = d_i, \quad i = 1, \ldots, n.$$

With the notation

$$A = \begin{pmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \ldots & a_{nn} \end{pmatrix}, \qquad \boldsymbol{d} = \begin{pmatrix} d_1 \\ \vdots \\ d_n \end{pmatrix}, \qquad \boldsymbol{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix},$$

the scheduling problem under the start-to-finish constraints leads us to solution of the equation

$$A\boldsymbol{x} = \boldsymbol{d}.$$

Consider $\Delta = (A(\boldsymbol{d}^- A)^-)^- \boldsymbol{d}$. According to Theorem 1, provided that the condition $\Delta = \mathbb{1} = 0$ is satisfied, the equation has a unique solution $\boldsymbol{x} = (\boldsymbol{d}^- A)^-$.

If it appears that $\Delta > 0$, then one can compute approximate solutions to the equation

$$\boldsymbol{x}_0 = \Delta^{1/2} (\boldsymbol{d}^- A)^-, \qquad \boldsymbol{x}_1 = (\boldsymbol{d}^- A)^-, \qquad \boldsymbol{x}_2 = \Delta (\boldsymbol{d}^- A)^-.$$

The completion times corresponding to these solution are given by

$$\boldsymbol{y}_0 = A\boldsymbol{x}_0, \qquad \boldsymbol{y}_1 = A\boldsymbol{x}_1 \le \boldsymbol{d}, \qquad \boldsymbol{y}_2 = A\boldsymbol{x}_2 \ge \boldsymbol{d},$$

and have their deviation from the due dates bounded with

$$\rho(\boldsymbol{y}_0, \boldsymbol{d}) = \Delta^{1/2}, \qquad \rho(\boldsymbol{y}_1, \boldsymbol{d}) = \rho(\boldsymbol{y}_2, \boldsymbol{d}) = \Delta.$$

Suppose that the due date constraints may be adjusted to some extent. As a new vector of due dates, it is natural to take a vector $\boldsymbol{d}'$ such that

$\boldsymbol{y}_1 \le \boldsymbol{d}' \le \boldsymbol{y}_2$. In this case, deviation of the new due dates from the original ones does not exceed Δ. The minimum deviation which is equal to $\Delta^{1/2}$ is achieved at $\boldsymbol{d}' = \boldsymbol{y}_0$.

As an example, consider a project with a constraint matrix and two due date vectors given by

$$A = \begin{pmatrix} 8 & 10 & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & 5 & 4 & 8 \\ 6 & 12 & 11 & 7 \\ \mathbb{0} & \mathbb{0} & \mathbb{0} & 12 \end{pmatrix}, \qquad \boldsymbol{d}_1 = \begin{pmatrix} 14 \\ 11 \\ 16 \\ 15 \end{pmatrix}, \qquad \boldsymbol{d}_2 = \begin{pmatrix} 15 \\ 15 \\ 15 \\ 15 \end{pmatrix}.$$

Fig. 1 demonstrates a network representation of the project.

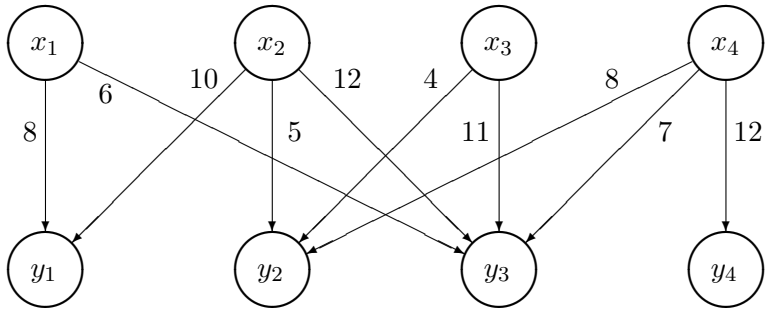


Figure 1: An activity network with Start-to-Finish precedence relations

First we examine the equation $A\boldsymbol{x} = \boldsymbol{d}_1$. Simple calculation gives $\Delta_1 = (A(\boldsymbol{d}_1^- A)^-)^- \boldsymbol{d}_1 = 0$. Therefore, the equation has a unique solution

$$\boldsymbol{x} = (\boldsymbol{d}_1^- A)^- = (6, 4, 5, 3)^T.$$

Consider the equation $A\boldsymbol{x} = \boldsymbol{d}_2$. Since $\Delta_2 = (A(\boldsymbol{d}_2^- A)^-)^- \boldsymbol{d}_2 = 4 > 0$, the equation does not have a solution. Evaluation of approximate solutions gives

$$\begin{aligned} \boldsymbol{x}_0 &= \Delta_2^{1/2}(\boldsymbol{d}_2^- A)^- = (9, 5, 6, 5)^T, & \boldsymbol{y}_0 &= A\boldsymbol{x}_0 = (17, 13, 17, 17)^T, \\ \boldsymbol{x}_1 &= (\boldsymbol{d}_2^- A)^- = (7, 3, 4, 3)^T, & \boldsymbol{y}_1 &= A\boldsymbol{x}_1 = (15, 11, 15, 15)^T, \\ \boldsymbol{x}_2 &= \Delta_2(\boldsymbol{d}_2^- A)^- = (11, 7, 8, 7)^T, & \boldsymbol{y}_2 &= A\boldsymbol{x}_2 = (19, 15, 19, 19)^T. \end{aligned}$$

4.2 Precedence Relations of the Start-to-Start Type

Suppose there is a project consisting of n activities and operating under Start-to-Start precedence constraints that determine the minimum (maximum) allowed time intervals between initiation of activities.

For each activity $i = 1, \ldots, n$, let b_i be an early possible initiation time, and let a_{ij} be a minimum possible time lag between initiation of activity $j = 1, \ldots, n$ and initiation of i. The problem is to find the earliest initiation time x_i for every activity i so as to provide for the relations

$$b_i \le x_i, \qquad a_{ij} + x_j \le x_i, \quad j = 1, \ldots, n,$$

which can be replaced with one equation

$$\max(\max(x_1 + a_{i1}, \ldots, x_n + a_{in}), b_i) = x_i.$$

Representation in terms of $\mathbb{R}_{\max,+}$, gives the scalar equations

$$a_{i1}x_1 \oplus \cdots \oplus a_{in}x_n \oplus b_i = x_i, \quad i = 1, \ldots, n.$$

With the notation $A = (a_{ij})$, $\boldsymbol{b} = (b_1, \ldots, b_n)^T$, $\boldsymbol{x} = (x_1, \ldots, x_n)^T$ we arrive at a problem that is to solve the equation

$$A\boldsymbol{x} \oplus \boldsymbol{b} = \boldsymbol{x}.$$

For simplicity, assume the matrix A to be irreducible. It follows from Theorem 2 that if $\operatorname{Tr} A \le \mathbb{1} = 0$ then the equation has a nontrivial solution given by $\boldsymbol{x} = A^+\boldsymbol{b} \oplus A^*\boldsymbol{v}$ for any vector $\boldsymbol{v}$.

Consider a project with start-to-start relations and examine two cases, with and without early initiation time constraints imposed. Let us define a matrix and two vectors as follows

$$A = \begin{pmatrix} 0 & -2 & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & 0 & 3 & -1 \\ -1 & \mathbb{0} & 0 & -4 \\ 2 & \mathbb{0} & \mathbb{0} & 0 \end{pmatrix}, \qquad \boldsymbol{b}_1 = \mathbb{0}, \qquad \boldsymbol{b}_2 = \begin{pmatrix} 1 \\ 1 \\ 2 \\ 1 \end{pmatrix}.$$

A graph representation of the project is depicted in Fig. 2.

Let us first calculate the initiation time of activities in the project when $\boldsymbol{b} = \boldsymbol{b}_1 = \mathbb{0}$ (that is, without early initiation time constraints given). Under this assumption, the equation takes the form $A\boldsymbol{x} = \boldsymbol{x}$.

As it is easy to see, the matrix A is irreducible and $\operatorname{Tr} A = 0$. Therefore, the equation has a solution.

Simple algebra gives

$$A^+ = A^\times = \begin{pmatrix} 0 & -2 & 1 & -3 \\ 2 & 0 & 3 & -1 \\ -1 & -3 & 0 & -4 \\ 2 & 0 & 3 & 0 \end{pmatrix}, \qquad A^* = \begin{pmatrix} -2 & -3 \\ 0 & -1 \\ -3 & -4 \\ 0 & 0 \end{pmatrix}.$$

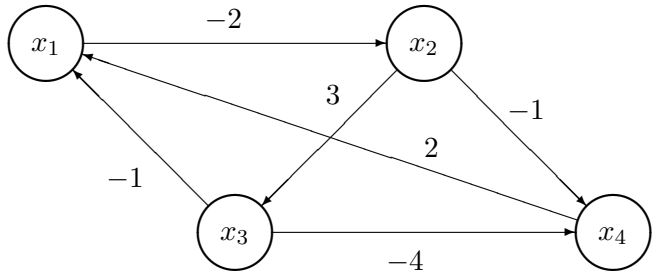


Figure 2: An activity network with Start-to-Start precedence relations

Note that, since A^+ and $A^\times$ coincide, one should define $A^* = A^+$. However, considering that the first three columns are proportional, we take only one of them.

The solution to the equation is given by

$$\boldsymbol{x} = A^*\boldsymbol{v} = \begin{pmatrix} -2 & -3 \\ 0 & -1 \\ -3 & -4 \\ 0 & 0 \end{pmatrix} \boldsymbol{v}, \quad \boldsymbol{v} \in \mathbb{R}^2_{\max,+}.$$

Consider the case with the vector $\boldsymbol{b}_2$ and the equation taking the form $A\boldsymbol{x} \oplus \boldsymbol{b}_2 = \boldsymbol{x}$. Now we have

$$A^+\boldsymbol{b}_2 = \begin{pmatrix} 3 \\ 5 \\ 2 \\ 5 \end{pmatrix}, \qquad \boldsymbol{x} = \begin{pmatrix} 3 \\ 5 \\ 2 \\ 5 \end{pmatrix} \oplus \begin{pmatrix} -2 & -3 \\ 0 & -1 \\ -3 & -4 \\ 0 & 0 \end{pmatrix} \boldsymbol{v}, \quad \boldsymbol{v} \in \mathbb{R}^2_{\max,+}.$$

4.3 Mixed Precedence Relations

Consider a project that has both Start-to-Finish and Start-to-Start constraints. Let A_1 be a given Start-to-Finish constraint matrix, $\boldsymbol{d}$ a vector of due dates, and $\boldsymbol{x}$ an unknown vector of activity initiation time. To meet the constraints, the vector $\boldsymbol{x}$ must satisfy the inequality

$$A_1\boldsymbol{x} \leq \boldsymbol{d}.$$

Furthermore, there are also Start-to-Start constraints defined by a constraint matrix A_2. This leads to the equation in $\boldsymbol{x}$

$$A_2\boldsymbol{x} = \boldsymbol{x}.$$

Suppose that the equation has a solution $\boldsymbol{x} = A_2^* \boldsymbol{v}$. Substitution of the solution into the above inequality gives

$$A_1 A_2^* \boldsymbol{v} \leq \boldsymbol{d}.$$

Since the maximum solution to the last inequality is $\boldsymbol{v} = (\boldsymbol{d}^- A_1 A_2^*)^-$, the solution to the whole problem is written in the form

$$\boldsymbol{x} = A_2^* (\boldsymbol{d}^- A_1 A_2^*)^-.$$

As an illustration, we evaluate the solution to the problem under the conditions

$$A_1 = \begin{pmatrix} 8 & 10 & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & 5 & 4 & 8 \\ 6 & 12 & 11 & 7 \\ \mathbb{0} & \mathbb{0} & \mathbb{0} & 12 \end{pmatrix}, \qquad A_2 = \begin{pmatrix} 0 & -2 & \mathbb{0} & \mathbb{0} \\ \mathbb{0} & 0 & 3 & -1 \\ -1 & \mathbb{0} & 0 & -4 \\ 2 & \mathbb{0} & \mathbb{0} & 0 \end{pmatrix},$$

and

$$\boldsymbol{d} = (13, 11, 15, 15)^T.$$

By using results of previous examples, we successively get

$$A_1 A_2^* = \begin{pmatrix} 10 & 9 \\ 8 & 8 \\ 12 & 11 \\ 12 & 12 \end{pmatrix}, \qquad (\boldsymbol{d}^- A_1 A_2^*)^- = \begin{pmatrix} 3 \\ 3 \end{pmatrix}.$$

Finally, we have

$$\boldsymbol{x} = A_2^* (\boldsymbol{d}^- A_1 A_2^*)^- = (1, 3, 0, 3)^T.$$

4.4 Minimization of the Maximum Flow Time

Assume that a project operates under Start-to-Finish constraints. For each activity in the project, consider the time interval between its initiation and completion, which is usually referred to as the flow time and also as turnaround time or processing time.

In practice, one can be interested in constructing a schedule that minimizes the maximum flow time over all activities in the project. With $\boldsymbol{x}$ standing for a vector of initiation time, and A for a constraint matrix, we arrive at a problem formulated in terms of $\mathbb{R}_{\max,+}$ to find

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \rho(A\boldsymbol{x}, \boldsymbol{x}).$$

It follows from Lemma 5 that the above minimum is equal to the $\varrho \oplus \varrho^{-1}$, where ϱ is the spectral radius of A, and it is achieved at the vector given by $\boldsymbol{x} = A_{\varrho}^{*}\boldsymbol{v}$ for any vector $\boldsymbol{v}$.

Suppose a vector $\boldsymbol{d}$ is given to represent activity due dates. Consider a problem of evaluating the latest initiation time for all activities so as to provide both the due date constraints and the minimum flow time condition.

By combining the due date constraints represented in the form

$$A\boldsymbol{x} \leq \boldsymbol{d}$$

with the solution of the minimization problem, we have the inequality

$$AA_{\rho}^{*}\boldsymbol{v} \leq \boldsymbol{d}.$$

With the maximum solution to the inequality $\boldsymbol{v} = (\boldsymbol{d}^{-}AA_{\rho}^{*})^{-}$, we get the solution of the whole problem

$$\boldsymbol{x} = A_{\rho}^{*}(\boldsymbol{d}^{-}AA_{\rho}^{*})^{-}.$$

Let us evaluate the solution with the constraint matrix and due date vector defined as

$$A = \begin{pmatrix} 2 & 4 & 4 \\ 2 & 3 & 5 \\ 3 & 2 & 3 \end{pmatrix}, \qquad \boldsymbol{d} = \begin{pmatrix} 9 \\ 8 \\ 9 \end{pmatrix}.$$

First we get $\rho = 4$, and define the matrix

$$A_{\rho} = \begin{pmatrix} -2 & 0 & 0 \\ -2 & -1 & 1 \\ -1 & -2 & -1 \end{pmatrix}.$$

Furthermore, we have the matrices

$$A_{\rho}^{+} = A_{\rho}^{*} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ -1 & -1 & 0 \end{pmatrix}, \qquad A_{\rho}^{*} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}.$$

Finally, we arrive at the solution

$$\boldsymbol{x} = A_{\rho}^{*}(\boldsymbol{d}^{-}AA_{\rho}^{*})^{-} = \begin{pmatrix} 4 \\ 4 \\ 3 \end{pmatrix}.$$

5 Conclusion

We have presented an approach that exploits idempotent algebra to solve computational problems in project scheduling. It is shown how to reformulate the problems in an algebraic setting, and then find related solutions based on appropriate results in the idempotent algebra theory. The solutions are given in a compact vector form that provides a basis for the development of efficient computation algorithms and software applications, including those intended for parallel implementation.

6 Acknowledgments

The work was supported in part by the Russian Foundation for Basic Research Grant #09-01-00808.

Bibliography

[1] *A Guide to the Project Management Body of Knowledge: PMBOK Guide.* Newtown Square, PA: Project Management Institute, 2008. 459 p.

[2] K. Neumann, C. Schwindt, J. Zimmermann, *Project Scheduling with Time Windows and Scarce Resources.* Berlin, Springer, 2003. 385 p.

[3] R. Cuninghame-Green, *Minimax Algebra.* Berlin, Springer, 1979. 258 p. (Lecture Notes in Economics and Mathematical Systems, vol. 166)

[4] F. Baccelli, G. Cohen, G. J. Olsder, J.-P. Quadrat, *Synchronization and Linearity: An Algebra for Discrete Event Systems.* Chichester, Wiley, 1993. 514 p.

[5] V. N. Kolokoltsov, V. P. Maslov, *Idempotent Analysis and Its Applications.* N. Y., Springer, 1997. 324 p.

[6] J. S. Golan, *Semirings and Affine Equations Over Them: Theory and Applications.* N. Y., Springer, 2003. 256 p.

[7] B. Heidergott, G. J. Olsder, J. van der Woude, *Max-Plus at Work: Modeling and Analysis of Synchronized Systems.* Princeton, Princeton Univ. Press, 2005. 226 p.

[8] N. K. Krivulin, *Solution of Generalized Linear Vector Equations in Idempotent Algebra* Vestnik St. Petersburg Univ. Math., 39 (1), 2006. pp. 16–26.

[9] N. K. Krivulin, *Eigenvalues and Eigenvectors of Matrices in Idempotent Algebra* Vestnik St. Petersburg Univ. Math., 39 (2), 2006. pp. 72–83.

[10] N. K. Krivulin, *Idempotent Algebra Methods for Problems in Modeling and Analysis of Complex Systems.* St. Petersburg, St. Petersburg Univ. Press, 2009. 256 p. (in Russian)

Moment properties and long-range dependence of queueing processes

Prof. Evsei V. Morozov, Alexander S. Rumyantsev

Institute of Applied Mathematical Research,
Karelian Research Centre, RAS

E-mail: {emorozov, ar0}@krc.karelia.ru

Abstract

The aim of this study is to empirically extend known results on long range dependence for a separated system to 2-station tandem system. More exactly, we study connection of a long range dependence effect at the second station in a tandem network with moment properties of the input and service times in both stations. Simulation results are presented, and some related difficulties are discussed.

1 Introduction

The main motivation of this study is to empirically verify results obtained in the paper [4]. This verification for a separate system has been presented in [2].

We briefly recall related definitions and results. Consider a single-server queueing system GI/G/1 with a renewal input with arrival epochs $\{t_i\}$ and the i.i.d. interarrival times $\{T_i = t_{i+1} - t_i\}$ with distribution $A(x) = P(T \leq x)$ (T denotes generic interarrival). It is assumed that service times $\{S_i\}$ are i.i.d with distribution $B(x) = P(S \leq x)$. Denote by W_i the waiting time of customer i in queue. Recall famous Lindley's recursion which defines the sequence $\{W_i\}$:

$$W_{i+1} = (W_i + S_i - T_i)^+,$$

where $(\cdot)^+ = \max(0, \cdot)$. Let the stability condition holds, i.e. $ES/ET < 1$, or, equivalently, $E(S - T) < 0$. Then the sequence $\{W_i\}$ has a weak limit $W_i \Rightarrow W$, $i \to \infty$. Moreover, the following stochastic equality connects this stationary limit and supremum of the associated random walk with negative drift:

$$W = \sup_{i \geqslant 1}(S_i - T_i).$$

The stationary waiting time (or *delay*) W is widely used as a QoS parameter. In particular, it is important to know the properties of the delay for delay-sensitive systems, such as real-time voice and video traffic. We give the following well known result from [4] defining *long range dependence* of the sequence of delays in a GI/G/1 system. More exactly, provided that $ET < \infty, ES^3 < \infty$ and $ES^4 = \infty$,

$$\sum_{i=1}^{\infty} corr(W_0, W_i) = \infty. \tag{1.1}$$

In practice the result (1.1) means that the waiting time process stays above or below it's mean value unexpectedly long. It makes difficult the use of sample mean based estimator to estimate required parameter with a given accuracy in reasonable simulation time [9]. An extended discussion of this topic is in the work [8]

It is obvious that $t_i + W_i + S_i$ is the departure instant of customer i. Hence, these instants define an inter-departure process,

$$\begin{aligned} D_i &= t_{i+1} + W_{i+1} + S_{i+1} - t_i - W_i - S_i \\ &= S_{i+1} + (T_i - W_i - S_i) + W_{i+1} = S_{i+1} + (T_i - W_i - S_i)^+, \, i \geq 1. \end{aligned}$$

Consider a two-station tandem, where after being served in the first single-server queue, the customer (task) enters the second system. Thus, the output from first system is an input to the second one. For node j, denote by $\{T_i^{(j)}\}$ interarrival times, by $\{S_i^{(j)}\}$ service times and by $\{W_i^{(j)}\}$ waiting times, $j = 1, 2$, and note that $T_i^{(2)} = D_i^{(1)}$. It is interesting to estimate the impact of characteristics of the first station on the delay at the second one. The main difference from first station, is that the inter-arrival times for second station form not a renewal but rather a regenerative process. Results on the output process can be found in [3].

