УДК 004.42–027.45

МЕТОД ПРОГНОЗИРОВАНИЯ НАДЕЖНОСТИ ПРИКЛАДНЫХ ПРОГРАММНЫХ СРЕДСТВ НА РАННИХ ЭТАПАХ ИХ РАЗРАБОТКИ

С.М. БОРОВИКОВ[1], С.С. ДИК[1], Н.К. ФОМЕНКО[2]

[1]*Белорусский государственный университет информатики и радиоэлектроники, Республика Беларусь*

[2]*Филиал БНТУ «Минский государственный политехнический колледж», Республика Беларусь*

Поступила в редакцию 13 июня 2018

Аннотация. Для прогнозирования ожидаемой надежности разрабатываемых прикладных программных средств предлагается метод, основанный на использовании статистической модели работоспособности программных средств. Модель базируется на взятых из отечественной и зарубежной печати обобщенных статистических данных об ожидаемом числе ошибок в разрабатываемом прикладном программном средстве в зависимости от числа строк кода компьютерной программы, продолжительности ее тестирования, квалификации и опыта тестировщиков. Приводится пример прогнозирования ожидаемой надежности прикладного программного средства.

Ключевые слова: прикладное программное средство, статистическая модель работоспособности, ожидаемая надежность, прогнозирование.

Abstract. To predict the expected reliability of the developed software applications, a method based on the use of the statistical model of the operability of software is proposed. The model is based on generalized statistical data taken from domestic and foreign printing on the expected number of errors in the developed application software, depending on the number of lines of computer program code, the duration of its testing, the qualifications and experience of testers. An example of predicting the expected reliability of an application software is given

Keywords: application software, statistical model of failure-free operation, expected reliability, forecasting.

Doklady BGUIR. 2019, Vol. 123, No. 5, pp. 45-51
Method for predicting the reliability of applied software at the early stages of their development
S.M. Borovikov, S.S. Dick, N.K. Fomenko

DOI: http://dx.doi.org/10.35596/1729-7648-2019-123-5-45-51

Введение

Под надежностью программного средства (ПС) понимают способность ПС обеспечивать требуемый уровень его пригодности для решения задач пользователя в заданных условиях и режимах применения. Ограничения уровня пригодности ПС являются следствием наличия дефектов (ошибок), внесенных в компьютерные программы и содержание ПС в процессе постановки и решения задачи их создания. Ошибка компьютерной программы – это погрешность или искажение кода, которые при определенном наборе входных данных в ходе выполнения этой программы могут вызвать ее отказ или привести к неверным результатам. Когда ПС становится сложным, ошибки просто неизбежны. Невозможно найти и исправить все ошибки. [1]. Количество и характер отказов ПС, являющихся следствием этих ошибок, зависят от способа применения ПС и выбираемых вариантов его функционирования. Входные исходные данные, используемые пользователем

при выполнении компьютерной программы, могут попасть на такую трассу ее выполнения, которая не была проверена на этапе тестирования ПС.

Актуальность разработки

В сложных информационно-компьютерных системах вклад программного обеспечения в ненадежность систем составляет до 40…50 %. По мнению некоторых авторов, этот вклад может превышать вклад, вносимый техническими средствами (компьютерами), поскольку входные данные могут быть сложнее и их формат все время меняется. Надежность электронно-вычислительных средств ограничивается качеством используемых комплектующих элементов, ошибками проектирования, производственными дефектами и частотой сбоев. В общем случае программное обеспечение сложнее технических средств. Его объем для современных информационно-компьютерных систем может достигать сотен миллионов и более команд [1].

Разработанные методы и алгоритмы оценки надежности прикладных ПС предполагают, что устранены ошибки, обусловленные нарушениями правил языка программирования, выполнена отладка кода программы и имеются определенные данные о тестировании ПС. Однако в большинстве случаев проектировщики информационно-компьютерных систем хотели бы знать ожидаемый уровень надежности прикладных ПС на ранних этапах их разработки. Поэтому актуальным является разработка метода, показывающего, как спрогнозировать уровень надежности ПС до написания кода компьютерной программы на языке программирования. Наличие соответствующего метода позволит хотя бы ориентировочно оценить ожидаемый уровень надежности будущего ПС, что даст возможность определить его вклад в общую ненадежность информационно-компьютерной системы.

