УДК 519.217.2

Обзор существующих моделей нестационарных систем обслуживания и методов их расчета

Бубнов В. П., Сафонов В. И., Шардаков К. С.

Актуальность. *В настоящее время аппаратно-программные комплексы (АПК), входящие в контур управления подвижными объектами и технологическими процессами, все чаще функционируют в условиях пиковых нагрузок. Это обусловлено с одной стороны возрастанием угроз, вызванных техногенными, природными и человеческими факторами, с другой желанием использовать уже существующие АПК для решения новых более сложных задач. Для определения возможности реализации всех операций, связанных с технологическим циклом управления, на заданном временном интервале применяют математическое моделирование. Однако классические стационарные модели теории массового обслуживания не позволяют проводить моделирование в случае пиковых изменений рабочей нагрузки на заданном (директивном) временном интервале, что потребовало развития моделей нестационарных систем обслуживания (НСО).* ***Целью работы*** *является анализ основных публикаций направленных на развитие теоретических основ, моделей и методов расчета моделей НСО.* ***Результаты.*** *В статье представлены результаты систематизации и анализа моделей НСО с конечным числом заявок, правил их построения и методов расчета вероятностно-временных характеристик.* ***Элементами новизны работы*** *являются два подхода к построению и расчету моделей НСО, полный перечень моделей НСО, разработанный к настоящему времени, сравнительный анализ методов расчета вероятностно-временных характеристик и обсуждение особенностей их программной реализации.* ***Практическая значимость.*** *Материал статьи может быть эффективно использован в задачах анализа реализации операций управления АПК, находящихся в контуре управления технологическими процессами и подвижными объектами, расчета показателей их функциональной надежности. Применим для повышения точности расчета показателей надежности программных средств, для сокращения времени их испытаний на основе выбора стратегии отладки. А также для дальнейшего построения новых моделей НСО и методов их расчета.*

Ключевые слова: *нестационарная система обслуживания, НСО, диаграмма переходов и состояний, система однородных дифференциальных уравнений, уравнение Чепмена – Колмогорова, марковский процесс, заявка, модель роста надежности программного обеспечения.*

Введение

Для определения возможности реализации всех операций, связанных с технологическим циклом управления, на заданном временном интервале применяют математическое моделирование. Математической базой является теория массового обслуживания (ТМО), позволяющая решать разнообразные задачи анализа и синтеза аппаратно-программного комплекса (АПК) путем определения технико-экономических показателей эффективности функционирования комплексов в целом при известных технических параметрах их элементов и ра-

Библиографическая ссылка на статью:
Бубнов В. П., Сафонов В. И., Шардаков К. С. Обзор существующих моделей нестационарных систем обслуживания и методов их расчета // Системы управления, связи и безопасности. 2020. № 3. С. 65–121. DOI: 10.24411/2410-9916-2020-10303

Reference for citation:
Bubnov V. P., Safonov V. I., Shardakov K. S. Overview of existing models of non-stationary queuing systems and methods for their calculation. *Systems of Control, Communication and Security*, 2020, no. 3, pp. 65–121 (in Russian). DOI: 10.24411/2410-9916-2020-10303

бочей нагрузке. Широкую известность приобрели фундаментальные работы по теории вычислительных систем и компьютерных сетей, использующие ТМО Клейнрока А., Майорова С.А., Башарина Г.П., Новикова Г.И., Феррари Д., Авена О.И., Когана Я.И., Вишневского В.М., Рыжикова Ю.И., Хомоненко А.Д. и других авторов. С помощью моделей ТМО рассчитываются вероятностно-временные характеристики функционирования центральных процессоров и узлов коммутации, выполняется расчет потерь данных и загрузки линий связи, анализ буферной памяти и алгоритмов маршрутизации, решения пакета задач, необходимых для выдачи управляющего воздействия и т.п.

Большинство авторов используют модели ТМО в предположении, что очередь заявок бесконечна, существует стационарный режим, а коэффициент загрузки не превышает единицы. Однако наибольший практический и теоретический интерес представляют модели ТМО, учитывающие поведение АПК, функционирующих в условиях перегрузок на заданном (директивном) временном интервале. В работах Хинчина А.Я., Такача К., Гнеденко Б.В., Коваленко И.Н., Прохорова Ю.В., Ежова И.Ч. [1–7] положено начало «нестационарной» ТМО. Ряд результатов исследования моделей ТМО, параметры которых зависят от состояния системы, получены в работах Арсенишвили Г.Л., Абольникова Л.М. [8–9]. Некоторые характеристики однолинейных систем массового обслуживания с ординарным входящим потоком, интенсивность которого обратно пропорциональна величине очереди, рассмотрены в работах Конолли Б. и Хидиди Н. [10–11]. В дальнейшем появились работы авторов: Сиголова Г.Г., Люперсольского А.М., Скляревича Ф.А., Greenberg H., Leese E., Leguesdron P., Neuts M.F., Read R.R., Syshi R., Головко Н.И., Зефмана А.И. [12–18], посвященные анализу и расчету нестационарных вероятностных характеристик моделей ТМО с постоянными интенсивностями входящего потока и обслуживания с бесконечными или конечными накопителями. Анализ результатов, полученных в рассматриваемых работах, показывает, что точное исследование протекающих в моделях ТМО процессов при поступлении на вход потока заявок с изменяющейся интенсивностью в нестационарном режиме чрезвычайно трудно даже при экспоненциальных законах распределения вероятностей. Недостаточно хорошо изучено поведение нестационарных систем массового обслуживания с более общими предположениями о законах распределения времени между моментами поступления и обслуживания заявок. Любой учет особенностей, присущих реальным АПК (пиковые нагрузки, приоритетность, порядок выбора заявок из очереди на обслуживание, доступность каналов, отличие законов распределения временных интервалов от экспоненциального и т.п.) в математических моделях нестационарных систем массового обслуживания, в свою очередь, усложняет порядок расчета вероятностно-временных характеристик процесса обслуживания заявок.