The stability condition for the whole network is [12],

$$ET^{(1)} > \max(ES^{(1)}, ES^{(2)}),$$

while condition

$$\max(ET^{(1)}, ES^{(1)}) > ES^{(2)},$$

implies stability of the second station solely, $W_i^{(2)} \Rightarrow W^{(2)}$ (with proper limit $W^{(2)}$), leaving a possibility of instability of first station, $W^{(1)} \to \infty$ (in probability of with probability 1).

In a stable system, the knowledge of moment properties of delay may be useful for instance, to approximate the tail of the delay via Chebyshev's

inequality. A well-known (for separated station) result that the finiteness of $r+1$-th moment of service time implies finiteness of the r-th moment of waiting time is extended to a tandem network in [12]. More exactly, for a two-station case, if $ES^{(1)} > ES^{(2)}$, then (similar to one station case) sufficient stability condition holds:

$$E\left(S^{(2)}\right)^{r+1} < \infty \quad \Rightarrow \quad E\left(W^{(2)}\right)^{r} < \infty.$$

However, if $ES^{(1)} \leqslant ES^{(2)}$, then an additional condition is placed on moments of service times at the first station:

$$E\left(S^{(j)}\right)^{r+1} < \infty \quad \Rightarrow \quad E\left(W^{(2)}\right)^{r} < \infty, \quad j = 1, 2.$$

It turns out to be that the latter assumption is not only technical one caused by the method of the proof (as it has been conjectured in [12]), but as has been shown in [10], violation of the assumption may lead to *infinite mean stationary delay* at the second node.

Even more surprising dependence of moment properties at a given station on the properties of other stations in tandem-like networks is found in [7] for the so-called *heavy-tailed* case. First recall that a random variable (r.v.) X with distribution F is called subexponential if asymptotic equivalence holds $P(X_1 + X_2 > x) \sim 2(1 - F(x)) := 2\bar{F}(x)$, where $X_{1,2}$ are i.i.d. copies of X. A particular case is Pareto r.v. with tail distribution (for $x \geqslant x_0 > 0$)

$$\overline{F}(x) = x^{-\alpha}, \quad \alpha > 0. \tag{1.2}$$

If $\overline{B}(x)$ is the tail distribution of service time S, then an integrated tail distribution (of a stationary remaining service time S_e) is defined as

$$\overline{B}_e(x) := P(S_e > x) = \frac{1}{ES} \int_x^\infty \overline{B}(x)\, dx, \ x \geq 0.$$

When both $\overline{B}(x)$ and $\overline{B}_e(x)$ are subexponential, then we call that distribution B belongs to a useful subclass $\mathcal{S}^*$.

The crucial result of [7] (for two-station tandem) is as follows. Assume stability, that is $\rho_i := ES^{(i)}/ET^{(1)} < 1$ for $i = 1, 2$, and let the service time distribution at the second station belong to $\mathcal{S}^*$. Also assume that $P(S^{(1)} > x) = o(P(S^{(2)} > x))$ and that service time at the first station also belongs to $\mathcal{S}^*$ or is *light-tailed* [7]). Then

$$P(W^{(2)} > x) \sim \frac{\rho_2}{1 - \rho_2} P(S_e^{(2)} > x).$$

In other words, if service time at first station has lighter tail then the tail of delay asymptotically behaves like in a single station, and previous station does not matter.

Note that some of given above results hold for some more general networks.

2 Long range dependence in tandem

In this section we discuss some difficulties which arise when we try to empirically verify the theoretical results mentioned in the previous section.

2.1 Pareto tail modeling

To simulate the i.i.d r.v. $X_1, \dots, X_n$ with a given distribution F, we sample i.i.d. pseudo-random numbers $U_1, \dots, U_n$ and then use inverse transform. More exactly, we use the inverse function $F^{-1}(U_i)$ to get the sample values X_i:

$$X_i = \overline{F}^{-1}(U_i),\ i = 1, \dots, n.$$

(It ie easy to check that obtained r.v. indeed have distribution F.) In particular, for Pareto (tail) distribution (1.2),

$$X_i = U_i^{-1/\alpha},\ i = 1, \dots, n.$$

The problem which arises in practice is that the values of U_i have limited accuracy, say, $U_i \geqslant 10^{-\beta}$ for some $\beta > 1$. Then the maximum value $x_{\max}$ obtained by inverse transform sampling is

$$x_{\max} \leqslant 10^{\beta/\alpha}.$$

(Note that in this case the sample size has to be approximately 10^{β}.) Thus, instead of sampling from Pareto distribution we in fact obtain *truncated Pareto* distribution [5], that is

$$\overline{F}(x) = \frac{(x_0 x_{\max})^{\alpha}}{x_{\max}^{\alpha} - x_0^{\alpha}}(x^{-\alpha} - x_{\max}^{-\alpha}), \quad x_0 \leqslant x \leqslant x_{\max} < \infty,$$

with $\overline{F}(x) = 0$ for $x \geq x_{\max}$ and $\overline{F}(x) = 1$ for $x \leq x_0$. (We mention an asymptotic level-q test in [1] to verify the hypothesis about the truncated Pareto distribution.) We recall that for classical Pareto (1.2) $EX^n = \infty$ for $n \geqslant \alpha$. However, in our case,

$$EX^n = \int_{x_0}^{x_{\max}} x^n \frac{\alpha(x_0 x_{\max})^{\alpha}}{x_{\max}^{\alpha} - x_0^{\alpha}} x^{-\alpha-1}\, dx = \frac{\alpha(x_0 x_{\max})^{\alpha}}{n - \alpha} \frac{x_{\max}^{n-\alpha} - x_0^{n-\alpha}}{x_{\max}^{\alpha} - x_0^{\alpha}}.$$

Hence, if $x_{\max} = 10^{\beta/\alpha}$ and $x_0 = 1$ (for standard Pareto),

$$EX^n = \frac{\alpha 10^{\beta}}{n-\alpha} \frac{10^{(n-\alpha)\beta/\alpha} - 1}{10^{\beta} - 1} \approx \frac{\alpha}{n-\alpha} 10^{\beta\cdot(n-\alpha)/\alpha}. \tag{2.1}$$

For instance, let $\beta = 16$ (the double precision accuracy of C language defined variable), $\alpha = 3.5$ and $n = 4$. Substitution this in (2.1) implies

$$EX^n \approx 7 \cdot 10^{2.2857}.$$

This value is far from being infinite. Moreover, for our case, to have EX^n at least an order of 10^{10}, one needs $\beta \approx 70$, see from (2.1). (The so-called *long arithmetics* provides an arbitrary order of accuracy but sample size 10^{70} hard to get in a reasonable simulation time.)

Nevertheless, note that if $n > \alpha$ and α approaches zero, then $(n-\alpha)/\alpha$ increases. Thus, for $0 < \alpha < 2$ one may get reasonable results for the value of EX^n.

2.2 Numerical results

The experiments were carried out on a High-Performance cluster [6]. The autocorrelation coefficients were calculated by formulae

$$\hat{\rho}_i = \frac{M\sum_{j=1}^{M} W_0(j)W_i(j) - \sum_{j=1}^{M} W_0(j)\sum_{j=1}^{M} W_i(j)}{M\sum_{j=1}^{M} (W_0(j))^2 - \left(\sum_{j=1}^{M} W_0(j)\right)^2},$$

where $W_i(j)$ corresponds to the waiting time for task i in the independent run j. Note that independent runs are preferable than a single long run in the presence of long-range dependence [11].

The problem discussed in the previous subsection means that if in distribution (1.2) $\alpha < 4$, we in fact obtain empirically finite forth moment implying convergence of autocorrelation series. A possibility to obtain (quasi)divergence in simulation is to take coefficient $\alpha < 2$, in which case the variance of stationary delay is (theoretically) infinite. Thus the main conclusion is that it is difficult to verify long-range dependence of the workload (delay) process neither in single-server, nor in tandem case, applying divergence of the autocovariance series stated in [4].

Nevertheless, an interesting case that leads to the divergence of autocorrelation series is an instability of a station. Consider an M/Pareto/1 $\rightarrow$ /Pareto/1 tandem system renewal input with interarrival time T and with (corresponding) Pareto service time (in more convenient for simulation form)

$$P(S^{(i)} > x) = (1+x)^{-\alpha_i},\ x \geq 0,\ i = 1, 2.$$

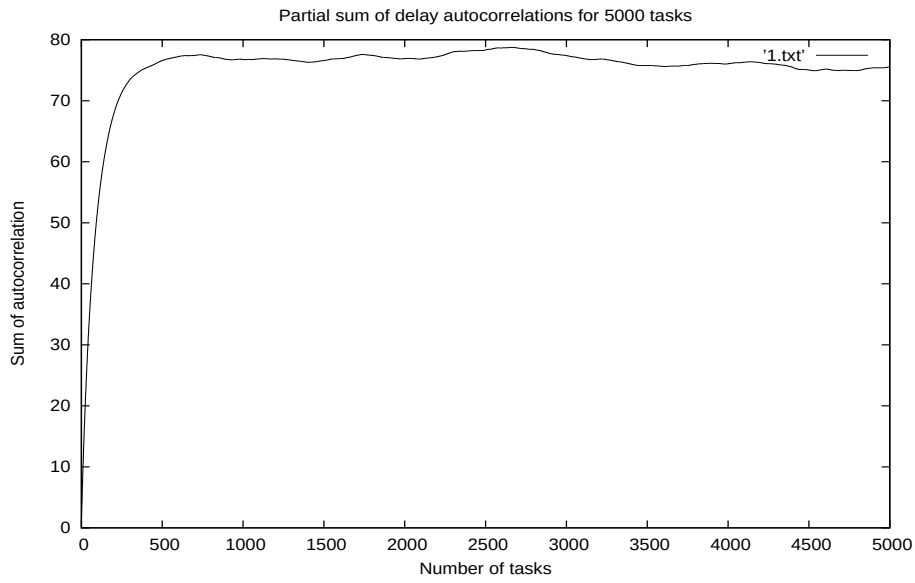


Figure 1: Convergence of autocorrelations, $\lambda = 3, \mu_1 = 3.5, \mu_2 = 4.2$.

Then the mean service time is $ES^{(i)} = (\alpha_i - 1)^{-1}$, $i = 1, 2$. Denote $\lambda = 1/ET$, $\mu_i = 1/ES^{(i)}$ and assume that $\mu_1 < \lambda < \mu_2$. Then first station becomes overloaded (because $\rho_1 = \lambda/\mu_1 > 1$) and in the limit has the output with rate μ_1. But because $\mu_1/\mu_2 < 1$, then the second station is stable (in limit) and we do not observe divergence of autocorrelations of waiting times, as Fig. 1 shows. If $\mu_2 < \lambda < \mu_1$, then the first station is stable, but the second station is unstable (since $\rho_2 > 1$). In this case the delays on the second station may become arbitrary high implying the divergence of autocorrelation series, see Fig. 2.

3 Conclusion

Detection of the long-range dependence in the networking traffic is extremely important to estimate QoS provided in the network. In this note, we verify by simulation this (second-order) property of the workload process in the second station of a two-station tandem network. We discuss the difficulties (caused by technical limitations) which arise when we apply simulation to establish (under appropriate moment conditions) divergence of the autocorrelation series, indicating theoretically the long-range dependence.

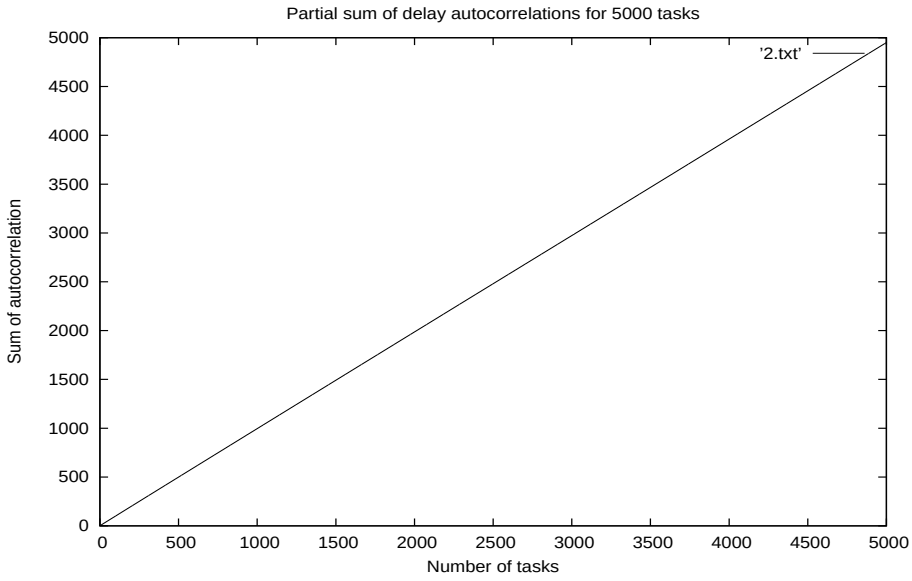


Figure 2: Divergence of autocorrelations, $\lambda = 3, \mu_1 = 4.2, \mu_2 = 3.5$.

4 Acknowledgments

This research is supported by Russian Foundation for Basic Research, project No 10-07-00017.

Bibliography

[1] I. Aban, M. Meerschaert, A. Panorska *Parameter Estimation for the Truncated Pareto Distribution* Journal of the American Statistical Assoc. vol. 101, 2006. pp. 270–277.

[2] D. V. Bodyonov, and E. V. Morozov, *Regenerative simulation of a tandem network with long-range dependent workload process.* Proceedings of FDPW'2004, vol. 6, Petrozavodsk State University, 2005. pp. 170–180.

[3] D. Daley *Queueing Output Process* Adv. Appl. Prob., vol. 8, Applied Probability Trust, 1976. pp. 395–415.

[4] D. Daley *The serial correlation coefficients of waiting times in a stationary single server queue.* J. Austr. Math. Soc. vol. 8, 1968. pp. 683–699.

[5] M. Harchol-Balter *The Effect of Heavy-Tailed Job Size Distribution on Computer System Design.* Proceedings of ASA-IMS Conference on Applications of Heavy Tailed Distributions in Economics, Engineering and Statistics, Washington, DC. 1999.

[6] *High-performance Data Center, KRC RAS* http://cluster.krc.karelia.ru/

[7] T. Huang, K. Sigman. *Steady-state asymptotics for tandem, split-match and other feedforward queues with heavy tailed service* Queueing Systems, vol. 33, 1999. pp. 233–259.

[8] E. Morozov, A. Rumyantsev *Regeneration and correlation properties of stationary delay in single-server queue (in Russian)* Proceedings of the International Wokrshop on Distributed Computer and Communication Networks (DCCN-2010). Moscow: R&D Company "Information and Networking Technologies", 2010. pp. 58–67.

[9] G. Samorodnitsky. *Long Range Dependence* Foundations and Trends in Stochastic Systems. vol. 1, No. 3, 2006. pp. 163–257.

[10] A. Scheller-Wolf, K. Sigman *Moments in Tandem Queues* Operations Research, vol. 46, 1996. pp. 378–380.

[11] W. Whitt. *The Efficiency of One Long Run versus Independent Replications in Steady-State Simulation* Management Science. vol. 37, No. 6, 1991. pp. 645–666.

[12] B. Wolfson *Some Moment Results for Certain Tandem and Multiple-Server Queues* J. Appl. Prob., vol. 21, Applied Probability Trust, 1984. pp. 901–910.

Introduction to store data in Redis, a persistent and fast key-value database

Matti Paksula

Department of Computer Science, University of Helsinki

P.O.Box 68, FI-00014 University of Helsinki, Finland

E-mail: {paksula@cs.helsinki.fi}

Abstract

This article gives short introduction to key-value storage called Redis. In addition a pattern to persist simple objects is provided. The pattern focuses on the consistency of objects under race and failure conditions. Proposed pattern is useful for cloud storage implementations that require speed and scalability. This approach can optionally be implemented with schemaless data model. Implementation can be done with any language, although the examples are presented with Ruby.

1 Introduction

Recent beginning of "NoSQL movement" has brought a lot of attention to key-value databases or stores (from now on *KVS*). Cloud web applications require fast requests and the use of a relational database (from now on *RDBMS*) can sometimes be major bottleneck in the architecture [10]. This does not however imply that RDBMSes should be avoided. The problem is that they are being used as a generic building block for every problem. Simplified KVS brings speed, but also provides very little. This leads to major shift of responsibility from data persistence layer to developers.

NoSQL is easily understood as anti-SQL, but it should be understood as "Not *Only* SQL". KVS are not suitable for replacing relational databases, but to provide better approach in meeting non-functional requirements especially in cloud architectures. These databases or stores can not provide ACID style transactions. However, object persistence needs to be consistent nevertheless. This can be achieved with good design patterns that are implemented both in the architectural and component level. In this paper an example pattern for storing objects in KVS is given and at some level compared to RDBMS persistence.

The rest of this paper is organized as follows: First some background on cloud application architectures and key concepts of KVS are given. Some comparison of KVS against RDBMS is also done. Then introduction for object persistence in KVS is given. After these main principles of object persistence, different KVS implementations are shortly compared and then the key concepts of Redis KVS are explained. A Redis specific implementation of object persistence is illustrated with pseudocode like examples.

2 Architectural motivation

Web application needs to serve requests as quickly as possible. Most of the requests consist mainly of reads from the database. Brewers CAP theorem says that in distributed environment, like cloud, we have three requirements: ***C****onsistency*, ***A****vailability* and ***P****artition Tolerance*, but we can only have *two* out of these three. What this means is that system operations on data need to be consistent, it has to be available for the user and data has to be stored in multiple locations. In order to satisfy all three, we can select two and solve the third requirement by a workaround [1, 2].

Instead of writing our application to satisfy ACID requirements (***A****tomicity*, ***C****onsistency*, ***D****urability* and ***I****solation*) we write our architecture using BASE approach ((***B****asically Available*, ***S****oft-state* and ***E****ventually consistent*) [3]. This means that we do our best to keep data available, but for example if some network link goes down, we still serve that data we have access to. Modifications to the data are guaranteed to perform eventually for example by deferring write operations to a job queue. Due the nature of web applications where requests are independent and have some time between them, this approach can be very successfull.

Dropping ACID on architectural level gives us also the opportunity to drop it in the database level also. This is where the KVS option becomes interesting. Usually most of the operations are reads and scaling web application means scaling up the reads.

3 KVS vs RDBMS

RDBMS has to satisfy ACID requirements when modifying the data. This makes partitioning data accross different physical locations hard as data needs to be written in all the locations before *OK* can be returned. Also the write operations tend to be as hard disk IO becomes the bottleneck. RDBMS also have fixed schema and this can for example lead to bloated table designs with lot of unused columns.

KVS are really simple by design. A value can be saved with a unique key. That value can later be loaded by that same key. This is more generally known as hash addressing described already in 1968 by R.Morris [6]. KVS do not natively support relations or full text queries, but can be a better match to some situations. Examples of such situations are storing operational data, metadata, logging, message queues, caching and serving pre-computed data. One approach to implement a KVS is to store keys in-memory and persist the dataset to file asynchronously. When the KVS server is restarted, all the data is loaded into memory from that file.

RDBMS have serious advantages over KVS: they are well tested, have good management tools and programming patterns available. Key-values on the other hand are fairly new, each of them have different approach and excel only in a very narrow segment [10].

Sometimes the difference between RDBMS and a KVS is almost non-existing. For example when using MySQL by indexed primary keys it can perform really well and depending on the needs can also be easier to partition than current KVS implementations. Also some operations, like sorting can be faster to do at the application level when data is suitable for that. There is no single technology to implement a scalable system. Such system has to be crafted with a combination of different technologies and architectural designs.

Developers existing familarity with RDBMS and the fact that as matured systems they are easier to understand, might be good enough reason for not to consider KVS option. On the other hand scalability requirements might force to consider it.