Теоретический анализ

Среди специалистов по программированию в качестве единицы измерения объема ПС, относительно хорошо устанавливающей соотношение с числом возможных ошибок, используют количество строк кода (в англоязычном варианте: Lines Of Code – LOC). LOC – это метрика ПС, используемая для измерения его размера путем подсчета числа строк в тексте исходного кода компьютерной программы.

Обычно используют физические строки кода, их определяют как общее число строк исходного кода, включая комментарии. Однако результаты подсчета существенным образом зависят от правил оформления и форматирования исходного кода. Тем не менее физические строки кода интуитивно понятнее и их проще считать, нежели логические строки кода, для определения числа которых надо подсчитать количество собственно операторов в программе.

Программное обеспечение современных информационно-компьютерных систем является достаточно сложным, и есть веские основания считать, что оно станет еще сложнее в ближайшем будущем. Например, по данным Nathan Myhrvold [2], в 1983 году текстовый редактор Microsoft Word включал 27 тыс. строк кода, а к 1995 году размер этого редактора заметно увеличился, достигнув 2 млн строк кода.

Чтобы оценить всю сложность современного программного обеспечения, рекомендуется проанализировать информацию, приведенную в табл. 1.

Таблица 1. **Объем программного обеспечения**

Программное средство (объект)	Космическая станция	Космический корабль	Boeing 777	Windows NT5	Linux 2,6	Windows 2000	Windows XP	Windows 7
Количество строк кода, млн	40	10	7	35	5,6	35	45	> 60

По данным, приводимым в технической литературе, среднее число ошибок $N_{ош}$, приходящихся на тысячу строк кода (KLOC), определяется назначением и особенностью разработки компьютерной программы. Согласно [3], значение $N_{ош}$ для ответственных прикладных ПС лежит в диапазоне от 5 до 35 ошибок на тысячу строк кода.

В программных средствах, которые тестировались только на предмет выполнения ими своих функций, что имеет место для большого числа ПС, поставляемых пользователям на коммерческой основе, присутствует намного больше ошибок: до 15…50 ошибок на 1000 строк кода (табл. 2).

Таблица 2. **Среднее число дефектов (ошибок) на 1000 строк кода для прикладных программных средств, прошедших тестирование**

Назначение ПС, разработчик	Intel	Microsoft	Linux	NASA	Коммерческие прикладные ПС
Среднее число дефектов (ошибок) на 1000 строк кода	4…5	8…18	8…34	3…8	15…48

В качестве количественных показателей, характеризующих уровень надежности программного средства, можно использовать показатели, сходные с показателями безотказности аппаратной части информационно-компьютерных систем. Далее в качестве таких показателей для ПС используются интенсивность проявления дефектов (ошибок) $\lambda_{ПС}$ и наработка на проявление ошибки T_0. Между показателями $\lambda_{ПС}$ и T_0 имеет место определенная связь, позволяющая по значению одного из них найти другой показатель.

Надежность ПС зависит от используемых технологий и продолжительности процедуры тестирования. С помощью тестирования должно быть выявлено как можно больше смысловых ошибок с учетом возможного формата изменения исходных данных. Тестирование требует значительного времени, и даже после его завершения некоторые ошибки в ПС остаются необнаруженными.

Возможные подходы к оценке ожидаемой надежности прикладных программных ПС на ранних этапах их разработки обсуждались в работах [4, 5]. В данной статье в систематизированном виде изложен разработанный авторами метод оценки уровня надежности ПС до написания кода компьютерной программы на языке программирования. Этот метод использует модель надежности, основанную на статистическом подходе к оценке уровня работоспособности компьютерной программы. Обобщенные сведения о надежности прикладных ПС взяты из опыта использования их по назначению в различных областях деятельности людей. Ниже приводятся основные статистические данные, принятые в модели надежности разрабатываемых прикладных ПС [3]:

– профессиональные программисты со стажем не менее 10 лет на 1000 строк кода допускают в среднем 131,3 ошибки;

– до 50 % ошибок программного кода выявляется на этапе компиляции (если транслятор имеет развитую систему предупреждений);

– на этапе тестирования обнаруживается 50 % и более ошибок, оставшихся после устранения синтаксических ошибок, обусловленных нарушением правил языка программирования;

– среднее число ошибок, приходящихся на тысячу строк кода компьютерной программы, прошедшей тестирование, находится в диапазоне от 5 до 50.