4 Persisting objects in KVS

Objects are traditionally peristed in RDBMS with Object-Relational mapper (ORM). While KVS implementations differ, there are some common requirements for persistence.

4.1 Objects

Objects are instances of classes implemented in host language. As structure for object and methods etc. is in the code, the entity itself is defined by its unique identifier and attribute values. Simple attribute types like *strings*, *integers*, *booleans* and *floats* are suitable for storing in KVS.

A collection of objects can be defined as a collection of unique identifiers. From these identifiers we can get all the attributes from the database required to create these objects.

An objects attribute can be stored with following key-value pair *ClassName:Identifier:AttributeName* $\Rightarrow$ *Value*.

Example 1. *A car with two attributes as separate keys*
Car:34:color $\Rightarrow$ *green*
Car:34:speed $\Rightarrow$ *120*

This approach can be problematic as descibed in the next section.

4.2 Consistency of objects

Concurrency of reads and writes makes previous approach problematic as KVS does not provide same kind of transactions that we are used to in the RDBMS world. If this is not considered, it can lead to potential inconsistencies. For example during attribute read operation, a concurrent update operation can be modifying the same object. When object is returned to application, it can have some old and some new values. Also, when deleting an object it is possible that execution is terminated for example by power failure and only some of the attributes were deleted.

4.3 Schemaless data model

It is common that not all of the instances have exactly the same attributes than others. Some might have attributes that exist only in the minority of all objects. Schemaless data model is useful during the development phase, but also interesting for the production. For example, an administrator could add attributes on the fly for certain instances and the application could be designed to show only those attributes that are present in the instance. Schemaless approach does not necesary equal chaos if the architecture is designed to support it. Simple example of this is provided in the evaluation chapter.

5 Key-value storages

There are different implementions of KVS and most of them have not yet fully matured. Cloud services have defined a new set of problems what pioneers like Facebook and Amazon are addressing these problems with their own distributed implementations like Cassandra [4] and Dynamo [5]. These distributed large scale KVS have proven to work well for them, but for smaller scale (not massively big) web application they might be too heavy. Things we get for granted with RDBMS like indexing data, providing query language and views are currently under research [7, 10, 11].

Smaller scale KVS implementations include for example CouchDB, MongoDB and Redis. They differ from each other and are suitable for different kind of tasks. For example MongoDB provides a relational database like query language and is essentially a document database. All of these three support partitioning and replication of the data at some level. Mostly experimental object persistence adapters are available for all implementations.

6 Redis introduction

Redis does not support complex queries or indexing, but has support for data structures as values. Being very simple it is also fastest KVS implementations for many basic operations. Speed and data structures are interesting combination that can be used for simple object persistence.

Data structures in Redis are more generally called *Redis Datatypes*. These include strings, lists, sets, sorted sets and hashes. Redis provides commands that can be used to modify these types. For example list supports normal list operations, like push and pop.

The whole dataset is kept in-memory and therefore can not exceed the amount of physical RAM. Redis server writes entire dataset to disk at configurable intervals. This can also be configured so that each modification is always written on the disk before returning OK. Master-Slave replication is also available and clustering is currently under development [9].

Atomic commands and Redis Transactions

Every Redis command is executed atomically. For example the INCR command increments integer value of key atomically. Two separate commands are not atomical without combining them with *Redis Transactions*.

Transactions in Redis are actually queued commands that are executed atomically in sequence. If one command raises an error, all the other commands in the queue are still processed. Redis starts queuing commands after command MULTI and executes those in one atomic transaction with EXEC.

6.1 Sorted sets and Hashes

Sorted sets are similar to RDBMS indexes. Sorted sets contain members that are sorted by a special integer value *score*. For example *"ZADD Foo:all 10 10"* stores value 10 to key *Foo:all* with score of 10.

Hashes are key-value pairs inside a key and suitable for storing objects attributes. For example *"HMSET Cat:2:attributes color black age 20"* adds two key-value pairs to the key *Cat:2:attributes*.

7 Implementation

In following implementation each persisted object has its own unique integer identifier. Based on this identifier a key *Foo:id:attributes* containing the objects attributes is created (assuming that object is named Foo). This key stores the attributes as Redis hash. When the object is created, a special sorted set *Foo:all* is updated to contain the identifier. With this "master" set it is possible to know if object is persisted or not.

Because write operations are different atomic operations, concurrency of writes can lead to inconsistent objects. If a delete operation is stopped (for e.g. power outage) in the between of attribute deletion and removal from the master set, object loses it's attributes. Also some garbage keys can exist in the memory if the object is removed from the master set of *Foo:all* and attributes are not deleted left. To prevent these scenarior each operation has to be designed for race conditions and process termination.

Redis specific implementation details are given in next pseudocode like algorithms. Redis commands are written in capital letters. Only the basic operations are described as further work is needed to provide simple relations and indexes. In the examples class named *Cat* is used over *Foo*, because cats have names and lengths unlike foos.

7.1 Create

Create operation is shown in algorithm 1. Operation fetches new identifier from shared sequence. Then all attributes are set in Redis hash. After this object is added to set of all persisted objects. This is done with ZADD command, that adds objects identifier integer with the same integer as score in the sorted set. Command HMSET accepts many key-value pairs ($HMSET\ key\ field1\ value2\ ...\ fieldN\ valueN$). Each attribute can also be written separately, this approach is shown in alternative create algorithm 2.

Increasing identifier and storing attributes are done in sperate atomic operations. This could lead to situation where identifier is increased, but no object is persisted. This is acceptable for worst case scenario as the dataset is still consistent. Atomic transaction guarantees that attributes do not get written without being added to sorted set *all*.

Algorithm 1 Create an object

Require: *attrs*
id ← **INCR** "Cat:sequence"
MULTI
{HMSET "Cat:3:attributes" "name" "lolcat" "age" "3"} **HMSET** "Cat:*id*:attributes" *attrs*
ZADD "Cat:all" *id id*
EXEC

Algorithm 2 Alternative object creation

Require: *attrs*
id ← **INCR** "Cat:sequence"
MULTI
for all *key*, *value* in *attrs* **do**
  HSET "Cat:*id*:attributes" *key value*
end for
ZADD "Cat:all" *id id*
EXEC

7.2 Load

It is possible to get all the attributes after the object is persisted by using unique object identifier. Returning value of ZSCORE operation against this key is the score of the key in sorted set. If key was not found then a special value *NIL* is returned. Attribute read returns an empty set also when an object does not have any attributes. Therefore reading score and attributes atomically is required to determine if object was stored without any attributes. Also a race condition with delete operation is possible when operations are not atomic.

Algorithm 3 Load an object

Require: *id*
  MULTI
  id ← **ZSCORE** "Cat:all" *id*
  attrs ← **HGETALL** "Cat:*id*:attributes"
  EXEC
  if *id* is zero **then**
    return **false**
  else
    return *attrs*
  end if

7.3 Update

Updating an existing object is equal to the creation as shown in algorithm 1. Alternative creation algorithm (algorithm 2) can provide better performance when only some of the keys are updated. In the worst case performance is the same as in create.

Update can also be used when adding and removing attributes to and from an existing object. This requires a host language that supports adding new attributes dynamically in objects or some other method.

7.4 Delete

Delete (in algorithm 4) is done by combining deletion of attributes and removal from sorted set *all* into one atomic transaction.

Algorithm 4 Delete an object

Require: *id*
  MULTI
  DEL "Cat:*id*:attributes"
  ZREM "Cat:all" *id*
  EXEC

7.5 Count

Counting (in algorithm 5) the number of persisted objects is the cardinality of sorted set *all*. Time complexity is O(1).

Algorithm 5 Count objects

  $i \leftarrow$ **ZCARD** "Cat:all"
  return *i*

8 Conclusion

Using novel database techniques is still pretty experimental. A quote from the Redis mailing list summarises it all:

> "However, I have to admit, I've definitely had sleepless nights. Redis is a new technology. We explored new territory and had little experience (internal or otherwise) to rely upon. We've had to debug and patch the driver, write our own query layer, backups, data injection scripts, all sorts of stuff you take for granted from the SQL world, and we're still tweaking it, and wishing for pre-made solutions." [8]

Key-value databases have not yet fully matured, but do give interesting options for developing distributed and scalable web applications. Each of different implementations are good for a set of problems, but not for every problem. Therefore well defined architectural design patterns and adapters for storing objects are needed to provide a tested approach for developing stable software. Relational databases are still good for handling relations and supporting strong consistency requirements. The future of web databases is moving from homogeneous to heterogeneous collection of different application specific database systems.

Bibliography

[1] Brewer, E. A. 2000. Towards robust distributed systems (abstract). In Proceedings of the Nineteenth Annual ACM Symposium on Principles of Distributed Computing (Portland, Oregon, United States, July 16 - 19, 2000). PODC '00. ACM, New York, NY, 7. DOI= http://doi.acm.org/10.1145/343477.343502

[2] Gilbert, S. and Lynch, N. 2002. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. SIGACT News 33, 2 (Jun. 2002), 51-59. DOI= http://doi.acm.org/10.1145/564585.564601

[3] Pritchett, D. 2008. BASE: An Acid Alternative. Queue 6, 3 (May. 2008), 48-55. DOI= http://doi.acm.org/10.1145/1394127.1394128

[4] A. Lakshman, P. Malik, and K. Ranganathan. Cassandra: A Structured Storage System on a P2P Network, product presentation at SIGMOD 2008.

[5] DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., and Vogels, W. 2007. Dynamo: amazon's highly available key-value store. In Proceedings of Twenty-First ACM SIGOPS Symposium on Operating Systems Principles (Stevenson, Washington, USA, October 14 - 17, 2007). SOSP '07. ACM, New York, NY, 205-220. DOI= http://doi.acm.org/10.1145/1294261.1294281

[6] Morris, R. 1968. Scatter storage techniques. Commun. ACM 11, 1 (Jan. 1968), 38-44. DOI= http://doi.acm.org/10.1145/362851.362882

[7] Agrawal, P., Silberstein, A., Cooper, B. F., Srivastava, U., and Ramakrishnan, R. 2009. Asynchronous view maintenance for VLSD databases. In Proceedings of the 35th SIGMOD international Conference on Management of Data (Providence, Rhode Island, USA, June 29 - July 02, 2009). C. Binnig and B. Dageville, Eds. SIGMOD '09. ACM, New York, NY, 179-192. DOI= http://doi.acm.org/10.1145/1559845.1559866

[8] Message in Redis mailing list http://groups.google.com/group/redis-db/msg/ca398a90ea78bfc5

[9] Redis home page http://code.google.com/p/redis/

[10] Armbrust, M., Lanham, N., Tu, S., Fox, A., Franklin, M., and Patterson, D. A. Piql: A performance insightful query language for interactive applications. First Annual ACM Symposium on Cloud Computing (SOCC)

[11] Acharya, S., Carlin, P., Galindo-Legaria, C., Kozielczyk, K., Terlecki, P., and Zabback, P. 2008. Relational support for flexible schema scenarios. Proc. VLDB Endow. 1, 2 (Aug. 2008), 1289-1300. DOI= http://doi.acm.org/10.1145/1454159.1454169

The optimal implementation of n FIFO-queues in single-level memory

Prof. A.V. Sokolov[†], A.V. Drac[‡]

[†,‡]Institute of Apply Mathematical Researh,
Karelian Scientific Center of Russian Academic of Science [‡] Petrozavodsk State University, Petrozavodsk

E-mail: avs@krc.karelia.ru, adeon88@mail.ru

Abstract

In this article we research methods of n FIFO-queues allocation in the memory of size m units. The problem of optimal memory partition between queues in the case of consecutive circular implementation and the problem of the analysis of linked list implementation are investigated. As mathematical models we proposed random walks into different areas of n-measured space.

1 Introduction

In many applications there is a problem of allocation of multiple queues in single-level memory. There are two fundamentally different ways of organizing work with dynamic data structures – consecutive and linked list allocation. This paper is the extension of [1]. For queues with sequential presentation all the memory is splitted into several parts and each queue is allocated in separate section of memory. In this case we will have losses of memory when any queue overflows and other queues don't overflow.

The linked list implementation is the second method. In this view the data structure is stored as a list. In this case any number of elements of the lists can coexist in the memory area until the free memory is exhausted. But on the other hand, this approach requires an additional link field for each element.

In this article we considered the system where memory overflow is not an emergency situation. If free memory is exausted then all attempts to include an element into queue will lead to it loss until the appearance of free

memory i.e. until the deletion of an element from the tail of queue. Such scheme is used for example in the routers and such behaviour of queues is name "reset tail". As the optimality criterion we considered the minimal part of time which the system is situated in the state of reset tail. It is reasonably to minimize this value in order to minimize the part of lost elements. In this paper we considered linked and sequential presentation of queues and calculated in symbolic form the average part of time which the system is situated in the state of "reset tail".

2 The problem

Consider n queues in single level memory size m. Assume that the time is discrete and only one of the following operations can happen during each time step:

- insertion of element into j-th data stucture with the probability p_j $(1 \leq j \leq n)$,
- deletion of element from j-th data stucture with the probability q_j $(1 \leq j \leq n)$,
- access the element with the probability r (Data structures don't change their lengths).

$p_1 + \cdots + p_n + q_1 + \cdots + q_n + r = 1$.
Values p_j, q_j, r are constants. They don't depend on the current lengths of queues and on the operations on the previous steps. All elements have the same lengths. The length of queue is the number of elements that it contains. Denote i_j is the lengths of data structure with number j.

Consider the memory overflow and transition into the states of "reset tail" in different cases:

1. Linked list implementation.
   Denote l is the ratio of the size of element to the size of a pointer (for the linked presentation). Denote $m(1 - 1/l) = M$. An overflow will occur (and, hence, the system will move to the state of "reset tail") when the queues will occupy all memory (i.e. $i_1 + \cdots + i_n = M$) and an element to any of them will be attempted to include.

2. Consecutive presentation.
   Each queue is allocated in its own part of memory. k_j is the size of allocated memory for j-th queue. An overflow will occur when j-th queue occupy all allocated memory (its length will be equal to k_j)

and an element to this queue will be attempted to include. The free memory allocated to other queues will not be redistributed.

We suppose, that the process starts from the state, when data structures are empty, and in the case of deletion from an empty data structure there is no shutdown. The problem is to find the average part of time when the system is situated in the state of reset tail in both cases of representation and compare them. To solve this problem we used apparatus of regular markov's chains.

3 Consecutive allocation of the queues

3.1 The problem

Consider $k_1, \dots, k_n$ is the fixed partition of the memory.

As the mathematical model we consider the random walk on an integer lattice space inside n-dimensional parallelepiped with vertex at the origin, edges parallel to the axes and the lengths of edges $k_1, \dots, k_n$. Number of states is equal to $\prod_{i=1}^{n}(k_i+1)$. $(i_1, \dots, i_n)$ is the state of the system. $0 \le i_1 \le k_1 + 1, \dots, 0 \le i_n \le k_n + 1$. $i_j = k_j + 1$ are the states of "reset tail". $\alpha_{i_1 \dots i_n}$ is the limit probability which system is situated in the state $(i_1, \dots, i_n)$.

Conversion of the system from state $(i_1, \dots, i_n)$ to state $(i'_1, \dots, i'_n)$ occurs in according to the following rules (fig. 1):

$$(\dots, i_s, \dots, i_t, \dots) \xrightarrow{p_s} \begin{cases} (\dots, i_s+1, \dots, i_t, \dots) & 0 \le i_s \le k_s, i_t \le k_t \\ (\dots, i_s+1, \dots, i_t-1, \dots) & 0 \le i_s \le k_s, i_t = k_t+1 \\ (\dots, i_s, \dots, i_t, \dots) & i_s = k_s+1, i_t \le k_t \\ (\dots, i_s, \dots, i_t-1, \dots) & i_s = k_s+1, i_t = k_t+1 \end{cases}$$

$$(\dots, i_s, \dots, i_t, \dots) \xrightarrow{q_s} \begin{cases} (\dots, i_s, \dots, i_t, \dots) & i_s = 0, x_j \le k_t \\ (\dots, i_s, \dots, i_t-1, \dots) & i_s = 0, x_j = k_t+1 \\ (\dots, i_s-1, \dots, i_t, \dots) & 1 \le i_s \le k_s, i_t \le k_t \\ (\dots, i_s-1, \dots, i_t-1, \dots) & 1 \le i_s \le k_s, i_t = k_t+1 \\ (\dots, i_s-2, \dots, i_t, \dots) & i_s = k_s+1, i_t \le k_t \\ (\dots, i_s-2, \dots, i_t-1, \dots) & i_s = k_s+1, i_t = k_t+1 \end{cases}$$

$$(\dots, i_s, \dots, i_t, \dots) \xrightarrow{r} \begin{cases} (\dots, i_s, \dots, i_t, \dots) & 0 \le i_s \le k_s, i_t \le k_t \\ (\dots, i_s, \dots, i_t-1, \dots) & 0 \le i_s \le k_s, i_t = k_t+1 \\ (\dots, i_s-1, \dots, i_t, \dots) & i_s = k_s+1, i_t \le k_t \\ (\dots, i_s-1, \dots, i_t-1, \dots) & i_s = k_s+1, i_t = k_t+1 \end{cases}$$

$1 \le s \le n, \quad 1 \le t \le n, \quad s \ne t$

Construct the balance equation $\alpha_i = \sum_i \alpha_j P_{ji}$. For our system it will be the following (for internal states):

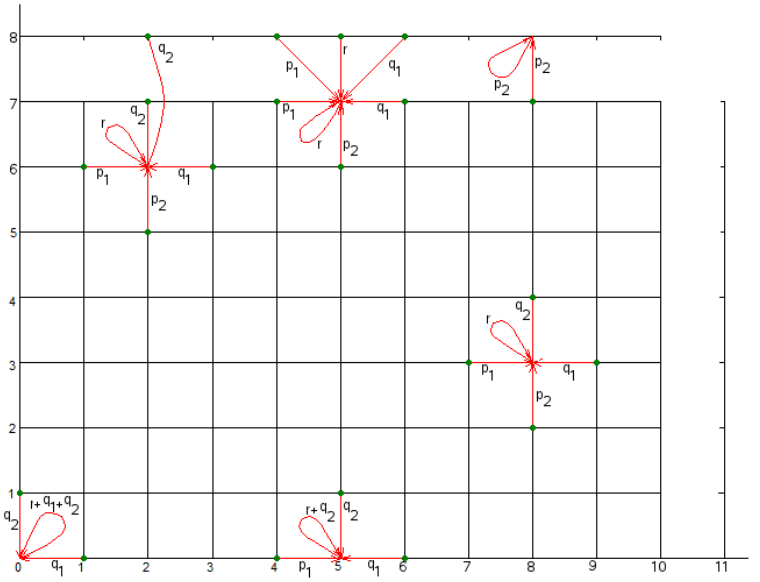


Figure 1: Transition between states in the case of consecutive presentation

1. $\alpha_{i_1 i_2 \ldots i_n} = p_1 \alpha_{i_1 - 1, i_2 \ldots i_n} + q_1 \alpha_{i_1 + 1, i_2 \ldots i_n} + \cdots + p_n \alpha_{i_1 i_2 \ldots i_n - 1} + q_n \alpha_{i_1 i_2 \ldots i_n + 1} + r \alpha_{i_1 i_2 \ldots i_n}$
   $(1 \le i_j \le k_j - 2,\ 1 \le j \le n)$

2. $\alpha_{0 i_2 \ldots i_n} = q_1 \alpha_{1, i_2 \ldots i_n} + \cdots + p_n \alpha_{0 i_2 \ldots i_n - 1} + q_n \alpha_{0 i_2 \ldots i_n + 1} + (r + q_1) \alpha_{0 i_2 \ldots i_n}$
   $(i_1 = 0, 1 \le i_j \le k_j - 2,\ 2 \le j \le n)$

3. $\alpha_{k_1 - 1 i_2 \ldots i_n} = p_1 \alpha_{k_1 - 2, i_2 \ldots i_n} + q_1 \alpha_{k_1, i_2 \ldots i_n} + q_1 \alpha_{k_1 + 1, i_2 \ldots i_n} + \cdots + p_n \alpha_{i_1 i_2 \ldots i_n - 1} + q_n \alpha_{i_1 i_2 \ldots i_n + 1} + r \alpha_{k_1 - 1, i_2 \ldots i_n}$
   $(i_1 = k_1 - 1, 1 \le i_j \le k_j - 2,\ 2 \le j \le n)$

4. $\alpha_{k_1 i_2 \ldots i_n} = p_1 \alpha_{k_1 - 1, i_2 \ldots i_n} + r \alpha_{k_1 + 1, i_2 \ldots i_n} + \cdots + p_n \alpha_{i_1 i_2 \ldots i_n - 1} + q_n \alpha_{i_1 i_2 \ldots i_n + 1} + r \alpha_{k_1 i_2 \ldots i_n}$
   $(i_1 = k_1, 1 \le i_j \le k_j - 2,\ 2 \le j \le n)$

5. $\alpha_{k_1 + 1 i_2 \ldots i_n} = p_1 \alpha_{k_1, i_2 \ldots i_n} + p_1 \alpha_{k_1 + 1, i_2 \ldots i_n}$
   $(i_1 = k_1, 1 \le i_j \le k_j - 2,\ 2 \le j \le n)$

Split the set of indexes $I = \{1, \ldots, n\}$ into the subsets:
$I_1 = \{j : i_j = 0\}$

$I_2 = \{j : 1 \le i_j \le k_j - 2\}$
$I_3 = \{j : i_j = k_j - 1\}$
$I_4 = \{j : i_j = k_j\}$
$I_5 = \{j : i_j = k_j + 1\}$

Since only one of queues can be into the state of "reset tail" then $|I_5| = 0$ or $|I_5| = 1$.

1. For usual states (equations 1–4 in common view, i.e. when $I_5 = \emptyset$):
$$\alpha_{i_1 \dots i_n} = (r + \sum_{j \in I_1} q_j)\alpha_{i_1 \dots i_n} + \sum_{j \in I_1} q_j \alpha_{i_1 \dots i_j + 1 \dots i_n} + \sum_{j \in I_2 + I_3} (p_j \alpha_{i_1 \dots i_j - 1 \dots i_n} + q_j \alpha_{i_1 \dots i_j + 1 \dots i_n}) + \sum_{j \in I_3} q_j \alpha_{i_1 \dots i_j + 2 \dots i_n} + \sum_{j \in I_4} \Bigg(r\alpha_{i_1 \dots i_j + 1 \dots i_n} + p_j \alpha_{i_1 \dots i_j - 1 \dots i_n} + \sum_{l \in I_2 + I_3} (p_l \alpha_{i_1 \dots i_l - 1 \dots i_j + 1 \dots i_n} + q_l \alpha_{i_1 \dots i_l + 1 \dots i_j + 1 \dots i_n}) + \sum_{l \in I_1} q_l (\alpha_{i_1 \dots i_l \dots i_j + 1 \dots i_n} + \alpha_{i_1 \dots i_l + 1 \dots i_j + 1 \dots i_n}) \Bigg)$$

2. For the states of "reset tail" (equation 5):
$$\alpha_{i_1 \dots k_j + 1 \dots i_n} = p_j(\alpha_{i_1 \dots k_j \dots i_n} + \alpha_{i_1 \dots k_j + 1 \dots i_n})$$

The system has the following solution:

$$\alpha_{i_1 \dots i_n} = C \left(\frac{p_1}{q_1}\right)^{i_1} \dots \left(\frac{p_n}{q_n}\right)^{i_n} \left(1 - \sum_{j \in I_4} p_j\right) \qquad I_5 = \emptyset$$

$$\alpha_{i_1 \dots i_j \dots i_n} = C \left(\frac{p_1}{q_1}\right)^{i_1} \dots \left(\frac{p_j}{q_j}\right)^{i_j - 1} \dots \left(\frac{p_n}{q_n}\right)^{i_n} p_j \qquad I_5 = \{j\}$$

Denote $p_i/q_i = x_i$, $i = 1, \dots, n$. Let for queues with numbers $1, \dots, s$ $x_i \ne 1$, i.e. $p_i \ne q_i$, and for queues with number $s+1, \dots, n$ $x_i = 1$, i.e. $p_i = q_i$. Find the constant C from the normalizatin equation:

$$\begin{aligned} \frac{1}{C} &= \sum_{i_1=0}^{k_1} \dots \sum_{i_n=0}^{k_n} x_1^{i_1} \dots x_n^{k_n} = \sum_{i_1=0}^{k_1} x_1^{i_1} \dots \sum_{i_n=0}^{k_n} x_n^{k_n} = \\ &= \prod_{i=1}^{s} \frac{x_i^{k_i+1} - 1}{x_i - 1} \prod_{i=s+1}^{n} (k_i + 1) \end{aligned}$$

Summarise all $\alpha_{i_1 \dots i_n}$ which the states of "reset tail". For queue with number 1 it will be the states with the condition $i_1 = k_1$, $0 \le i_j \le i_n$,

$2 \le j \le n$.

$$p_1^* = p_1 x_1^{k_1} \frac{1}{C} \sum_{i_2=0}^{k_2} \cdots \sum_{i_n=0}^{k_n} x_2^{i_2} \dots x_n^{k_n} = p_1 x_1^{k_1} \frac{1}{C} \sum_{i_2=0}^{k_2} x_2^{i_2} \cdots \sum_{i_n=0}^{k_n} x_n^{k_n} =$$

$$p_1 x_1^{k_1} \frac{1}{C} \prod_{i=1}^{s} \frac{x_i^{k_i+1} - 1}{x_i - 1} \prod_{i=s+1}^{n} (k_i + 1) = p_1 x_1^{k_1} \frac{x_1 - 1}{x_1^{k_1+1} - 1} =$$

$$\frac{p_1 \left(\frac{p_1}{q_1}\right)^{k_1} \left(\frac{p_1}{q_1} - 1\right)}{\left(\frac{p_1}{q_1}\right)^{k_1+1} - 1} = \frac{p_1(p_1^{k_1+1} - p_1^{k_1} q)}{p_1^{k_1+1} - q_1^{k_1+1}} = \frac{p_1 - q_1}{1 - \left(\frac{q_1}{p_1}\right)^{k_1+1}} = \frac{q_1 - p_1}{\left(\frac{q_1}{p_1}\right)^{k_1+1} - 1}$$

Similarly for the queues with numbers $2, \dots, s$.

For the queue with number $s + 1$:

$$p_{s+1}^* = \frac{p_{s+1}}{k_{s+1} + 1}$$

Similarly for the queues with numbers $s + 2, \dots, n$.

Then the summary part of time which the system is situated in the state of "reset tail" is equal to:

$$P_c^* = \sum_{i=1}^{n} p_i^* = \sum_{i=1}^{s} \frac{q_i - p_i}{\left(\frac{q_i}{p_i}\right)^{k_i+1} - 1} + \sum_{i=s+1}^{n} \frac{p_i}{k_i + 1}$$

The problem of optimal division of memory was solved in [1].

4 Linked list presentation of queues

As the mathematical model we consider the random walk on an integer lattice space inside n-dimensional pyramid with edges $0 \le x_1 \le M$, $0 \le x_2 \le M$, $\dots$, $0 \le x_n \le M$ and base $x_1 + x_2 + \cdots + x_n = M$.

For each state in the face $x_1 + x_2 + \cdots + x_n = M$, i.e. when all the memory is exausted, define the corresponding state of "reset tail". Denote it as $(\bar{x_1}, \dots, \bar{x_n})$. It can be reached in case of inserting of an element into any of queues. Conversion of the system from state $(x_1, \dots, x_n)$ to state $(x'_1, \dots, x'_n)$ occurs in according to the following rules (fig. 2):

$$(\dots, i_s, \dots, i_t, \dots) \xrightarrow{p_s} \begin{cases} (\dots, i_s + 1, \dots, i_t, \dots) & 0 \le i_1 + \cdots + i_n < M \\ (\dots, \bar{x_s}, \dots, \bar{x_j}, \dots) & i_1 + \cdots + i_n = M \end{cases}$$

$$(\dots, i_s, \dots, i_t, \dots) \xrightarrow{q_s} \begin{cases} (\dots, i_s - 1, \dots, i_t, \dots) & i_s > 0 \\ (\dots, i_s, \dots, i_t, \dots) & i_s = 0 \end{cases}$$

$$(\dots, i_s, \dots, i_t, \dots) \xrightarrow{r} (\dots, i_s, \dots, i_t, \dots)$$

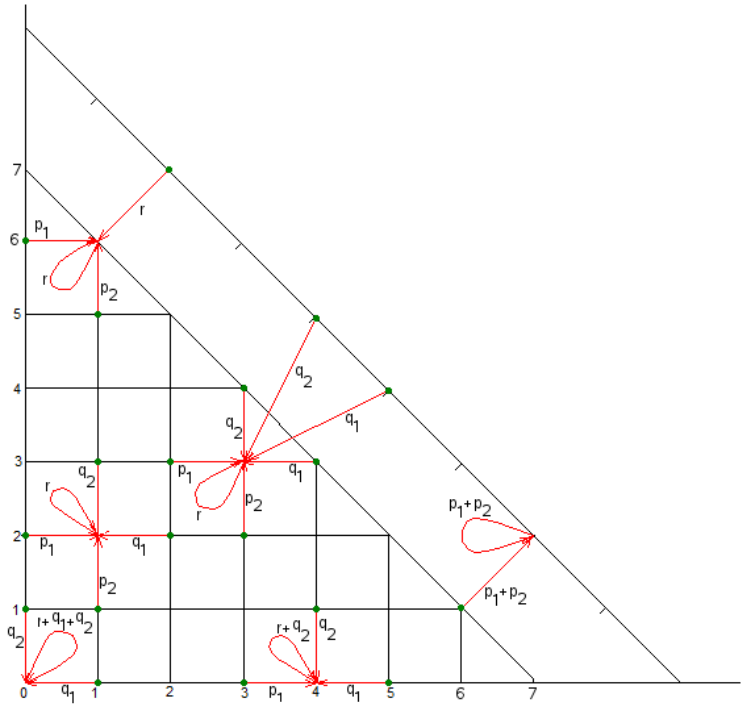


Figure 2: Transition between states in the case of linked list presentation

$$(\dots, \bar{i_s}, \dots, \bar{i_t}, \dots) \xrightarrow{p_s} (\dots, \bar{i_s}, \dots, \bar{i_t}, \dots)$$

$$(\dots, \bar{i_s}, \dots, \bar{i_t}, \dots) \xrightarrow{q_t} \begin{cases} (\dots, i_s - 1, \dots, i_t, \dots) & i_s > 0 \\ (\dots, i_s, \dots, i_t, \dots) & i_s = 0 \end{cases}$$

$$(\dots, \bar{i_s}, \dots, \bar{i_t}, \dots) \xrightarrow{r} (\dots, i_s, \dots, i_t, \dots)$$

Split the set of indexes $I = \{1, \dots, n\}$ into the subsets:

$I_1 = \{j : i_j = 0\}$

$I_2 = \{j : i_j > 0\}$

Construct the balance equation $\alpha_i = \sum_i \alpha_j P_{ji}$. For our system it will be the following:

1. $\alpha_{i_1 \dots i_n} = \sum_{j \in I_1} (q_j \alpha_{i_1 \dots i_j \dots i_n} + q_j \alpha_{i_1 \dots i_j + 1 \dots i_n}) + \sum_{j \in I_2} (p_j \alpha_{i_1 \dots i_j - 1 + i_n} + q_j \alpha_{i_1, \dots i_j + 1 \dots i_n}) + r \alpha_{i_1 \dots i_n}$
   $(0 \le i_1 + \dots + i_n \le M - 2)$

2. $\alpha_{i_1\dots i_n} = \sum\limits_{j\in I_1}(q_j\alpha_{i_1,\dots i_j+i_n} + q_j(\alpha_{i_1,\dots i_j+1\dots i_n} + \overline{\alpha_{i_1\dots i_j\dots i_n}})) +$
$\sum\limits_{j\in I_2}(p_j\alpha_{i_1\dots i_j-1+i_n} + q_j(\alpha_{i_1,\dots i_j+1\dots i_n} + \overline{\alpha_{i_1\dots i_j\dots i_n}})) + r\alpha_{i_1\dots i_n}$
$(i_1+\dots+i_n = M-1)$

3. $\alpha_{i_1\dots i_n} = p_1\alpha_{i_1-1,i_2\dots i_n} + \dots + p_n\alpha_{i_1 i_2\dots i_n-1} + r(\alpha_{i_1 i_2\dots i_n} + \overline{\alpha_{i_1,i_2\dots i_n}})$
$(i_1+\dots+i_n = M)$

4. $\overline{\alpha_{i_1\dots i_n}} = (p_1+\dots+p_n)(\alpha_{i_1 i_2\dots i_n} + \overline{\alpha_{i_1 i_2\dots i_n}})$
$(i_1+\dots+i_n = M)$

The system has the following solution:

$$\alpha_{i_1 i_2\dots i_n} = C\left(\frac{p_1}{q_1}\right)^{i_1}\dots\left(\frac{p_n}{q_n}\right)^{i_n} \qquad 0 \le i_1+\dots+i_n \le M-1$$

$$\alpha_{i_1 i_2\dots i_n} = C(1-p_1-\dots-p_n)\left(\frac{p_1}{q_1}\right)^{i_1}\dots\left(\frac{p_n}{q_n}\right)^{i_n} \qquad i_1+\dots+i_n = M$$

$$\overline{\alpha_{i_1 i_2\dots i_n}} = C(p_1+\dots+p_n)\left(\frac{p_1}{q_1}\right)^{i_1}\dots\left(\frac{p_n}{q_n}\right)^{i_n} \qquad i_1+\dots+i_n = M$$

Find the constant C from the normalizatin equation:

$$\frac{1}{C} = \sum_{i_n=0}^{M}\sum_{i_{n-1}=0}^{M-i_n}\cdots\sum_{i_1=0}^{M-i_2-\dots-i_n} x_1^{i_1}\dots x_n^{i_n}$$

Find the summary part of time which system situated in the state of "reset tail".

4.1 Statement

Let $a_1,\dots,a_k$ are the distinct numbers, $0 \le s \le k-1$, then:

$$\frac{a_1^s}{(a_1-a_2)(a_1-a_3)\dots(a_1-a_k)} + \frac{a_2^s}{(a_2-a_1)(a_2-a_3)\dots(a_2-a_k)} + \dots +$$
$$+ \frac{a_k^s}{(a_k-a_1)(a_k-a_2)\dots(a_k-a_{k-1})} = 0$$

Proof:

Lead the left part to the common denominator. p-th summand will have $p-1$ changes of sign. All brackets without a_p will be into the p-th summand a_p

$$\frac{1}{\prod\limits_{1\le i<j\le k}(a_i-a_j)}\left(a_1^s\prod_{2\le i<j\le k}(a_i-a_j)+\dots+(-1)^{p+1}a_p^s\prod_{\substack{1\le i<j\le k\\ i\ne p\ne j}}(a_i-a_j)+\right.$$
$$\dots +$$
$$\left. + a_k^s\prod_{1\le i<j\le k-1}(a_i-a_j)\right) =$$

Represent the sum in brackets in the form of a determinant. In every summand the procuct will be the value of Vandermond's determinant $(p-1)$-th order.

$$= \begin{vmatrix} a_1^s & a_2^s & a_3^s & \dots & a_k^s \\ 1 & 1 & 1 & \dots & 1 \\ a_1 & a_2 & a_3 & \dots & a_k \\ a_1^2 & a_2^2 & a_3^2 & \dots & a_k^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_1^{k-1} & a_2^{k-1} & a_3^{k-1} & \dots & a_k^{k-1} \end{vmatrix} = 0$$

Determinant is equal to 0, because it has 2 equal rows

4.2 Case of different values of probabilities

Consider the case when $x_i \neq 1 \ \ \forall i$ and $x_i \neq x_j$ when $i \neq j$:

$$\frac{1}{C} = \frac{x_1^{M+n}}{(x_1-1)(x_1-x_2)\dots(x_1-x_n)} + \dots + \frac{x_n^{M+n}}{(x_n-1)(x_n-x_1)\dots(x_n-x_{n-1})} + \frac{1}{(1-x_1)\dots(1-x_n)} = f_n(X,M)$$

Method of mathematical induction:

1) Base $n = 2$:

$$\sum_{i_2=0}^{M}\sum_{i_1=0}^{M-i_2} x_1^{i_1}x_2^{i_2} = \sum_{i_2=0}^{M} x_2^{i_2}\sum_{i_1=0}^{M-i_2} x_2^{i_2} = \sum_{i_2=0}^{M} x_2^{i_2}\frac{x_1^{M-i_2+1}-1}{x_1-1} =$$

$$= \sum_{i_2=0}^{M}\frac{x_1^{M-i_2+1}x_2^{i_2}}{x_1-1} - \sum_{i_2=0}^{M}\frac{x_2^{i_2}}{x_1-1} = \frac{x_1}{x_1-1}\frac{x_2^{M+1}-x_1^{M+1}}{x_2-x_1} - \frac{x_2^{M+1}-1}{(x_1-1)(x_2-1)} =$$

$$= \frac{x_1^{M+2}}{(x_1-1)(x_1-x_2)} + \frac{1}{(1-x_1)(1-x_2)} - x_2\left(\frac{x_1}{(x_1-1)(x_1-x_2)} + \frac{1}{(1-x_1)(1-x_2)}\right) = \frac{x_1^{M+2}}{(x_1-1)(x_1-x_2)} + \frac{x_2^{M+2}}{(x_2-1)(x_2-x_1)} + \frac{1}{(1-x_1)(1-x_2)}$$

2) Suppose it is true for $(n-1)$-th queue.