Метод оценки ожидаемой надежности прикладного программного обеспечения

Оценка надежности выполняется по модели надежности, использующей усредненные статистические данные об ожидаемом числе ошибок в ПС в зависимости от объема кода компьютерной программы, динамики уменьшения числа ошибок при тестировании, квалификации, опыта программистов и тестировщиков, а также от продолжительности тестирования. В качестве основы определения итоговой (прогнозной) надежности прикладного ПС выбрана модель Шумана [6, 7] в предположении, что вся продолжительность процедуры тестирования ПС рассматривается как один этап тестирования ($i = 1$). При оценке ожидаемой надежности ПС с использованием статистической модели его работоспособности приняты следующие основные допущения:

– написание программного кода, отладка и дальнейшее тестирование ПС будут выполняться профессиональными специалистами со стажем работы не менее 10 лет;

– интенсивность проявления ошибок $\lambda(t)$ для времени t прямо пропорциональна оставшемуся числу ошибок в ПС в момент времени t;

– время до проявления следующей ошибки распределено по экспоненциальному закону;

– устранение ошибок на этапе тестирования осуществляется без внесения в программу других ошибок.

В качестве исходных данных рассматриваются:

– предполагаемый размер (объем) компьютерной программы в строках кода (LOC);

– продолжительность выполнения этапа тестирования ПС;

– количество специалистов, привлекаемых к тестированию ПС.

Ниже приводится описание основных этапов применения метода.

1. Определение прогнозного значения числа ошибок $N_{\text{нач}}$, оставшихся в компьютерной программе после написания ее кода и устранения ошибок, обусловленных нарушениями правил языка программирования (ошибок трансляции):

$$N_{\text{нач}} = \frac{131{,}3 \cdot S\left(1 - K_{\text{отл}}\right)}{1000}, \quad (1)$$

где S – объем ПС в строках кода; $K_{\text{отл}}$ – коэффициент, показывающий долю ошибок, устраняемых при отладке ПС; по умолчанию для профессиональных программистов со стажем работы не менее 10 лет, согласно [3], может быть принят $K_{\text{отл}} = 0{,}5$.

2. Расчет ожидаемого числа ошибок, оставшихся в ПС после проведения тестирования, $N_{\text{п.тест}}$:

$$N_{\text{п.тест}} = (1 - K_{\text{тест}}) N_{\text{нач}}, \quad (2)$$

где $K_{\text{тест}}$ – коэффициент, показывающий долю ошибок, выявляемых при тестировании; для профессиональных специалистов со стажем работы не менее 10 лет может быть принят $K_{\text{тест}} \approx 0{,}5$ [3]. Для специалистов со стажем менее 10 лет коэффициент $K_{\text{тест}}$ подлежит уточнению, в этих случаях $K_{\text{тест}} < 0{,}5$.

3. Определение общего времени тестирования $t_{\text{тест}}$:

$$t_{\text{тест}} = n_{\text{дн}} m_{\text{т}} t_{\text{дн}} K_{\text{ПС}}, \quad (3)$$

где $n_{\text{дн}}$ – число рабочих дней, отводимое для тестирования ПС; $m_{\text{т}}$ – количество задействованных тестировщиков; $t_{\text{дн}}$ – продолжительность рабочего дня тестировщика; $K_{\text{ПС}}$ – коэффициент, показывающий, какая часть времени в течение рабочего дня в среднем используется тестировщиком для прогона (исполнения на компьютере) ПС; по умолчанию можно принять $K_{\text{ПС}} = 0{,}15$ (по данным Объединенного института проблем информатики НАН Беларуси).

4. Согласно модели Шумана [6, 7], для интенсивности проявления ошибок на i-м этапе λ_i, справедливо выражение

$$\lambda_i = (N_{\text{нач}} - n_{i\text{-}1}) C, 1/\text{ч}, \quad (4)$$

где $n_{i\text{-}1}$ – число ошибок, исправленных к началу i-го этапа тестирования; C – коэффициент пропорциональности.