3) Proof for n:

$$f_n(X,M) = \sum_{i_n=0}^{M} x_n^{i_n}\sum_{i_{n-1}=0}^{M-i_n}\cdots\sum_{i_1=0}^{M-i_2-\dots-i_n} x_1^{i_1}\dots x_{n-1}^{i_{n-1}} =$$

$$\sum_{j=0}^{n} x_n^j (f_{n-1}(M))^{n-j}$$

Simplify summands:

$$\sum_{j=0}^{M} x_n^j = \frac{x_n^{M+1}-1}{x_n-1}$$

$$\sum_{j=0}^{M} x_n^j x_i^{M-j+n-1} = \frac{x_i^{n-1}(x_n^{M+1}) - x_i^{M+1}}{x_n-x_i} = \frac{x_i^{M+n}}{x_i-x_n} - \frac{x_n^{M+1}x_i^{n-1}}{x_i-x_n}$$

$(1 \le i \le n-1)$

Then:

$$f_n(X,M) = \frac{x_1^{M+n}}{(x_1-1)(x_1-x_2)\dots(x_1-x_n)} + \dots + \frac{x_{n-1}^{M+n}}{(x_{n-1}-1)(x_{n-1}-x_1)\dots(x_{n-1}-x_n)} + \frac{1}{(1-x_1)\dots(1-x_n)} - x_n^{M+1}\left(\frac{x_1^{n-1}}{(x_1-1)(x_1-x_2)\dots(x_1-x_n)} + \dots + \frac{x_{n-1}^{n-1}}{(x_{n-1}-1)(x_{n-1}-x_1)\dots(x_{n-1}-x_n)} + \frac{1}{(1-x_1)\dots(1-x_n)}\right) =$$
$$= \frac{x_1^{M+n}}{(x_1-1)(x_1-x_2)\dots(x_1-x_n)} + \dots + \frac{x_n^{M+n}}{(x_n-1)(x_n-x_1)\dots(x_n-x_{n-1})} + \frac{1}{(1-x_1)\dots(1-x_n)}$$

Then $C = \dfrac{1}{f_n(X,M)}$. Summarise $\alpha_{i_1\dots i_n}$, which corresponds to the states of "reset tail" :

$$(p_1 + \dots + p_n)\sum_{i_n=0}^{M}\sum_{i_{n-1}=0}^{M-i_n}\cdots\sum_{i_2=0}^{M-i_3-\dots-i_n} x_1^{M-i_2-\dots-i_n}x_2^{i_2}\dots x_n^{i_n} = g_n(X,M)$$

Proof that

$$g_n(X,M) = (p_1 + \dots + p_n)\frac{x_1^{M+n-1}}{(x_1-x_2)\dots(x_1-x_n)} + \dots + \frac{x_n^{M+n-1}}{(x_n-x_1)\dots(x_n-x_{n-1})}$$

Method of mathematical induction:

1) Base $n = 2$:

$$(p_1 + p_2)\sum_{i=0}^{M} x_1^i x_2^{M-i} = (p_1 + p_2)\frac{x_1^{M+1}-x_2^{M+1}}{x_1-x_2} = (p_1 + p_2)\left(\frac{x_1^{M+1}}{x_1-x_2} + \frac{x_2^{M+1}}{x_2-x_1}\right)$$

Steps 2) and 3) are the same as in calculating constant C

$$P^* = (p_1 + \dots + p_n)\frac{g_n(X,M)}{f_n(X,M)}$$

4.3 Common case

Suppose that there are k_0 queues that have the probabilities: $\dfrac{p_i}{q_i} = x_0 = 1$,

k_1 queues, that: $\dfrac{p_{i_1}}{q_{i_1}} = \dots = \dfrac{p_{i_{k_1}}}{q_{i_{k_1}}} = x_1$

...

k_s queues, that: $\dfrac{p_{j_1}}{q_{j_1}} = \dots = \dfrac{p_{j_{k_s}}}{q_{j_{k_s}}} = x_s$

$k_0 + k_1 + \dots + k_s = n$

Find the constant C:

$$\begin{aligned}\frac{1}{C} &= \sum_{i_s=0}^{M}\sum_{i_{s-1}=0}^{M-i_s}\cdots\sum_{i_1=0}^{M-i_2-\cdots-i_s}\sum_{i_0=0}^{M-i_1-\cdots-i_s} \\ &\binom{k_s+i_s-1}{k_s}x_s^{i_s}\binom{k_{s-1}+i_{s-1}-1}{k_{s-1}}x_{s-1}^{i_{s-1}}\cdots\binom{k_1+i_1-1}{k_1}x_1^{i_1}\binom{k_0+i_0}{k_0} = \\ &= f_s^*(X,M)\end{aligned}$$

Proof that

$$\begin{aligned}f^*(X,M) &= \frac{\partial^{n-s}}{\partial^{k_0}x_0\partial^{k_1-1}x_1\ldots\partial^{k_s-1}x_s}\left\{\frac{1}{k_0!(k_1-1)!\ldots(k_s-1)!}\right. \\ &\left.\left(\frac{x_0^{M+n}}{(x_0-x_1)\ldots(x_0-x_s)}+\cdots+\frac{x_s^{M+n}}{(x_s-x_0)\ldots(x_s-x_{s-1})}\right)\right\}\end{aligned}$$

Method of mathematical induction:

1) Base $s=1$:

$$\sum_{j=0}^{M}\binom{k_1+j-1}{j}x_1^j\binom{M+k_0-j}{M-j} = \sum_{j=0}^{M}\frac{(k_1+j-1)!}{j!(k_1-1)!}x_1^j\frac{(M+k_0-j)!}{(M-j)!k_0!}x_0^{M-j} =$$

$$= \frac{1}{k_0!(k_1-1)!}\sum_{j=0}^{M}(M-j+1)\ldots(M-j+k_0)x_0^{M-j}(j+1)\ldots(j+k_1-1)x_1^j =$$

$$= \frac{\partial^{k_0+k_1-1}}{\partial^{k_0}x_0\partial^{k_1-1}x_1}\left\{\frac{1}{k_0!(k_1-1)!}\sum_{j=0}^{M}x_0^{M-j+k_0}x_1^{j+k_1-1}\right\} =$$

$$= \frac{\partial^{k_0+k_1-1}}{\partial^{k_0}x_0\partial^{k_1-1}x_1}\left\{\frac{1}{k_0!(k_1-1)!}\sum_{j=-k_1+1}^{M+k_0}x_0^{M-j+k_0}x_1^{j+k_1-1}\right\} =$$

$$= \frac{\partial^{k_0+k_1-1}}{\partial^{k_0}x_0\partial^{k_1-1}x_1}\left\{\frac{1}{k_0!(k_1-1)!}\frac{x_0^{M+k_0+k_1}-x_1^{M+k_0+k_1}}{x_0-x_1}\right\} =$$

$$= \frac{\partial^{n-s}}{\partial^{k_0}x_0\partial^{k_1-1}x_1}\left\{\frac{1}{k_0!(k_1-1)!}\left(\frac{x_0^{M+n}}{x_0-x_1}+\frac{x_1^{M+n}}{x_1-x_0}\right)\right\}$$

2) It is true for $0,\ldots,s-1$

3) Proof for s:

$$\begin{aligned}f_s^*(X,M) &= \sum_{i_s=0}^{M}\binom{k_s+i_s-1}{k_s}x_s^{i_s}\sum_{i_{s-1}=0}^{M-i_s}\cdots\sum_{i_1=0}^{M-i_2-\cdots-i_s}\sum_{i_0=0}^{M-i_1-\cdots-i_s} \\ &\binom{k_{s-1}+i_{s-1}-1}{k_{s-1}}x_{s-1}^{i_{s-1}}\cdots\binom{k_1+i_1-1}{k_1}x_1^{i_1}\binom{k_0+i_0}{k_0} = \\ &= \sum_{j=0}^{M}\binom{k_s+i_s-1}{k_s}x_s^{i_s}f_{s-1}^*(X,M-j)\end{aligned}$$

Simplify the summand:

$$\sum_{j=0}^{M}\binom{k_s+j-1}{k_s}x_s^jx_i^{M-j+k_0+\cdots+k_{s-1}} = \frac{1}{k_s!}\sum_{j=0}^{M}(k_s+1)\ldots(k_s+i_s-1)x_s^jx_i^{M-j+k_0+\cdots+k_{s-1}} =$$

$$\frac{\partial^{k_s-1}}{\partial^{k_s-1}x_s}\left\{\frac{1}{(k_s-1)!}\sum_{j=0}^{M}x_s^{j+k_s-1}x_i^{M-j+k_0+\cdots+k_{s-1}}\right\} =$$

$$\frac{\partial^{k_s-1}}{\partial^{k_s-1}x_s}\left\{\frac{1}{(k_s-1)!}\sum_{j=-k_s+1}^{M}x_s^{j+k_s-1}x_i^{M-j+k_0+\cdots+k_{s-1}}\right\} =$$

$$\frac{\partial^{k_s-1}}{\partial^{k_s-1}x_s}\left\{\frac{1}{(k_s-1)!}\left(\sum_{j=-k_s+1}^{M+k_1+\cdots+k_{s-1}} x_s^{j+k_s-1}x_i^{M-j+k_0+\cdots+k_{s-1}} - \sum_{j=M+1}^{M+k_0+\cdots+k_{s-1}} x_s^{j+k_s-1}x_i^{M-j+k_0+\cdots+k_{s-1}}\right)\right\} =$$

$$\frac{\partial^{k_s-1}}{\partial^{k_s-1}x_s}\left\{\frac{1}{(k_s-1)!}\left(\frac{x_s^{M+k_0+\cdots+k_s}-x_i^{M+k_0+\cdots+k_s}}{x_s-x_i} - (x_i^{k_0+\cdots+k_{s-1}-1}x_s^{M+k_s} + \cdots + x_s^{M+n-1})\right)\right\} =$$

$$\frac{\partial^{k_s-1}}{\partial^{k_s-1}x_s}\left\{\frac{1}{(k_s-1)!}\left(- \frac{x_s^{M+n}}{x_i-x_s} + \frac{x_i^{M+n}}{x_i-x_s} - (x_i^{k_0+\cdots+k_{s-1}-1}x_s^{M+k_s} + \cdots + x_s^{M+n-1})\right)\right\} \qquad (*)$$

$(1 \le i \le s-1)$

Summarise the last expression for i from 1 to $s-1$ and simplify it:

From the Lemma:

$0 \le p \le s-1$:

$$\frac{x_0^p}{(x_0-x_1)\dots(x_0-x_{s-1})} + \frac{x_1^p}{(x_1-x_0)\dots(x_1-x_{s-1})} + \cdots + \frac{x_{s-1}^p}{(x_{s-1}-x_0)\dots(x_{s-1}-x_{s-2})} = 0$$

Apply the formula from the consecutive representation in reverse order:

$s \le p \le k_0+\cdots+k_{s-1}-1$:

$$\frac{x_0^p}{(x_0-x_1)\dots(x_0-x_{s-1})} + \frac{x_1^p}{(x_1-x_0)\dots(x_1-x_{s-1})} + \cdots + \frac{x_{s-1}^p}{(x_{s-1}-x_0)\dots(x_{s-1}-x_{s-2})} =$$

$$= \sum_{i_0=0}^{p-s+1}\sum_{i_1=0}^{p-s+1-i_0}\cdots\sum_{i_{s-1}=0}^{p-s+1-i_0-\cdots-i_{s-2}} x_0^{i_0}x_1^{i_1}\dots x_{s-1}^{i_{s-1}}$$

$$\frac{\partial^{n-s}}{\partial^{k_0}x_0\partial^{k_1-1}x_1\dots\partial^{k_{s-1}-1}x_{s-1}}$$

$$\left\{\sum_{i_0=0}^{p-s+1}\sum_{i_1=0}^{p-s+1-i_0}\cdots\sum_{i_{s-1}=0}^{p-s+1-i_0-\cdots-i_{s-2}} x_0^{i_0}x_1^{i_1}\dots x_{s-1}^{i_{s-1}}\right\} = 0$$

As all the summands have the total power not more than

$p-s+1 \le k_0+\cdots+k_{s-1}-1-s+1 = k_0+\cdots+k_{s-1}-s$,

and the order of the partial derivative is

$k_0+(k_1-1)+\cdots+(k_{s-1}-1) = k_0+\cdots+k_s-s+1$,

i.e. at least one less the total power of each summand.

Hence, while summaring in (*) the derivative of sum the following summands will be equal to 0

$$\frac{\partial^{n-s}}{\partial^{k_0}x_0\partial^{k_1-1}x_1\dots\partial^{k_{s-1}-1}x_{s-1}}$$

$$\left\{\sum_{i=0}^{s-1}\frac{x_i^{k_0+\cdots+k_{s-1}-1}x_s^{M+k_s}+\cdots+x_s^{M+n-1}}{(x_i-x_0)\dots(x_i-x_{s-1})} = 0\right\} \qquad 1 \le i \le s-1$$

From the Lemma:

$$\frac{1}{(x_0-x_1)\dots(x_0-x_{s-1})(x_0-x_s)}+\dots+\frac{1}{(x_{s-1}-x_0)\dots(x_{s-1}-x_{s-2})(x_{s-1}-x_s)}=\frac{1}{(x_s-x_0)\dots(x_s-x_{s-1})}$$

Finally obtain:

$$\frac{1}{C}=f_s^*(X,M)=\frac{\partial^{n-s}}{\partial^{k_0}x_0\partial^{k_1-1}x_1\dots\partial^{k_s-1}x_s}\left\{\frac{1}{k_0!(k_1-1)!\dots(k_s-1)!}\left(\frac{x_0^{M+n}}{(x_0-x_1)\dots(x_0-x_s)}+\dots+\frac{x_s^{M+n}}{(x_s-x_0)\dots(x_s-x_{s-1})}\right)\right\}$$

5 Comparison between consecutive and linked list presentations

In previous sections we obtain the formulas which express the average part of time which the system is situated in the state of "reset tail". In this section we will compare consecutive and linked list presentations. Our results were obtained when $m\to\infty$, but they are correct in prelimit form when the size of memory is rather small about 10-20 units. To check results we used system of vector algebra maxima.

We will distinguish several cases of dependences between probabilities:

1. $p_1>q_1$ and $\dfrac{p_1}{q_1}>\dfrac{p_i}{q_i}$ for $i=2,\dots,n$.

Consecutive implementation:

$$\lim_{m\to\infty}\frac{q_i-p_i}{\left(\dfrac{q_i}{p_i}\right)^{k_i+1}-1}=\begin{cases}p_i-q_i, & p_i>q_i\\ 0, & p_i<q_i\end{cases}$$

$$\lim_{m\to\infty}\frac{p_i}{k_i+1}=0$$

Hence, $P_N^*\to\sum\limits_{i=1}^{n}\max(p_i-q_i,0)$

Linked list implementation:

$$P_l^*=(p_1+\dots+p_n)\frac{\sum\limits_{i=1}^{n}\dfrac{x_i^{M+n-1}}{\prod\limits_{\substack{j=1\\j\neq i}}^{n}(x_i-x_j)}}{\sum\limits_{i=1}^{n}\dfrac{x_i^{M+n}}{(x_i-1)\prod\limits_{\substack{j=1\\j\neq i}}^{n}(x_i-x_j)}+\dfrac{1}{\prod\limits_{j=1}^{n}(1-x_j)}}\to$$

$$(p_1+\dots+p_n)\left(\frac{x_i-1}{x_i}\right)=(p_1+\dots+p_n)\left(1-\frac{q_1}{p_1}\right)$$

$p_i(1-\frac{q_1}{p_1}) > 0 \quad 2 \le i \le n$, because $p_1 > q_1$
$p_i(1-\frac{q_1}{p_1}) = p_i - p_i\frac{q_1}{p_1} > p_i - q_i$
Hence, $p_i(1-\frac{q_1}{p_1}) > \max(p_i - q_i, 0)$
Summarise the last inequality for i from 2 to n and add $p_i - q_i$:
$\lim\limits_{m\to\infty} P_c^* = \sum\limits_{i=1}^{n} \max(p_i - q_i, 0) < (p_1 + \cdots + p_n)\left(1 - \frac{q_1}{p_1}\right) = \lim\limits_{m\to\infty} P_l^*$
$P_c^* < P_l^*$ even when the size of memory is rather small.

2. $p_i = q_i = \dfrac{1}{2n}$ for $i = 1, \ldots, n$.

$P_c^* = \sum\limits_{i=1}^{n} \dfrac{p_i}{k_i+1} = \sum\limits_{i=1}^{n} \dfrac{\frac{1}{2n}}{\frac{m}{n}+1} = \dfrac{n}{m+n}$ $P_c^* \le \dfrac{n}{m+n}$
$P_l^* = \dfrac{n}{M+n}$
$P_c^* < P_l^*$

3. $p_i < q_i$ for $i = 1, \ldots, n$. and $\dfrac{p_1}{q_1} > \dfrac{p_i}{q_i}$

In [1] we found the optimal partition of memory in the case of consecutive presentation using the method of dynamic programming. All the queues will spend roughly the same part of time in the state of "reset tail". I.e.

$$\frac{q_i - p_i}{\left(\frac{q_i}{p_i}\right)^{k_i+1} - 1} \approx \frac{q_j - p_j}{\left(\frac{q_j}{p_j}\right)^{k_j+1} - 1} \qquad \forall i \ne j$$

Using this equations and condition $k_1 + \cdots + k_n$ we can find the roughly values of variables $k_1, \ldots, k_n$:

$$P_c^* \approx \frac{1}{\exp\left(\dfrac{m-n}{\sum\limits_{i=1}^{n} \frac{1}{\log(\frac{q_i}{p_i})}} - \log n\right)} = \mathrm{O}\left(\exp\left(-\frac{m}{\sum\limits_{i=1}^{n} \frac{1}{\log(\frac{q_i}{p_i})}}\right)\right)$$

The behaviour of the function $P_l^*(M)$ will be determined by the value

$$\sum_{i=1}^{n} \frac{x_i^{M+n-1}}{\prod\limits_{\substack{j=1 \\ j\ne i}}^{n} (x_i - x_j)}$$

because in the denominator $\sum\limits_{i=1}^{n} \dfrac{x_i^{M+n}}{(x_i-1)\prod\limits_{\substack{j=1 \\ j\ne i}}^{n} (x_i - x_j)} \to 0$

and value $\dfrac{1}{\prod\limits_{j=1}^{n}(1-x_j)}$ is a constant. Thus

$$P_l^* = \mathrm{O}\Big(\Big(\frac{q_1}{p_1}\Big)^M\Big) = \mathrm{O}\Big(\exp\Big(-m(1-\frac{1}{l})\log\frac{q_1}{p_1}\Big)\Big)$$

In this case the part of time which the system is situated in the state of "reset tail" exponentially tends to 0 in both cases of presentation. To choose best of them we need to compare the exponents and choose minimal of them.

Bibliography

[1] E. A. Aksenova, A. V. Drac, A. V. Sokolov. *The optimel control of n FIFO-queues for infinit time. Information and Conrot Systems. p. 46-54. 2009. In russian.*

[2] D. E. Knuth. *The art of computer programming. Vol. Addison-Wesley, Reading, MA, 2001.*

[3] M. Tolley, G. Louchard, R. Schott. *Random walks, heat equation and distributed algorithms. J. Comput. Appl. Math. p. 243-274. 1994.*

[4] J. Riordan. *Introduction to Combinatorial Analysis. Dover Publications, 1958.*

[5] J. L. Snell, J. G. Kemeny. *Finite Markov Chains. Van Nostrand, Princeton, New Jersey, 1960.*

Optimizing performance in heavy-tailed system: a case study

Alexander S. Rumyantsev[†], Luybov V. Potakhina[‡]

[†]Institute of Applied Mathematical Research, Karelian Research Centre, RAS

[‡]Petrozavodsk State University

E-mail: ar0@mainpgu.karelia.ru, lpotahina@gmail.com

Abstract

This article is devoted to certain aspects of optimizing the performance of a single-server queue with heavy-tailed service time distribution, three case-studies are presented: choosing an optimal queueing policy, switching to multi-server system, choosing a task assignment policy. We discuss main statistical properties of heavy-tailed distributions and some practice examples, which illustrate evidence of heavy tails in systems. The main goal of the work is to find and describe ways, which can minimize the effect of heavy tails on system performance.

1 Introduction

The most recent advances in computer systems design show the trend of switching to multiple cores and many cores from the former single-core machine types [1, 2]. While taking into account the thermal effects and frequency limits, one may also consider the differences in the workload processes of this two types of machines. One of the goals of this work is to illustrate the advantages of using multi-core architecture instead of a single-core one.

In modern computer systems analysis one of the challenges one could face are the heavy-tailed distributions of stochastic processes involved. Recent research has highlighted that this kind of distributions can describe a behavior of some real network process better than others (e.g. exponential distribution). One could find empirical evidence and some discussion about the subject in works [3, 4, 5] and the most recent work [6]. Among the most used distributions in practice is Pareto distribution with tail

$$P(X > x) = x^{-\alpha}, \quad x > 1,\ \alpha > 1.$$

Consider some key properties of heavy tailed distributions:

- the so-called Pareto law, or *mass-count disparity*: provided heavy-tailed service time, a small fraction of tasks requires a large part of capacity and vice-versa, vast majority of small tasks requires only a little bandwidth (CPU time, traffic etc.).
- heavy-tailed distributed random values may have infinite variance (and moreover, if $\alpha < 1$, even infinite mean).
- under heavy-tailed service time, a jobs waiting time process has *burstiness*: the process has some bursts, which are much bigger than "normal" values of the process. Besides, an infinite second moment of service time leads to an infinite mean waiting time in a classical single-server system.

This effects have negative influence on system characteristics: performance, reliability, quality of service. In this paper we discuss some practical recommendations how to minimize this effect.

This paper is organized as follows. Section 2 describes the influence of a service discipline on a single-server system delays, section 3 highlights the differences in workloads of a single-server and multi-server systems. Section 4 evaluates the effect of a task assignment policy in multi-server queue, and the conclusion goes in section 5.

2 Choosing a service discipline

For a non-negative random variable X we introduce the distribution function $F(x) = P(X \leqslant x),\ x \geqslant 0$, the tail distribution $\overline{F}(x) = P(X > x)$ and the equilibrium distribution (stationary residual lifetime distribution)

$$F_r(x) = \frac{1}{EX} \int_0^x \overline{F}(y)dy, \quad x \geqslant 0.$$

Consider a M/G/1 system with heavy-tailed (Pareto) distribution of a typical task service time S:

$$\overline{B}(x) := P(S > x) = x^{-\alpha}, \quad x > 1,\ \alpha > 1.$$

One should note a well-known result connecting the moment properties of random variable X and it's equilibrium version X_r:

$$EX^{\alpha} < \infty \text{ iff } EX_r^{\alpha-1} < \infty.$$

Basically this means that a random variable X_r having residual lifetime distribution has moment properties one moment worse than X. Due to the

basic properties of Pareto random variables (among other *regularly varying* r.v.), one may conclude that asymptotically

$$\frac{P(X + X_r > x)}{P(X_r > x)} \to 1, \quad x \to \infty. \tag{2.1}$$

Here we highlight recent theoretic results of the tail asymptotics of the typical waiting time W and (which is needed in case of a preemptive discipline) sojourn time V distributions for six key service disciplines. An interested reader can find the details of the analysis in the work [7].