Вся продолжительность процедуры тестирования ПС рассматривается как один этап тестирования ($i = 1$). Поскольку $i = 1$, следовательно, в данном случае $n_{i\text{-}1} = n_0 = 0$. С учетом этого коэффициент C в данном случае рассчитывается как

$$C = \frac{\frac{n_{\text{тест}}}{N_{\text{нач}} - n_0}}{t_{\text{тест}}} = \left| \begin{array}{l} \text{С учетом того, что} \\ n_{\text{тест}} = N_{\text{нач}} - N_{\text{п.тест}} \end{array} \right| = \frac{N_{\text{нач}} - N_{\text{п.тест}}}{N_{\text{нач}} \cdot t_{\text{тест}}}, \ 1/\text{ч}, \quad (5)$$

где $t_{\text{тест}}$ – планируемое время тестирования ПС, представляющее собой суммарное время прогона ПС (выполнения на компьютере) на этапе тестирования; $n_{\text{тест}}$ – прогнозное число обнаруженных ошибок при тестировании ($n_{\text{тест}} = N_{\text{нач}} - N_{\text{п.тест}}$).

5. Определение прогнозного значения ожидаемой интенсивности проявления ошибок λ (интенсивности отказов ПС) после завершения процедуры тестирования и исправления выявленных ошибок, т. е. расчет значения λ, соответствующего начальному этапу эксплуатации ПС:

$$\lambda_{\text{эксп}} = (N_{\text{нач}} - n_{\text{тест}})C = N_{\text{п.тест}} \cdot C \text{, 1/ч,} \quad (6)$$

где $\lambda_{\text{эксп}}$ – интенсивность проявления ошибок для начального периода использования ПС по его функциональному назначению.

Примечание. С учетом выражения (5) для коэффициента C прогнозное значение $\lambda_{\text{эксп}}$ может быть сразу рассчитано по формуле

$$\lambda_{\text{эксп}} = \frac{N_{\text{п.тест}}(N_{\text{нач}} - N_{\text{п.тест}})}{N_{\text{нач}} \cdot t_{\text{тест}}} \text{, 1/ч.} \quad (7)$$

6. Определение прогнозного значения наработки на проявление ошибки $T_{0.\text{ПС}}$ для начального периода использования ПС по его функциональному назначению. Принимая гипотезу об экспоненциальном распределении времени использования ПС до возникновения ошибки, значение T_0 может быть найдено по выражению

$$T_{0.\text{ПС}} = \frac{1}{\lambda_{\text{эксп}}} = \frac{1}{N_{\text{п.тест}} \cdot C} = \frac{N_{\text{нач}} \cdot t_{\text{тест}}}{N_{\text{п.тест}}(N_{\text{нач}} - N_{\text{п.тест}})} \text{, ч.} \quad (8)$$

Предлагаемый метод позволяет ориентировочно оценить ожидаемую надежность ПС для начального периода его эксплуатации. Ниже приводится пример прогнозирования (оценки) ожидаемой надежности прикладного ПС с использованием метода.

Применение метода и обсуждение результатов

Пример. Разрабатываемое прикладное ПС будет включать ориентировочно 5 тысяч строк программного кода. Продолжительность выполнения тестирования программного ПС пятью специалистами с опытом работы более 10 лет определена в размере 30 недель при пятидневной рабочей неделе (150 рабочих дней) и 8-часовой продолжительности рабочего дня. Определим прогнозное значение наработки на проявление ошибки $T_{0.\text{ПС}}$, которое будет иметь место для ПС после завершения этапа его тестирования.

Применение метода. 1. Используя формулу (1), находим ожидаемое число ошибок $N_{\text{нач}}$, которое будет содержаться в компьютерной программе после написания ее кода на языке программирования и устранения ошибок, вызванных нарушением правил языка (ошибок трансляции программы): $N_{\text{нач}} \approx 328$ ошибок.

2. По выражению (2) определяем ожидаемое число ошибок $N_{\text{п.тест}}$, оставшихся в ПС после завершения этапа тестирования. Примем $K_{\text{тест}} = 0{,}5$. Получим $N_{\text{п.тест}} \approx 164$ ошибки.

3. По формуле (3) определяем прогнозное время тестирования ПС – время выполнения (прогона) на компьютере. Примем $K_{\text{ПС}} = 0{,}15$. Получим $t_{\text{тест}} = 150 \cdot 5 \cdot 8 \cdot 0{,}15 = 900$ ч.

4. По формуле (5) рассчитываем значение коэффициента C с учетом того, что $n_{\text{тест}} = N_{\text{нач}} - N_{\text{п.тест}} = 328$ ошибок, $n_0 = 0$: $C = 328/(656 \cdot 900) \approx 0{,}000556$ 1/ч.

5. Пользуясь выражением (6), находим интенсивность проявления ошибок $\lambda_{\text{эксп}}$, соответствующую окончанию этапа тестирования и, следовательно, начальному периоду использования ПС по его функциональному назначению: $\lambda_{\text{эксп}} = 328 \cdot 0{,}000556 \approx 0{,}091$ 1/ч.

6. Принимая во внимание выражение (8), определяем среднюю наработку на проявление ошибки $T_{0.\text{ПС}}$ для начального этапа эксплуатации ПС: $T_{0.\text{ПС}}$ 10,99 ≈ 11 ч.

С учетом того, что для реальных информационно-компьютерных систем наработка на отказ аппаратных средств (надежность технической части системы) составляет примерно тысячи часов, а ожидаемая наработка на проявление ошибки $T_{0.\text{ПС}}$ прикладного ПС, согласно расчету, всего 11 ч, может сложиться впечатление, что вклад ПС в общую ненадежность системы является определяющим, а не примерно до 50 %, как было отмечено ранее.

Сделаем пояснения. Во-первых, полученное расчетное значение $T_{0.\text{ПС}} = 11$ ч соответствует начальному периоду опытной эксплуатации прикладного ПС, и с увеличением времени эксплуатации значение $T_{0.\text{ПС}}$ возрастает за счет того, что ошибки, проявившиеся при эксплуатации, устраняются. И второе, время использования ПС при выполнении целевого задания, возлагаемого на информационно-компьютерную систему, оказывается, как правило, заметно меньше установленного нормативного календарного времени $t_{\text{к}}$ выполнения системой целевого задания, поэтому возникает резерв времени на восстановление разработанного прикладного ПС в случаях проявления ошибки и дальнейшего выполнения целевого задания с помощью обновленного ПС (после поиска ошибки и редактирования кода), т. е. для ПС

предоставляется возможным использовать временно́е резервирование [8]. Задержка в выполнении ПС целевого задания до истечения времени $t_к$ не рассматривается как срыв в выполнении целевого задания, т. е. отказ информационно-компьютерной системы. Отказ системы по вине рассматриваемого ПС возникает, если восстановление его работоспособности и выполнение целевого задания с помощью восстановленной версии ПС не будут выполнены до момента окончания интервала времени $t_к$.

Заключение

Предложенный метод дает весьма приближенные результаты ожидаемой надежности разрабатываемого прикладного ПС. Однако даже такой ориентировочный расчет полезен, так как позволяет получить представление о надежности ПС на раннем этапе его разработки, до написания программного кода. Кроме того, используя описанный метод, можно ориентировочно определить требуемое время на тестирование, необходимое для обеспечения заданного уровня надежности разрабатываемого прикладного ПС.

При оценке ожидаемой надежности программного обеспечения, используемого для решения прикладных задач с помощью информационно-компьютерной системы, следует также принять во внимание надежность системного программного обеспечения и надежность используемых драйверов. Модель работоспособности этого программного обеспечения и метод прогнозирования его ожидаемой надежности будут отличаться от модели и метода, предложенных для прикладных ПС. Пользователи вычислительных систем хотели бы знать, как оценить ожидаемую надежность используемых в практике системных программных средств. Решение этой задачи является предметом отдельного исследования.

Список литературы

1. Программное обеспечение – источник всех проблем [Электронный ресурс]. URL: http://www.williamspublishing.com/PDF/5-8459-0785-3/part1.pdf. (дата обращения: 25.04.2018).
2. The physicist. Trending now [Электронный ресурс]. URL: http://www.wired.com/wired/archive/3.09/myhrvold.html?person=gordon_moore&topic_set=wiredpeople. (дата обращения: 25.04.2018).
3. Методы обеспечения аппаратно-программной надежности вычислительных систем. д.т.н., проф. Чуканов В.О., к.т.н., доц. Гуров В.В. [Электронный ресурс]. URL: http://www.mcst.ru/files/5357ec/dd0cd8/50af39/000000/seminar_metody_obespecheniya_apparatno-programmnoy_nadezhnosti_vychislitelnyh_sistem.pdf. (дата обращения: 25.04.2018).
4. Боровиков С.М., Дик С.С. Прогнозирование ожидаемой надежности прикладных программных средств с использованием статистических моделей их безотказности [Электронный ресурс]. URL: https://libeldoc.bsuir.by/handle/123456789/31393. (дата обращения: 25.04.2018).
5. Боровиков С.М., Будник А.В., Закривашевич М.Н. Оценка ожидаемой надежности прикладных программных средств на начальном этапе их разработки [Электронный ресурс]. URL: http://bsac.by/sites/default/files/content/event/2017/add/files/10-16/sbornik-ccc-2017-poslednii.pdf или https://libeldoc.bsuir.by/handle/123456789/28865 (дата обращения: 25.04.2018).
6. Чуканов В.О. Надежность программного обеспечения и аппаратных средств систем передачи данных атомных электростанций. М.: МИФИ, 2008. 68 с.
7. Shooman M.L. Software engineering: Reliability, Development and Management. McGraw-Hill: International. Book Co, 1983.
8. Надежность технических систем / Под ред. И.А. Ушакова. М.: Радио и связь, 1985. 608 с.