1. In First Come First Served (FCFS) queue the tasks are served in the order of their arrival. If an arrival finds the server busy, it has to wait at least the residual service time S_r for the task being served to complete it's request. Then, due to the property (2.1) the tail of residual service time dominates. One may prove that waiting time is asymptotically equivalent to residual service time (the details are provided in [7]).

$$P\{W > x\} \sim \frac{\rho}{1-\rho}\overline{B}_r(x), \quad x \to \infty$$

2. If there are n tasks in queue with Processor Sharing (PS) discipline, they are simultaneously served with equal part of server capacity. So, the influence of long tasks on the short tasks sojourn time is limited. Then the tail of sojourn time distribution is described asymptotically as

$$P\{V > x\} \sim \overline{B}\big((1-\rho)x\big), \quad x \to \infty$$

3. For a Last Come First Served Preemptive-Resume (LCFS-PR) discipline, an arriving task is immediately taken into service. However, this service is interrupted when another task arrives, and it is only resumed when all tasks who have arrived after it have left the system. The tail asymptotics of sojourn time in this case is as follows:

$$P\{V > x\} \sim \frac{1}{1-\rho}\overline{B}\big((1-\rho)x\big), \quad x \to \infty$$

4. On the contrary, for the Last Come First Served Non-Preemptive (LCFS-NP) queue, if an arriving task finds a busy server, it doesn't interrupt the service. So, the tail of it's waiting time is determined by the tail of a residual service time:

$$P\{W > x\} \sim \rho\overline{B}_r\big((1-\rho)x\big), \quad x \to \infty$$

5. The Foreground-Background Processor Sharing (FBPS) discipline assigns an equal part of the service capacity to the customers, which have received the least amount of service. The tail of sojourn time is the same as for the PS discipline:

$$P\{V > x\} \sim \overline{B}\big((1-\rho)x\big), \quad x \to \infty$$

6. For the Shortest Remaining Processing Time First (SRPTF) discipline the total service capacity is assigned to the task with shortest remaining processing time. This discipline is preemptive and the tail of sojourn time is the same as for the PS discipline:

$$P\{V > x\} \sim \overline{B}\big((1-\rho)x\big), \quad x \to \infty$$

Summarizing the cases above, one may conclude that for a single-server system the PS, FBPS and SRPTF disciplines have asymptotically the best moment properties for the waiting/sojourn time, while the most often used FIFO discipline is not optimal. Hence, one may consider optimizing the quality of service in such a system by changing the queueing discipline.

3 Choosing a server architecture

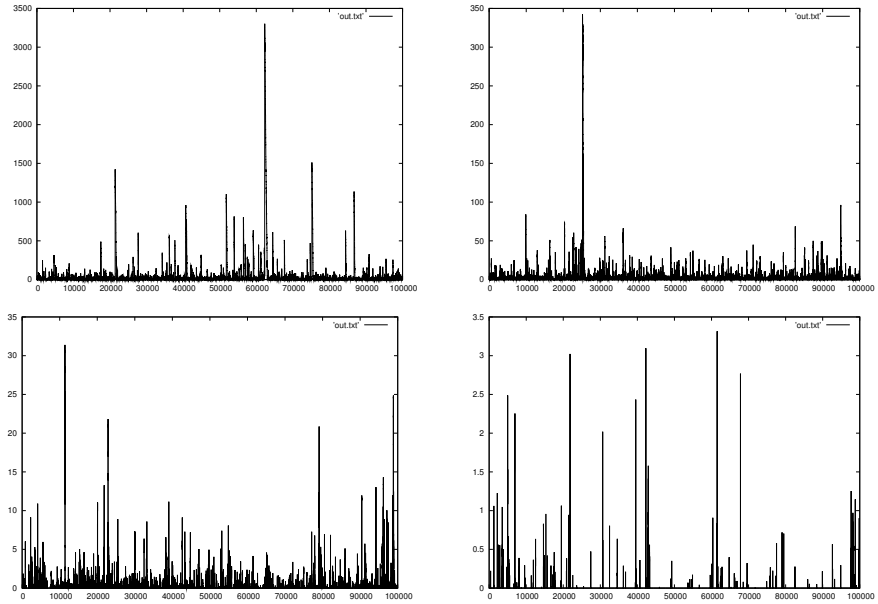
One of the trends in modern computer design is to use the low-cost power-efficient low-frequency multiple cores on a single processor instead of high-frequency systems with single core on chip. As an example one may consider the BlueGene/P systems with a PowerPC 450 cores at 850 Mhz. One of the questions arising in this regard is: which type of architecture is asymptotically best? More exactly, one may compare the moment properties of the waiting times in these two types of systems.

Consider an M/G/1 system with Pareto service times and a fast single server. Then increase the number of servers in a system with a lower speed so as to keep it's performance. (E.g. one server with a 4 GHz core, two servers with 2 GHz cores, four with 1 GHz ones, etc. One could find more details in the work [8].) For the M/G/s system one may find the following result [9]: denote $\rho = ES/ET$ the load of the system (where S and T are typical service and interarrival times respectively). Let $\rho < 1$, then one has the following moment conditions:

$$ES^{\beta} < \infty \text{ provides } EW^{s(\beta-1)} < \infty.$$

For a single server case ($s = 1$) this gives classical result. In general, these results mean that s *slow* servers provide better moment conditions for the mean waiting time than one *fast* server.

Table 1: Simulation results for waiting times in 1-, 2-, 4- and 8-core system (top-left to bottom-right).



In the numerical results below we take $\alpha = 1.5$, the total quantity of tasks is 100 000, service discipline is FIFO, and traffic intensity is $\rho = 0.3 < 1$.

At top-left picture in table 1 one can see waiting times in a simulated M/G/1 system. The maximum burst is near 3 500 time units. It is much bigger than "normal" values of the process. At the top-right the figure depicts waiting times in a system which has 2 cores. These cores are twice slower than core from 1st slide. (The identical tasks on a twice slower core in this system give service times $\hat{S}$ doubled compared to the service times S of the original system, $\hat{S} = 2S$). And one can note, that the maximum value is near 350 time units, that is an order smaller than burst at 1st slide. Bottom-left is connected with a system with 4 cores (each one with a quarter speed of the single-core server), Bottom-right is connected with system with 8 cores.

We may conclude that the increasing of core quantity can reduce mean waiting time in a system and decrease the system workload. Possibly, this is one of the results that motivates the computer system makers to use multi-core processors design.

4 Choosing a task assignment policy

Consider a system with n servers and a single queue for tasks. The queue manager (or dispatcher) should use special policy in order to assign tasks from this queue to servers, each having it's own queue. One may argue which task assignment policy is best applicable to our heavy-tailed system.

A common practical situation is when the task service times are upper and lower bounded. This leads to definition of a truncated Pareto distribution (widely used in modeling), with the density

$$f(x) = \frac{\alpha k^\alpha}{1-(k/p)^\alpha} x^{-\alpha-1}, \ where \ k \leqslant x \leqslant p$$

This distribution has all the moments finite. Nevertheless, as $p \to \infty$, this distribution approaches a standard Pareto one, which may have unbounded moments.

We assume that the dispatcher knows the size S of task. The tasks assigned to each server are served in FCFS (non-preemptive) discipline. Consider 4 key policies (the examples are based on the work [10]):

1. Random: an arriving task is sent to selected server with equal probability $1/n$.

2. Round-Robin: tasks are assigned to servers in cyclical order: i-th task being assigned to server $i \bmod n$.

3. Dynamic: Incoming task is assigned to the server with the smallest amount of remaining work time, which is the sum of the sizes of the tasks in the server's queue plus work remaining on that task currently being served.

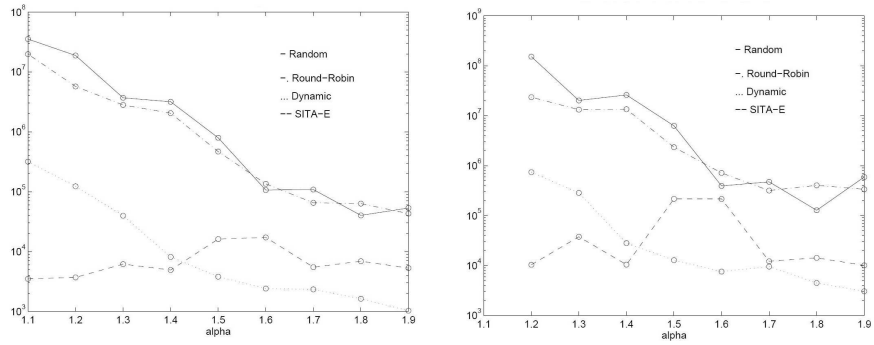
4. Size-based: so-called SITA-E (Size Interval Task Assignment with Equal Load) algorithm. The idea is to define the size range associated with each server such that the total work of each server is the same. Balancing the load this way minimizes the mean waiting time.

   Let the distribution function of task sizes be $B(x)$. Define "cutoff points" $x_i, i = 0..n, x_0 = k, x_n = p$ by the following rule:

   $$\int_{x_0}^{x_1} x dB(x) = ... = \int_{x_{n-1}}^{x_n} x dB(x) = \frac{ES}{n}$$

   and assign to the i-th core all tasks with size in the range $S \in [x_{i-1}, x_i]$ (ties defined arbitrary).

Table 2: Mean waiting time (left) and standard deviation (right) for different task assignment policies



This policies were compared via simulation in the work [10]. We consider values α from 1.1 (high variability) to 1.9 (lower variability). The figures in table 2 show mean waiting time and standard deviation for the described policies as function of parameter α.

As one can see, the mean waiting time in the system with Random and Round-Robin policies are quite similar and larger than the other two. For a large α Dynamic policy shows the best results, but if the variability of service time increases, Dynamic policy shows worse performance. In contrast, the SITA-E behavior remains quite stable, even if α tends to 1, or in other words, if the variability of service times increases.

5 Conclusion

We can conclude that a negative influence of heavy tails on system performance may be reduced in different ways. We have reviewed three practical examples on how one can decrease workload in system with heavy tailed in service times.

6 Acknowledgments

We thank prof. E. V. Morozov for useful comments. This work is partly supported by RFBR, project 07-10-00017a.

Bibliography

[1] J. Parkhurst, J. A. Darringer, B. Grundmann, *From single core to multi-core: preparing for a new exponential.* Proceedings of ICCAD'2006, San Jose, CA, 2006. pp. 67–72.

[2] Md. Tanvir Al Amin, *Multi-core: Adding a New Dimension to Computing.* Arxiv preprint arXiv10113382, 2010.

[3] E. Morozov, M. Pagano, A. Rumyantsev *Heavy-tailed distributions with applications to broadband communication systems.* Proceedings of AMICT'2007, 2008. Vol. 9, pp. 157–174.

[4] G. Samorodnitsky *Long Range Dependence* Foundations and Trends in Stochastic Systems. Vol. 1, No. 3, 2007. pp. 163–257

[5] A. Zwart, *Queueing Systems with Heavy Tails.* Eindhoven : Eindhoven University of Technology, 2001. 227 p.

[6] D. Feitelson, *Workload Modeling for Computer Systems Performance Evaluation.* Draft version. 497 p. URL: http://www.cs.huji.ac.il/ feit/wlmod/wlmod.eps

[7] S. C. Borst, O. J. Boxma, R. Nunez-Queija, *Heavy Tails: The Effect of the Service Discipline.* Proceedings of Performance TOOLS, 2002. pp. 1–30.

[8] E. V. Morozov, A. S. Rumyantsev, *Multi-server models to analyze high performance cluster.* Transactions of Karelian research center, RAS. No. 5, 2011. Pp. 75–85. (in Russian)

[9] A. Scheller-Wolf, *Further delay moment results for FIFO multiserver queues.* Queueing Systems, Vol. 34, 2000. Pp. 387–400.

[10] M. Harchol-Balter, *The Effect of Heavy-Tailed Job Size Distributions on Computer System Design.* In Proc. of ASA-IMS Conf. on Applications of Heavy Tailed Distributions in Economics, 1999.

Some analytical aspects of regenerative simulation of fluid models

Oleg V. Lukashenko†, Ruslana S. Nekrasova†, Evsey V. Morozov†, Michele Pagano‡

†Karelian Research Center RAS

‡University of Pisa

E-mail: {lukashenko-oleg@mail.ru, ruslana.nekrasova@mail.ru, emorozov@karelia.ru, m.pagano@iet.unipi.it}

Abstract

We discuss the estimation of the loss probability in a queueing system with finite buffer. We apply regenerative technique combined with the so-called Delta-method to construct confidence interval for the stationary loss probability. This work is supported by Russian Foundation for Basic research, project No 10-07-00017.

1 Introduction

We consider a single server queue with finite buffer of size b, constant service rate C and input process $A(t) = mt + N(0, t\sigma^2)$, consisting of deterministic linear process mt with positive drift $m > 0$, and Brownian motion $N(0, t\sigma^2)$. The workload process in this system is described by the well-known (discrete time) Lindley recursion:

$$Q_n = \min((Q_{n-1} - C + X_n)^+, b),\ n = 1, 2, \ldots, \tag{1.1}$$

where

$$X_n := A(n+1) - A(n) =_{st} m + N(0, \sigma^2)$$

are the i.i.d increments of the input process at instants $n = 1, 2, ...$ We denote this system as Bi/D/1/b system. A motivation of this model can be found in [3]. Denote by $L_b(T)$, the total lost workload in interval $[0, T]$, that is

$$L_b(T) := \sum_{k=1}^{T} (Q_{k-1} - C + X_k - b)^+,\ T = 1, 2, \ldots$$

The time average loss $¶_\ell(b,T)$ in this system during (discrete-time) interval $[0,T]$, is defined as the ratio of the amount of lost workload and the total arrived workload, during this interval, that is

$$¶_\ell(b,T) := \frac{L_b(T)}{A(T)}. \tag{1.2}$$

Because the buffer is finite, the system is stable and the loss ratio, as $T \to \infty$, converges to a stationary loss probability $¶_\ell(b)$, that is

$$¶_\ell(b) := \lim_{T\to\infty} ¶_\ell(b,T) = \frac{\mathsf{E}(Q+X-C-b)^+}{m}, \tag{1.3}$$

where Q is the stationary workload and X is a generic element of X_n. The following heuristic expression given in [4]

$$¶_\ell(b) \approx \frac{¶_\ell(0)}{¶(Q>0)} ¶(Q>b), \tag{1.4}$$

allows to calculate the loss probability provided there is an explicit formula (or a satisfactory approximation) for the overflow probability $¶(Q>x)$ in the associated infinite buffer system. In our case, it is possible to use the following continious-time approximation (see [5]):

$$¶(Q>x) \approx \exp\left(-2\cdot\frac{C-m}{\sigma}\cdot x\right). \tag{1.5}$$

Moreover, it is easy to calculate $¶_\ell(0)$, namely,

$$\begin{aligned} ¶_\ell(0) &= \frac{\mathsf{E}(X-C)^+}{m} \\ &= \frac{1}{m\sigma\sqrt{2\pi}}\int_c^\infty (x-c)e^{-(x-m)^2/2\sigma^2}dx. \end{aligned} \tag{1.6}$$

Thus results (1.4), (1.5), (1.6) allow to find an approximation of the overflow probability $¶_\ell(b)$ (in the following it will be denoted as $¶_\ell$) in our model.

2 Regenerative approach

In this section, we show how to estimate the steady-state loss probability P_ℓ using the regenerative approach. First we construct regeneration points for the content process. (More details can be found in [3].) Let $\beta_0 = 0$ and

$$\beta_{k+1} = \min\{n > \beta_k : Q_{n-1} > 0,\, Q_n = 0\},\ k \geq 1, \tag{2.1}$$

where, Q_n is defined in (1.1). Denote by L_i and A_i the workload lost and arrived per the ith regeneration cycle, respectively, with the corresponding generic elements L and A. It follows from the regenerative method, that

$$P_\ell = \frac{\mathsf{E}L}{\mathsf{E}A}.$$

To apply the regenerative confidence estimation, we generate i.i.d. replications $L_1, .., L_n$, $A_1, ..., A_n$, to estimate the unknown means $\mathsf{E}L$, $\mathsf{E}A$ and the probability $¶_\ell$ as

$$\widehat{L} := \frac{1}{n}\sum_{i=1}^{n} L_i,\ \widehat{A} := \frac{1}{n}\sum_{i=1}^{n} A_i,\ \widehat{¶}_\ell := \frac{\widehat{L}}{\widehat{A}}, \tag{2.2}$$

respectively. Using Delta-method, it is possible to show that

$$\sqrt{n}\left(\widehat{¶}_\ell - ¶_\ell\right) \Rightarrow N(0, \eta^2),\ \ n \to \infty, \tag{2.3}$$

where $\Rightarrow$ stands for weak convergence and

$$\eta^2 = \frac{\mathsf{E}\left[L - A \cdot ¶_\ell\right]^2}{(\mathsf{E}A)^2}.$$

(See [1, 2] for more detail on Delta-method.) In turn, to estimate η^2 we apply standard sample estimation

$$\widehat{\eta}^2 := \frac{\frac{1}{n-1}\sum\limits_{i=1}^{n}(L_i - \widehat{¶}_\ell A_i)^2}{\left(\frac{1}{n}\sum\limits_{i=1}^{n} A_i\right)^2} \tag{2.4}$$

Based on (2.3) one can obtain the following $(1-\gamma/2)\%$ asymptotical confidence interval for $¶_\ell$:

$$\left[\widehat{¶}_\ell - \frac{z_\gamma}{\sqrt{n}},\ \widehat{¶}_\ell + \frac{z_\gamma}{\sqrt{n}}\right], \tag{2.5}$$

where $z_\gamma = \widehat{\eta}\Phi^{-1}\left(\frac{\gamma}{2}\right)$, $\Phi^{-1}(x)$ is the inverse of Laplace function and γ is a given confidence probability.

3 Numerical examples

In this section, we present a few numerical results on confidence estimation of stationary loss probability in the above considered system $Bi/D/1/b$.

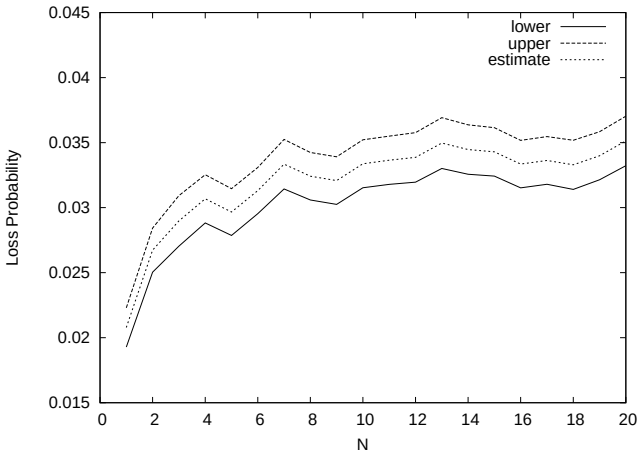


Figure 1: 95% Confidence interval for P_ℓ in Bi/D/1/4

First of all regeneration points are constructed as in (2.1). Then we calculate the confidence interval for the probability ¶$_\ell$ according to expression (2.5).

First we analyze a dependence the simulation results on the *step of digitization* $h := h_N = 1/N$ where $N = 1, 2, \ldots$ Figure 1 shows 95% confidence interval as a function of N for the model with parameters $m = 0.8$, service rate $C = 1$, buffer size $b = 4$ for a fixed simulation length of T time slots and $T = 10^5$. It is seen that confidence interval width is rather insensitive to the selection of the concrete value of step digitization h in a wide range of values of N. This remark may be useful to simplify simulation procedure and, in particular, to save simulation time.

Figure 2 compares the simulation results (based on the regenerative approach described above) with the approximation (1.4), where the following parameters are used: $C = 1$; $m = 0.7$; $T = 10^6$, and the confidence probability is $1 - \gamma = 0.95$.

Finally, we study the dependence of the simulation results on the choice of regeneration points. Namely, we form the *subsequences* of regeneration points $\{\beta_k^s\}$ of (2.1) for arbitrary (fixed) s as $\beta_i^s = \beta_{si}$ where $s = 1, 2, \ldots$ and $i = 0, 1, ...$ Figure 3 shows the 95% confidence interval width (for the loss probability ¶$_\ell$) vs. the parameter s. The following parameters are used in simulation: $b = 4$; $C = 1$; $m = 0.8$; $T = 10^5$. Again, simulation demonstrates an insensitivity to the choice of the parameter s, and, in our opinion, it can be exploit to speed-up estimation.