References

1. Programmnoe obespechenie – istochnik vseh problem [Electronic resourse]. URL: http://www.williamspublishing.com/PDF/5-8459-0785-3/part1.pdf. (date of access: 25.04.2018).
2. The physicist. Trending now [Electronic resourse]. URL: http://www.wired.com/wired/archive/3.09/myhrvold.html?person=gordon_moore&topic_set=wiredpeople. (date of access: 25.04.2018). (in Russ.)
3. Metody obespechenija apparatno-programmnoj nadezhnosti vychislitel'nyh sistem. d.t.n., prof. Chukanov V.O., k.t.n., doc. Gurov V.V. [Electronic resourse]. URL: http://www.mcst.ru/files/5357ec/dd0cd8/50af39/000000/seminar_metody_obespecheniya_apparatno-programmnoy_nadezhnosti_vychislitelnyh_sistem.pdf. (date of access: 25.04.2018). (in Russ.)

4. Borovikov S.M., Dik S.S. Prognozirovanie ozhidaemoj nadezhnosti prikladnyh programmnyh sredstv s ispol'zovaniem statisticheskih modelej ih bezotkaznosti [Electronic resourse]. URL: https://libeldoc.bsuir.by/handle/123456789/31393. (date of access: 25.04.2018). (in Russ.)
5. Borovikov S.M., Budnik A.V., Zakrivashevich M.N. Ocenka ozhidaemoj nadezhnosti prikladnyh programmnyh sredstv na nachal'nom jetape ih razrabotki [Electronic resourse]. URL: http://bsac.by/sites/default/files/content/event/2017/add/files/10-16/sbornik-ccc-2017-poslednii.pdf ili https://libeldoc.bsuir.by/handle/123456789/28865 (date of access: 25.04.2018). (in Russ.)
6. Chukanov V.O. Nadezhnost' programmnogo obespechenija i apparatnyh sredstv sistem peredachi dannyh atomnyh jelektrostancij. M.: MIFI, 2008. 68 s. (in Russ.)
7. Shooman M.L. Software engineering: Reliability, Development and Management. McGraw-Hill: International. Book Co, 1983.
8. Nadezhnost' tehnicheskih sistem / Pod red. I.A. Ushakova. M.: Radio i svjaz', 1985. 608 s. (in Russ.)

Сведения об авторах

Боровиков С.М., к.т.н., доцент кафедры проектирования информационно-компьютерных систем Белорусского государственного университета информатики и радиоэлектроники.

Дик С.С., аспирант Белорусского государственного университета информатики и радиоэлектроники.

Фоменко Н.К., преподаватель высшей категории филиала Белорусского национального технического университета «Минский государственный политехнический колледж».

Information about the authors

Borovikov S.M., PhD, associate professor of the department of information and computer systems design of Belarusian state university of informatics and radioelectronics.

Dick S.S., PG student of Belarusian state university of informatics and radioelectronics.

Fomenko N.K., teacher of the highest category of the branch of Belarusian national technical university «Minsk State Polytechnic College».

Адрес для корреспонденции

220013, Республика Беларусь,
г. Минск, ул. П. Бровки, 6,
Белорусский государственный университет
информатики и радиоэлектроники
тел. +375-29-610-24-34;
e-mail: bsm@bsuir.by
Боровиков Сергей Максимович

Address for correspondence

220013, Republic of Belarus,
Minsk, P. Brovka st., 6,
Belarusian state university
of informatics and radioelectronics
tel. +375-29-610-24-34;
e-mail: bsm@bsuir.by
Borovikov Sergei Maksimovich