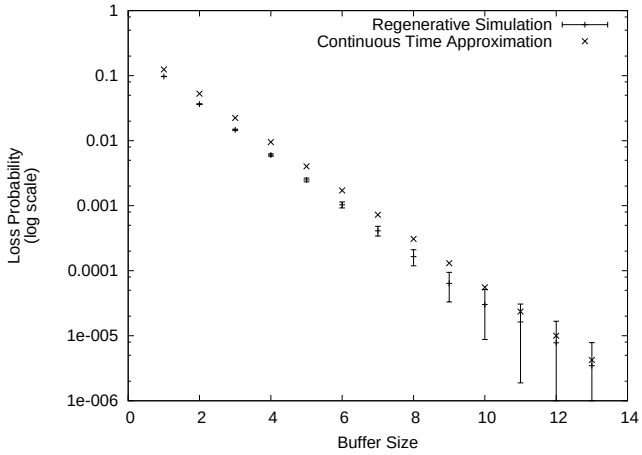


Figure 2: Estimate of P_ℓ in Bi/D/1/b: regenerative method vs. approximation (1.4)

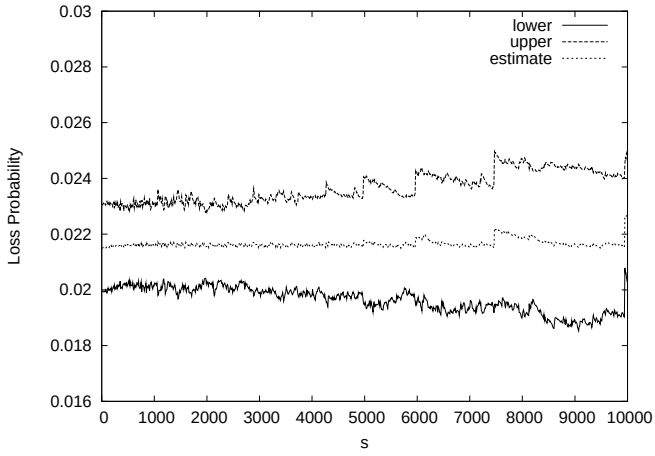


Figure 3: Dependence of the confidence interval width on the choice of subsequences of regeneration points

4 Conclusion

The estimation of the stationary loss probability in a single-server fluid queue with a Gaussian input using the regeneration approach is considered. A known approximation of the loss probability via the overflow probability is used to verify an accuracy the estimation based on the regenerative simulation. Some numerical results are presented.

Bibliography

[1] *Asmussen S.* Applied Probability and Queues, Springer, 2002.

[2] *Asmussen S., Glynn P.* Stochactic Simulation: algorithms and analysis. Springer, 2007.

[3] *Goricheva R. S., Lukashenko O. V., Morozov E. V., Pagano M.* Regenerative analysis of a finite buffer fluid queue, Proceedings of ICUMT 2010 (electronic publication).

[4] *Kim H. S., Shroff N. B.* Loss Probability Calculations and Asymptotic Analysis for Finite Buffer Multiplexers. IEEE/ACM Transactions on Networking, 2001, v. 9, 755–768.

[5] *Takacs L.* Combinatorial Methods in the Theory os Stochastic Processes, John Wiley&Sons, 1967.

Simulation of the fluid system with long-range dependent input

Oleg V. Lukashenko[†], Mikhail Nasadkin[‡]

[†] Karelian Research Center RAS

[‡] Petrozavodsk State University

E-mail: {lukashenko-oleg@mail.ru, mnasad@petrsu.ru}

Abstract

We discuss the application of the simulation to estimate the loss or overflow probability in a queuing system with a finite or infinite buffer, which is fed by a Gaussian input. We mainly consider fractional Brownian input (FBI) because it satisfies some properties such as self-similarity and long-range dependence that network traffic sometimes obeys. This work is supported by Russian Foundation for Basic research, project No 10-07-00017.

1 Introduction

We consider the so-called fluid queue with a constant service rate and a Gaussian input process. The work focuses on the estimation of the overflow probability $P(Q > b)$, that is the probability that the workload process exceed a threshold level b (in the infinite buffer case) and the loss probability P_ℓ, or the buffer overflow probability (in the finite buffer queue). Such probabilities can be useful for the QoS analysis of telecommunication systems. At the present time, for the queues with general Gaussian input (in particular, for the most important models with fractional Brownian input (FBI)) there are no explicit results, and only some asymptotics for the overflow probability are found. Thus, in general, only simulation remains an available and the most adequate way to estimate the required probability.

Finite buffer systems, being more realistic models of real-life networks, are more difficult to be analyzed, and by this reason explicit (and asymptotic) expressions for P_ℓ in such systems are much less available.

2 Queue with long-range dependent input

In this section we describe a single server queue with deterministic service rate C. Denote by $A(t)$ the amount of data (input traffic) arrived into a communication node within time interval $[0, t]$, $t \geq 0$. Let consider the following definition of the input [4]:

$$A(t) = mt + \sigma B_H(t)$$

where $\{B_H(t),\, t \geq 0,\}$ is a fractional Brownian motion (fBm), which describes random fluctuations of the input around its linearly increasing mean, $H \in (1/2,\, 1)$, $\sigma > 0$ is some scaling parameter. Let $r = C - m$, to guarantee stability of such a system we assume that $r > 0$.

Firstly consider system with infinite buffer (FBI/D/1). Denote by $B_H^*(n) = B_H(n+1) - B_H(n)$ the increments of fBm. Accorging to Lindley recursion (in discrete time) we have following expression for workload (queue content):

$$Q(t) = \left(Q(t-1) - r + \sigma B_H^*(t)\right)^+,\ t = 1, 2, \ldots \tag{2.1}$$

where $(x)^+ = \max(0, x)$. The overflow probability (queue tail probability) is defined as the amount of time the queue spends above some level b divided by the total time:

$$¶(Q > b) = \lim_{T\to\infty} \frac{1}{T} \sum_{k=1}^{T} I(Q(k) > b), \tag{2.2}$$

where I means indicator. Recursion (2.1) can be extended to the queue content $Q_b(t)$ in a system with buffer size $b < \infty$ (FBI/D/1/b) as follows:

$$Q_b(t) = \min\left(\left(Q_b(t-1) - r + \sigma B_H^*(t)\right)^+, b\right),\ t = 1, 2, \ldots. \tag{2.3}$$

The time average loss $¶_\ell(b, T)$ in this system during (discrete-time) interval $[0, T]$, is naturally defined as the ratio of the amount of loss to the total amount of input during this interval.

$$¶_\ell(b, T) = \frac{\sum_{k=1}^{T} (Q_b(k-1) - r + \sigma B_H^*(k) - b)^+}{A(T)}.$$

Under stability assumption, one can expect that stationary loss ratio converges to stationary loss probability $¶_\ell(b)$, that is

$$¶_\ell(b) = \lim_{T\to\infty} ¶_\ell(b, T) = \frac{\mathsf{E}(Q_{n-1} + \sigma B_H^*(k) - r - b)^+}{m}. \tag{2.4}$$

Unfortunatelly, so far there are no explicit results for (2.2), (2.4) in case FBI. But there are some asymptotic results. For example, for sufficiently large b there is continious-time approximation

$$¶(Q > b) \approx \exp\left(-\theta b^{2-2H}\right), \tag{2.5}$$

where constant θ depends on initial parameters as:

$$\theta = \frac{r^{2H}}{2\sigma^2} \cdot \frac{1}{H^{2H}(1-H)^{2(1-H)}}.$$

Actually, expression (2.5) means that for sufficiently large b queue tail distribution $¶(Q > b)$ is Weibullian.

Kim and Shroff have established the relationship between overflow and loss probability [2]:

$$\log ¶(Q > b) - \log ¶_\ell(b) = O(\log b), \;\; as \;\; b \to \infty. \tag{2.6}$$

Let rewrite (2.6) as

$$¶_\ell(b) = ¶(Q > b) b^{O(1)}, \;\; b \to \infty \tag{2.7}$$

Using the last expression, it is possible to derive approximation for the loss probability $¶_\ell(b)$ via the corresponding overflow probability. But existence of unknown function $O(1)$ in (2.7) makes it difficult in practice.

Let conider the boundary case $H = 1$. Obviously, $B_1(t) = t \cdot N(0,1)$ is a random line (the increments are the same). So there are two alternatives: first, when the system can complite all work and there are no losses; second, when the system can not complete work and after several steps the buffer is full and all subsequent work will be lost. It follows from above given arguments that there is explicit formula for $¶_\ell(b)$:

$$\begin{aligned} ¶_\ell(b) &= \frac{\mathsf{E}(N(0,\sigma^2) - r)^+}{m} \\ &= \frac{1}{m\sigma\sqrt{2\pi}} \int_r^\infty (x-r) e^{-x^2/2\sigma^2} dx. \end{aligned} \tag{2.8}$$

3 Simulation

However, all given above results are asymptotic. So simulation is often the only way to calculate the overflow/loss probability for small or moderate values of b.

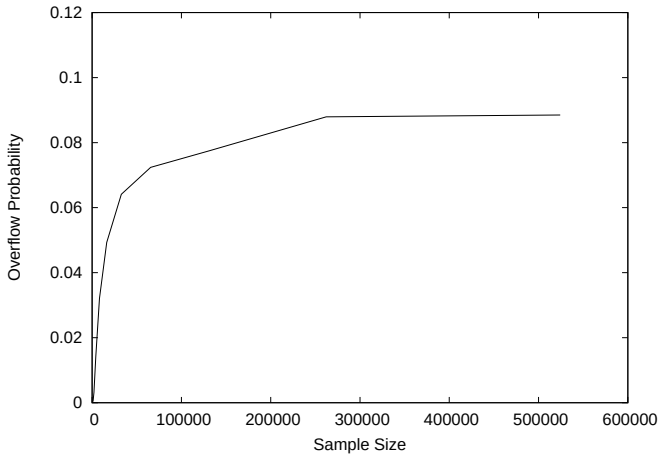


Figure 1: Stabilization of $\widehat{¶}_{500}$

let describe in brief the procedure of simulation. Firstly we should generate sufficiently large number N of fBm traces (sample paths). For generating fBm traces we use RMD-method (Random Midpoint Displacement) [3]. Each sample path should include sufficienly large number of observations T. After that based on (2.1), (2.3) we calculate sample paths $\{\widehat{Q}^i(k),\ k = 1, ..., T\}$, $\{\widehat{Q}^i_b(k),\ k = 1, ..., T\}$, $i = 1, ..., N$. Estimate for $P(Q > b)$ has following form:

$$\widehat{¶}_b = \frac{1}{T}\sum_{k=1}^{T} I\left(\widehat{Q}(k) > b\right).$$

Let L_n be the total amount of losses for the sample path $\widehat{Q}^n_b$. Then the estimate of $¶_\ell(b)$ is defined as folllows:

$$\widehat{¶}_\ell(b) \approx \frac{\mathsf{E}L}{mT}, \quad \mathsf{E}L = \frac{1}{N}\sum_{n=1}^{N} L_n$$

Figure 2 shows the dependence of the estimate of the overflow probability on sample path size T. Note that the sample path size must be sufficiently large to eliminate the influence of so-called initial period. The following parameters are used: $C = 1$; $m = 0.8$; $b = 500$; $N = 1000$.

Figure 2 compares the simulation results for the overflow probability in system $FBI/D/1$ with the approximation (2.5). The following parameters are used: $C = 1$; $m = 0.8$; $T = 2^{16}$, $N = 1000$. As expected from the

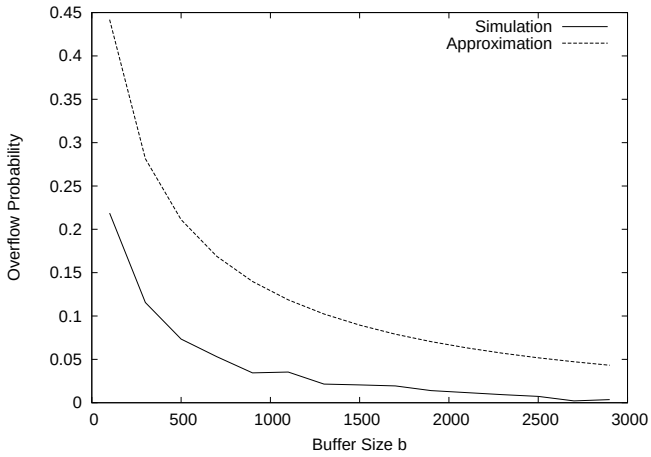


Figure 2: Simulation vs. approximation (2.5)

theory the difference between the simulation results and approximation decreases when the buffer size grows.

Finally, figure 3 shows the simulation results of $¶_\ell$ for the finite buffer system $FBi/D/1/b$ with parameter $H = 0.99$, According to (2.5), for H close to 1 the loss probability it is easy to calculate approximate value of $¶_\ell$. For given parameters $C = 1$; $m = 0.8$ we have $¶_\ell \approx 0.297$ and it does not depend on buffer size b.

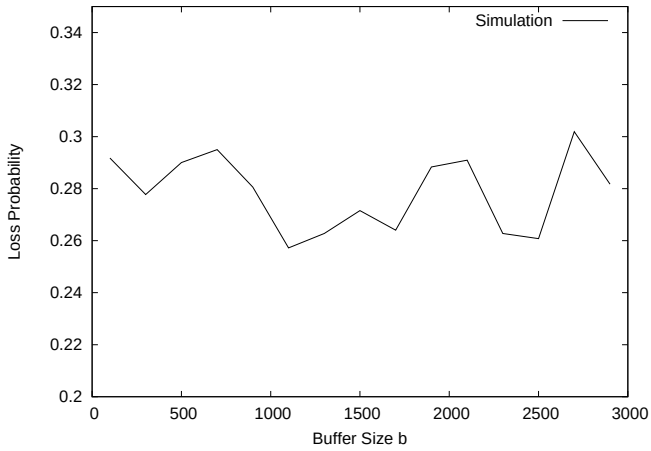


Figure 3: Estimate of P_ℓ for $H = 0.99$

4 Conclusion

The results of our study can be summarised as follows:

The procedure of estimation of the overflow and loss probability is discussed. We have compared the known approximation and the estimation using simulation technique. Numerical results are consistent with theoretical formulas.

Bibliography

[1] *Duffield N., O'Connell N.* Large deviations and overflow probabilities for the general single server queue, with applications // Proceedings of the Cambridge Philosophical Society. 1995. Vol. 118. P. 363–374.

[2] *Kim H. S., Shroff N. B.* Loss Probability Calculations and Asymptotic Analysis for Finite Buffer Multiplexers. IEEE/ACM Transactions on Networking, 2001, v. 9, 755–768.

[3] *Lau W-C., Erramilli A., Wang J. L., Willinger W.* Self-similar traffic generation: the random midpoint displacement algorithm and its properties. Proceedings of ICC '95. 1995.

[4] *Norros I.* A storage model with self-similar input, Queueing Systems, 1994. v. 16, 387–396.

How to avoid plagiarism?

Tiina Niklander

Department of Computer Science, University of Helsinki

P.O.Box 68, FI-00014 University of Helsinki, Finland

E-mail: {tiina.niklander@cs.helsinki.fi

Abstract

Plagiarism and its avoidance have gained for attention in educating students. A lot of institutes define it using examples. General agreement is that plain copying of existing material is definitely plagiarism. The current information flow in the Internet and the use of computers in the writing process makes it too easy for students to avoid the actual learning-by-writing and just to perform cut-and-paste for their essay or report. At the same time the information flow makes it almost impossible for teachers to detect all of these flaws. Automatic detection tools have been developed to help teachers in this process. However, the detection is just one part of the story. It is much more important that the authors (students as well as researchers) do not copy from others, but say it using their own voice and words. We should also remember that some students have copied material even before the current technology.

1 Introduction

Everybody agrees that plagiarism in general is unethical. The key problem with plagiarism is that it is not black and white. There is a huge gray area, in which one person would consider the text fragment as plagiarism while another one could interpret the issue just as missing reference or poorly formulated sentences.

Plagiarism is defined as "the use of another's thoughts, or work, without acknowledgement or permission. In plagiarism, one author takes another's idea and presents it as his/her own" [9]. Similar definitions can be found from all papers discussing this issue. For example, L. Bilic-Zulle et.al. [1] give the definition as "misappropriation of another person's ideas, methods, results or words, i.e. using the intellectual property of another person without giving appropriate credit".

The word plagiarism is clearly latin based, but there are at least two latin words, that have been given as the origin. Gu and Brooks [3] claim that the origin word is plagiaries, which means "the theft of words as well as slaves". On the other hand Naveed Imran [5] gives the word plagium as the latin origin. According to him the meaning of plagium is kidnapping. Either way, both words contain the idea of taking something that is not legally mine.

The modern definitions include the idea that the usage is OK if you have permission or you give the credits. For a student this might look like that the citations is all what the teachers require. While it is the beginning it is not enough. This paper will mainly discuss about the tools that will help teachers in this first step. The proper writing skills would be topic of its own.

2 Aspects of plagiarism

Naveed Imran [5] explains the different aspects of plagiarism using taxonomy. He has divided the aspects into three major groups. The method covers aspects that deal with the actual implementation. The cast covers all kind of forms for plagiarism. Finally, the purpose explains the different motivations behind the plagiarism. The purpose is simply either intentional or unintentional. Naturally the intentional behaviour is more problematic from the ethical perspective.

The methods used in plagiarism start with the most straightforward copy-and-paste. It is the simplest to do and easiest to detect, because no words have been changed. This is often due time limits or laziness, or lack of writing skills. It is always an easy way out.

The inappropriate paraphrasing is not as easy to detect as copy-and-paste, because in the paraphrasing some of the original wordings have been changed. The key problem here is the amount of partially copied text and the lack of the author's own voice. A lot of scientific papers use paraphrasing properly in small scale with quotation marks, if direct copying, and citation information.

Omitting quotation marks and/or citations, automatically causes otherwise properly paraphrased text to be classified as plagiarism. Had the author marked the direct loans and given the credits to the original material, it would have been accepted. Forgetting the citation is undistinguishable from the purposeful omission and both are thus considered as the missing was purposeful. Worse case would be faked citations, where the cited article does not even contain the material, citation was just added to mislead

the reader.

Plagiarism can even cover the stealing of ideas [5]. In such a case the text is presented as if the idea was the author's own and there is no citation available. Even when there are no textual similarities, but the original source of the idea has not been given credits, the author is guilty of improper behaviour and violates the ethics of academic writing. This could naturally happen with missing citations, but often the author has not even thought about adding the citation there.

3 Copy detection tools

The automated detections tools should be able to find both copy-and-paste and paraphrasing kinds of plagiarism. If they are capable of doing it, they would find most missing citations also.

According to [8] there were several tools available already in 2007. The paper compares tools, like Eve2, CopyCatchGold, WordCheck, Glatt, Moss, JPlag, with the de facto standard Turnitin [1]. However, in 2007, the detection tools were not that known and the Turnitin did not have such a strong position as it has today with its easy-to-use web-based user interface. In 2002 it was claimed [4] that "the available software tends to come and go: new software and websites surface and then disappear".

Most of the tools worked with plain text and tried to estimate the similarity between the submitted text and some other texts. For the comparison they used different metrics, such as N-gram, Euclidean distance, Jaccard measure. A nice collection of different metrics is presented in [8].

Most tools simply compare the document with its database. They do not try to analyse the content of the text, but just do some sort of character-based comparison. To catch also translations, the tool must be able to support some kind of semantical matching in addition to plain text comparison. This would be an issue, if and when a Finnish university wants to systematically detect the possible plagiarism in the bachelor thesis, which are written in Finnish using mainly English references.

4 Turnitin

Turnitin has gained the position of de facto standard over the years. It uses its own database to store the texts to be compared. It has also been criticized about the coverage and cheating possibilities. Kaner and Fiedler

[1] http://www.turnitin.com

[7] in 2008 claimed that the tool was missing articles from ACM and IEEE. If that is still true, the usability of the tool for computer science is much reduced.

Turnitin is a web-based tool, which has its own database of articles and pages. All submitted papers are compared against the existing material in the database. The tool works relatively fast and the comparation result is available for the teacher within minutes after the submission. According to [6] "the software looks for matches of strings of eight to ten words", while "ignoring the commonly used words". Basically the tool counts as similarity a sequence of identical words in identical order. The words do not have to be in consecutive sequence, there can be some additional words in between.

Turnitin works nicely when used exactly as the company has planned. The tool has been designed for identical submission deadlines for the whole class. It cannot handle nicely a situation where students have individual deadlines, and the others should be able to see the submitted papers before their own submission. Our department uses this model in some seminars, where students give their presentations one by one during the weeks. They must submit their paper a give time before their individual presentation. The other participants are expected to read and comment the paper before the presentation.

Using Turnitin through the web-interface is relatively easy. Teacher creates a course and the tasks for the students. She also adds students to the course and the students submit their papers themselves. There is also a possibility for teacher submission, but the tool is designed for student submission.

It is announced to be possible to integrate the Turnitin with course platforms like blackboard or moodle. From a teacher's perspective the integration may mean giving up some of the features that are available through the web-interface while gaining the more familiar environment of the platform. Integrated version, the main interface is the course platform and Turnitin is used only to check the originality of the submitted papers.

The author tried the web-interface based service in two separate courses. One of them was a seminar where students wrote their papers in English. The other one was a Bachelor thesis writing group, where the papers were written in Finnish. With Finnish papers, the main benefit came from the possibility of doing electronic peer reviews of the submitted papers by the students.

The tool shows the results of the originality analyses using traffic lights (blue, green, yellow, orange, red) with similarity percentage. The percent-

age tells how much of the report matches with existing material. The tool compares the submitted paper with its own database. The database contains articles collected from web and directly from publishers.

For the author, best feature of the tool is the visual report of the similarities. The tool shows the submitted student paper with colour-marked similarity locations. Each colour represents different original paper. The simple similarity percentage, as such, does not give any information about what kind of text is found to be similar counterpart in the tool's database stored papers.

Because the similarity percentage just shows how much of the student paper has identical short word sequences with other papers, it should not be used automatically by bureaucrats [3] to classify papers as copied vs. not copied. A native speaker is able to use rephrasing easier to avoid the detection of identical word sequences. A non-native speaker has smaller vocabulary, which makes e.g. using synonyms in rephrasing more difficult for them.

The low-similarity-percentage papers indicated with blue or green colour may still contain parts that fulfil the plagiarism definition. There can be one or two paragraphs that are identical with an other paper in the tool's database. It is also possible to have a high-percentage student paper with yellow or orange that does not fulfil the plagiarism definitions. Such a paper may contain directly copied formulas and definitions with proper citations. It is not custom to use quotation marks with formulas. Of course such a paper might not have high quality because the amount of student's original text is low. Usually the red-marked papers contain too much copied material in all cases.

5 Avoiding unintentional plagiarism

Most of the plagiarism in the student papers is not fully intentional, but could be due to the missing skills of students. To solve such issues teaching is more important than punishment. According to Smith and Wren [9] "avoiding plagiarism does not need to be difficult or require an in-depth knowledge of copyright law". Very simple mechanisms are enough to avoid unintentional plagiarism. It is enough to learn proper referencing and paraphrasing techniques. Naturally they mean that the writer (student or researcher) should summarize the referenced article content in own voice and mark it with citation. To be able to do that she has to actually understand the source information [5]. If the student does not understand what she is writing the direct copying might be the only (but wrong) solution within

the deadline.

When students were asked [2] about explanations to plagiarism, they had quite normal explanations from bad time management and simple opportunity to uninterested teachers and eminently theoretical subject. The time management included explanations like, too many assignments to be handed in short time or personal shortcomings in preparation. A lot of these can be solved by the institutions, when these issues gain the attention of the faculty and teachers.

6 Conclusion

The plagiarism avoidance can be done with just common sense. Ethical behaviour is the key here. Teachers must give a good example and instructions for the students about scientific writing practises. However, it should be clear for everyone that you are not allowed to steal ideas and you should give credit to the right persons and papers. The credit giving is done by using a proper citation technique.

The automatic plagiarism detection tools make the life of the teachers a bit easier. They save time in the evaluation process. Instead of having to manually search for excerpts of the student paper from the existing, published papers, the tool does the similarity check automatically in seconds or minutes. Teachers are still very much needed to evaluate the quality of the student paper, as well as, possible similarity findings of the tool. To improve the student's writing skills teachers also need to explain to the student how the writing process should be improved.

Bibliography

[1] Bilic-Zulle, L., Azman, J., Frkovic, V., and Petrovecki, M. *Is There an Effective Approach to Deterring Students from Plagiarizing?* Sci Eng Ethics (2008) 14:139-147

[2] Comas-Forgas, R. and J. Sureda-Negre, J. *Academics Plagiarism: Explanatory Factors from Students' Perspective.* J Acad Ethics (2010) 8:217-232. Springer.

[3] Gu. Q. and Brooks, J. *Beyond the accusation of plagiarism.* System 36 (2008) pp 337-352, Elsevier.

[4] Delvin, M. *Plagiarism detection software: how effective is it? Assessing Learning in Australian Universities, 2002.* Available

at: http://www.cshe.unimelb.edu.au/assessinglearning/docs/ Plag-Software.pdf

[5] Imran, N. *Electronic Media, Creativity and Plagiarism.* SIGCAS Computers and Society, Volume 40, No. 4, December 2010

[6] Jones. K. *Practical Issues for Academics Using the Turnitin Plagiarism Detection Software.* International Conference on Computer Systems and Technologies - CompSysTech'08, ACM, 2008, IV.1-1 - IV.1-5

[7] Kaner, C. and Fiedler, R. *A Cautionary Note on Checking Software Engineering Papers for Plagiarism.* IEEE Transactions on, Volume 51, No. 2, May 2008, pp. 184-188

[8] Lukashenko, R., Graudina, V., and Grundspenkis, J. *Computer-Based Plagiarism Detection Methods and Tools: An Overview.* International Conference on Computer Systems and Technologies - CompSysTech'07, ACM, 2007, IIIA.18-1 - IIIA.18-6

[9] Smith, N. and Wren. K. *Ethical and Legal Aspects Part 2: Plagiarism – "What Is It and How Do I Avoid It?".* Journal of PeriAnesthesia Nursing, Vol 25, No 5 (October), 2010: pp 327-330

Regenerative simulation of the loss probability in the finite buffer queue with Brownian input

Ruslana S. Goricheva[†], Oleg V. Lukashenko[†], Evsey V. Morozov[†], Michele Pagano[‡]

[†]Karelian Research Center RAS

[‡]University of Pisa

11, Pushkinskaya Street, Petrozavodsk, Karelia, Russia, 185640
Via Caruso,16, Pisa, Italy, I-56126

Abstract

We discuss the application of the regenerative simulation to estimate the loss probability in a queueing system with finite buffer which is fed by a Brownian input (Bi). Some numerical examples are also included. This work is supported by Russian Foundation for Basic research, project No 10-07-00017.

1 Introduction

In this work we are interested in systems with small or moderate size buffers, because it is motivated by real network applications, which have stringent requirements to queueing delay. So the loss rate prediction can be useful to provide suitable level of Quality of Service.

To motivate our interest to systems fed by Bi, we note that appropriately scaled superposition of large number of identically distributed (i.i.d.) on-off sources with finite variances converges weakly to Brownian motion (Bm) provided first, number of sources $M \to \infty$, and then scaling factor $T \to \infty$ (see [5, 8] for more details).

This result gives formal motivation for the following definition of Bi, which is used below:

$$A(t) = mt + \sqrt{am}B(t), \tag{1.1}$$

where m is the mean input rate, and Bm $\{B(t),\, t \geq 0,\}$ describes random fluctuations of the input around its linearly increasing mean, a - some constant, see [6].

2 Queue with Brownian input

It is known that the workload $Q(t)$ can be calculated by the following Lindley-type recursion for the finite buffer system [6]:

$$Q(t) = \min((Q(t-1) - C + m + \sqrt{am}(B(t) - B(t-1)))^+, b),\ t = 1, 2, \cdots \quad (2.1)$$

where $(x)^+ = \max(0, x)$.

A typical sample path of the workload process (2.1) is presented in Figure 1 (where $C = 1$, $m = 0.7$, $b = 3$).

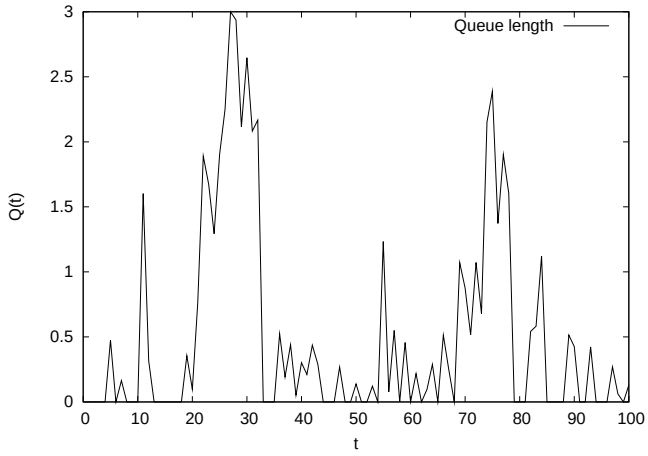


Figure 1: Finite buffer system with Bi sample path

3 Regenerative method

In this section, we describe in brief the method of regenerative simulation and the weakest known condition under which the regenerative method can be applied for the confidence estimation. A process $X = \{X_t, t \in T\}$, where $T = [0, \infty)$ (or $T = \{0, 1, \dots\}$) is called *regenerative process* if there exists an infinite sequence of instants $0 = \beta_0 < \beta_1 < \beta_2 < \cdots$ (regeneration points) such that the segments $G_n = (X_t,\ \beta_{n-1} \le t < \beta_n)$ (regeneration cycles) are i.i.d. The cycle periods $\beta_{n+1} - \beta_n$, $n \ge 0$, are also i.i.d. and we denote by β generic regeneration period.

For definiteness, we consider a discrete-time positive recurrent process X (that is $\mathsf{E}\beta < \infty$) and assume that regeneration cycle length β is aperiodic ($\P(\beta = 1) > 0$). Then the weak limit $X_n \Rightarrow X$ as $n \to \infty$ exists such

that $\mathsf{P}(X < \infty) = 1$. Moreover, if f is a measurable function, then the following statement holds:

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=0}^{n} f(X_i) = \frac{\mathsf{E}[\sum_{i=0}^{\beta-1} f(X_i)]}{\mathsf{E}\beta} \equiv r.$$

It is also assumed that $\mathsf{E}[\sum_{i=0}^{\beta-1} |f(X_i)|] < \infty$. (Note that an evident analog for continuous-time process also exists. More details can be found in [1].)

To estimate the unknown parameter r (steady-state performance measure), we group the data belonging to the same regenerative cycle to obtain the i.i.d. enlarged variables

$$Y_k := \sum_{n=\beta_{k-1}}^{\beta_k-1} f(X_n), \qquad k \geq 1.$$

We now define the main sample-mean ratio-type estimator as follows:

$$r_n \equiv \frac{\overline{Y}_n}{\overline{\alpha}_n},$$

where $\overline{\alpha}_n$ is the sample mean cycle period,

$$\overline{Y}_n = \frac{1}{n} \sum_{i}^{n} Y_i,$$

and n is the number of completed regeneration cycles obtained during simulation. Let us also denote the variance of the enlarged variable as $\sigma^2 = \mathsf{E}(Y - r\beta)^2$.

If the (minimal sufficient) condition

$$0 < \mathsf{E}(Y - r\beta)^2 < \infty$$

holds, then the following *Regenerative Central Limit Theorem* can be applied [2]:

$$n^{1/2}\overline{\alpha}_n[r_n - r] \Rightarrow \sigma N(0,1), \qquad n \to \infty.$$

This leads to the following $100(1-\gamma)\%$ asymptotic confidence interval for the unknown (steady-state) performance measure r:

$$\left[r_n - \frac{z_\gamma s(n)}{\overline{\alpha}_n \sqrt{n}},\ \ r_n + \frac{z_\gamma s(n)}{\overline{\alpha}_n \sqrt{n}} \right], \tag{3.1}$$

where

$$\P\left(N(0,1) > z_\gamma\right) \ = \ \frac{1-\gamma}{2}$$

and $s^2(n)$ is the empirical variance, which converges with probability 1 to the variance

$$s^2(n) \to \sigma^2,$$

when the number of observed regeneration cycles $n \to \infty$.

4 Regenerative structure and simulation results

Using regenerative approach, we present the way of loss rate estimation, which can be applied in system $Bi/D/1/n$. First we construct regeneration points for the content process. (More details can be found in [3].) Let $\beta_0 = 0$ and

$$\beta_{k+1} = \min\{t > \beta_k : Q(t-1) = 0,\, Q(t) > 0,\ k \geq 1\}, \tag{4.1}$$

where $Q(t)$ is the queue content at the end of slot t. It is important to stress that in continuous-time setting construction of regenerations meets a difficulty caused by structure of the Brownian input paths [4].

Now we denote by $L_b(t)$ the total load lost in time interval $[0,\, t]$. Denote by $\mathsf{E}L$ the mean load lost per regenerative cycle and let $\mathsf{E}A$ be the mean load arrived per regenerative cycle. Applying regenerative method, we obtain the following representation for the steady-state loss probability

$$\lim_{t\to\infty} \frac{L_b(t)}{A(t)} = \frac{\mathsf{E}L}{\mathsf{E}A} := P_\ell.$$

To apply confidence estimation based on regenerative central limit theorem to estimate probability $P_\ell := r$, we treat processes $\{L_b(t),\, t \geq 0\}$ and $\{A(t),\, t \geq 0\}$ as *cumulative processes* with embedded regenerations defined by recursion (4.1), see [7]. Then we use regenerative simulation to estimate limit ratios

$$r_1 := \lim_{t\to\infty} \frac{L_b(t)}{t} = \frac{\mathsf{E}L}{\mathsf{E}\beta},\ r_2 := \lim_{t\to\infty} \frac{A(t)}{t} = \frac{\mathsf{E}A}{\mathsf{E}\beta},$$

separately, and then use equality $r = r_1/r_2$. Denote by $I_i = [a_i,\, b_i]$ confidence interval (with a confidence level $1-\alpha$) for r_i, $i = 1, 2$. Then it is easy to see that unknown parameter r is covered by the confidence interval $I = [a_1/b_2,\, b_1/a_2]$ (provided $a_2 > 0$) with probability

$$\begin{aligned}
\P(r \in I) &\geq \P(r_i \in I_i,\, i = 1, 2) \\
&\geq \frac{1}{2}\Big(\P(r_1 \in I_1) + \P(r_2 \in I_2) - \P(r_1 \notin I_1) - \P(r_2 \notin I_2)\Big) \\
&= 1 - 2\alpha.
\end{aligned}$$

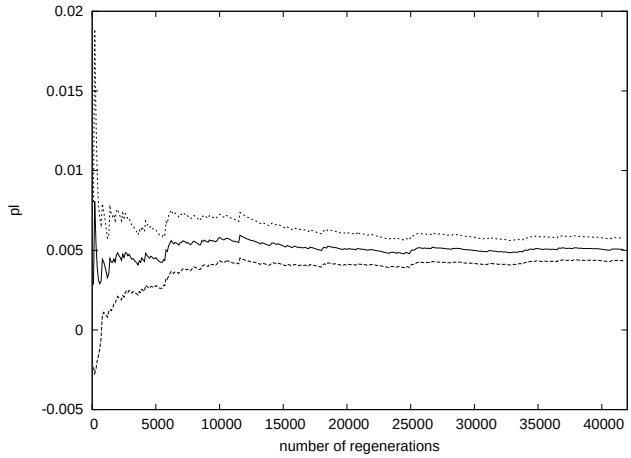


Figure 2: 90% Confidence interval for P_ℓ in Bi/D/1/4

In our experiments we have used $\alpha = 0.05$, so resulting confidence interval has level 90%.

Figure 2 shows 90% confidence interval for the loss probability in the system $Bi/D/1/b$ with Brownian input (with rate $m = 0.7$, service rate $C = 1$ and buffer size $b = 4$) as a function of the simulation length (in terms of regeneration cycles).

Figure 3 shows 90% confidence interval for the loss probability as a function of the buffer size. The following parameters are used: $C = 1$; $m = 0.7$; $N = 10^6$ (where N denotes the number of simulated time slots).

Finally, figure 4 shows 90% confidence interval for the loss probability as a function of the service rate. The following parameters are used: $b = 4$; $m = 0.7$; $N = 10^6$.

To explain an increasing of the confidence length on Figures (3), (4), we recall that use log scale. Moreover, shift below of the center of the intervals caused by decreasing of the loss probability as the buffer size increases.

We note that using regenerations leads to a reliable estimation due to the i.i.d. property of the regeneration cycles.

5 Conclusion

The results of our study can be summarised as follows:

A key observation is that Brownian process has i.i.d. increments and it allows us to construct confidence interval for the loss probability based on simulation of the i.i.d. regeneration cycles. Using such a simulation we

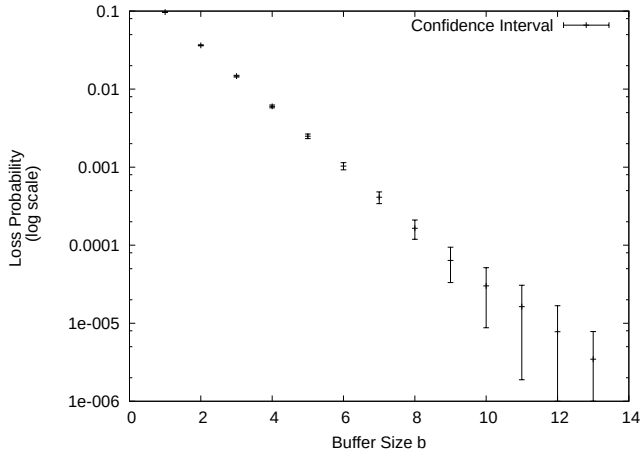


Figure 3: Dependence of log estimate of P_ℓ on buffer size b

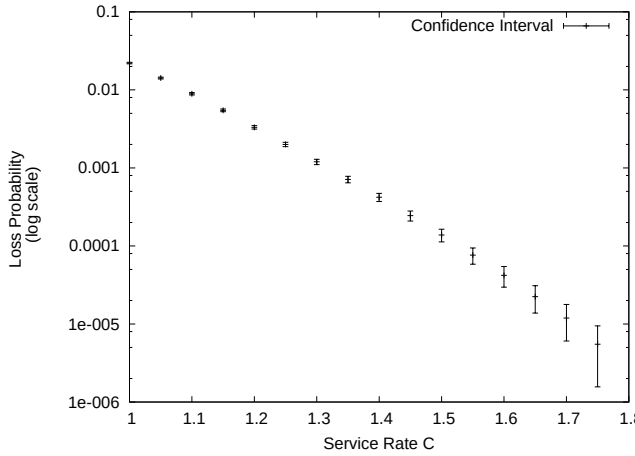


Figure 4: Dependence of log estimate of P_ℓ on service rate C

present a few numerical examples.

Bibliography

[1] *Asmussen S.* Applied Probability and Queues, Springer, 2002.

[2] *Glynn P. W., Iglehart D. L.* Conditions for the applicability of the regenerative method, Management Science, 1993, 39, 1108–1111.

[3] *Goricheva R. S., Lukashenko O. V., Morozov E. V., Pagano M.* Regenerative analysis of a finite buffer fluid queue, Proceedings of ICUMT 2010 (electronic publication).

[4] *Mandjes M.* Large Deviations of Gaussian Queues. Chichester: Wiley, 2007.

[5] *Mikosh T., Reshick S., Rootzen H., Stegeman A.* Is network traffic approximated by stable Levy motion or fractional Brownian motion. Annals of Applied Probability, 2002, v. 12, 23-68.

[6] *Norros I.* A storage model with self-similar input, Queueing Systems, 1994. v. 16, 387–396.

[7] *Smith W.L.* Regenerative stochastic processes, Proc. Roy. Soc., ser. A 232 1955, 6-31.

[8] *Taqqu M., Willinger W., Sherman R.* Proof of a fundamental result in self-similar traffic modellling. Computer Communication Review, v. 27, 1997, 5-23.