Таким образом, существует противоречие между практической потребностью в решении задач анализа и прогнозирования функционирования реальных АПК в условиях реальной рабочей нагрузки и ограниченными возможностями существующих методов их моделирования. Указанные обстоятельства потре-

бовали разработки моделей нестационарных систем обслуживания и методов их расчета.

Целью настоящей работы является обзор основных научных работ, направленных на построение теории нестационарных систем обслуживания с конечным числом заявок, включающей: основные понятия и определения; математические модели; методы расчета вероятностно-временных показателей; программную реализацию и конкретные приложения этих моделей для решения инженерных задач.

Весь материал в статье отображает динамику развития процесса построения и расчета моделей нестационарных систем обслуживания. От первого подхода: построение диаграммы состояний и переходов; запись дифференциальных уравнений Чепмена – Колмогорова для каждого состояния; вывод общего уравнения системы однородных дифференциальных уравнений Чепмена – Колмогорова и решение системы уравнений численными методами. До второго подхода: разработка алгоритма формирования всех возможных состояний нестационарной системы обслуживания и правил перехода между ними; формирование матрицы коэффициентов системы дифференциальных уравнений Чепмена – Колмогорова и решение системы уравнений с помощью программного комплекса, основанного на численно-аналитическом методе.

Материал данной статьи, ввиду своей объемности, декомпозирован на ряд логических подразделов.

1. Основные понятия и определения, базовая модель нестационарной системы обслуживания (НСО).
2. Аппроксимация распределениями фазового типа.
3. Применение моделей НСО к оцениванию надежности программных средств и планированию их испытаний.
    3.1. Нестационарная модель надежности программных средств.
    3.2. Варианты стратегии отладки.
    3.3. Результаты прикладных расчетов и обоснование стратегии отладки.
    3.4. Обобщенная модель надежности программных средств (ПС).
    3.5. Комплекс программ расчета надежности и планирования испытаний ПС.
    3.6. Модели НСО надежности программных средств, учитывающие возможную вероятность обнаружения ошибок при тестировании и состоятельность используемых тестов.
4. Начальный подход к построению моделей НСО и расчету их вероятностей состояний.
5. Численно-аналитический метод расчета моделей нестационарных систем обслуживания.
    5.1. Алгоритм нумерации состояний НСО.
    5.2. Решение системы ОДУ.
    5.3. Особенности программной реализации метода.
    5.4. Оптимизация реализации численно-аналитического метода.
    5.5. Оценка точности.

5.6. Сравнительная оценка скорости работы программной реализации метода.

6. Алгоритмы формирования матрицы коэффициентов системы однородных дифференциальных уравнений Чепмена – Колмогорова.
   6.1. Рекурсивный алгоритм генерации матрицы коэффициентов системы однородных дифференциальных уравнений Чепмена – Колмогорова.
   6.2. Рекурсивный алгоритм формирования матрицы коэффициентов для двухканальной НСО.
   6.3. Рекурсивный алгоритм формирования матрицы коэффициентов для сетевой модели НСО.
   6.4. Сортировка элементов матрицы.
   6.5. Последовательный алгоритм генерации матрицы коэффициентов.
7. Новый подход к построению моделей НСО и расчету их вероятностей состояний.

1. Основные понятия и определения, базовая модель НСО

Под нестационарной системой обслуживания (НСО) понимается система обслуживания с переменными во времени вероятностями состояний, каждое из которых описывается минимум двумя переменными: число заявок, находящихся в системе, и число заявок, получивших обслуживание. Число заявок, поступающих на обслуживание, конечно. Впервые базовая модель одноканальной НСО представлена в публикации [19]. На вход одноканальной системы обслуживания последовательно поступает N заявок на обслуживание. Распределения длительности интервалов между моментами поступления и обслуживания заявок описываются экспоненциальными законами с интенсивностями, зависящими от номера заявки $\{\lambda_1, \lambda_2, ..., \lambda_N\}$ и $\{\mu_1, \mu_2, ..., \mu_N\}$ соответственно. Такая система представляется цепью Маркова с дискретным множеством состояний и непрерывным временем. Состояние системы в каждый момент времени характеризуется парой (i, j), где i – число поступивших, но еще не обслуженных заявок $(i = \overline{0, N})$, а j – число уже обслуженных заявок $(j = \overline{0, N-i})$. Переход из состояния (i, j) в состояние $(i+1, j)$ означает, что в систему поступила $(i + j + 1)$ заявка. Переход из состояния (i, j) в состояние $(i-1, j+1)$ означает, что была обслужена $(j+1)$-я заявка. Общее число состояний N_c вычисляется по формуле: $N_c = (N+1)(N+2)/2$. На рис. 1 представлена диаграмма переходов между состояниями системы.

Система из N_c линейных однородных дифференциальных уравнений Чепмена – Колмогорова с постоянными коэффициентами имеет вид:

$$\frac{dP_{i,j}(t)}{dt} = H(i)(P_{i-1,j}(t)\lambda_{i+j} - P_{i,j}(t)\mu_{j+1}) + \\ +H(j)(P_{i+1,j-1}(t)\mu_j) - H(N-i-j)P_{i,j}(t)\lambda_{i+j+1}), \quad (1)$$

где $H(k)=\begin{Bmatrix}1, если\ k>0;\\ 0, если\ k\le 0,\end{Bmatrix}$ – функция Хевисайда.

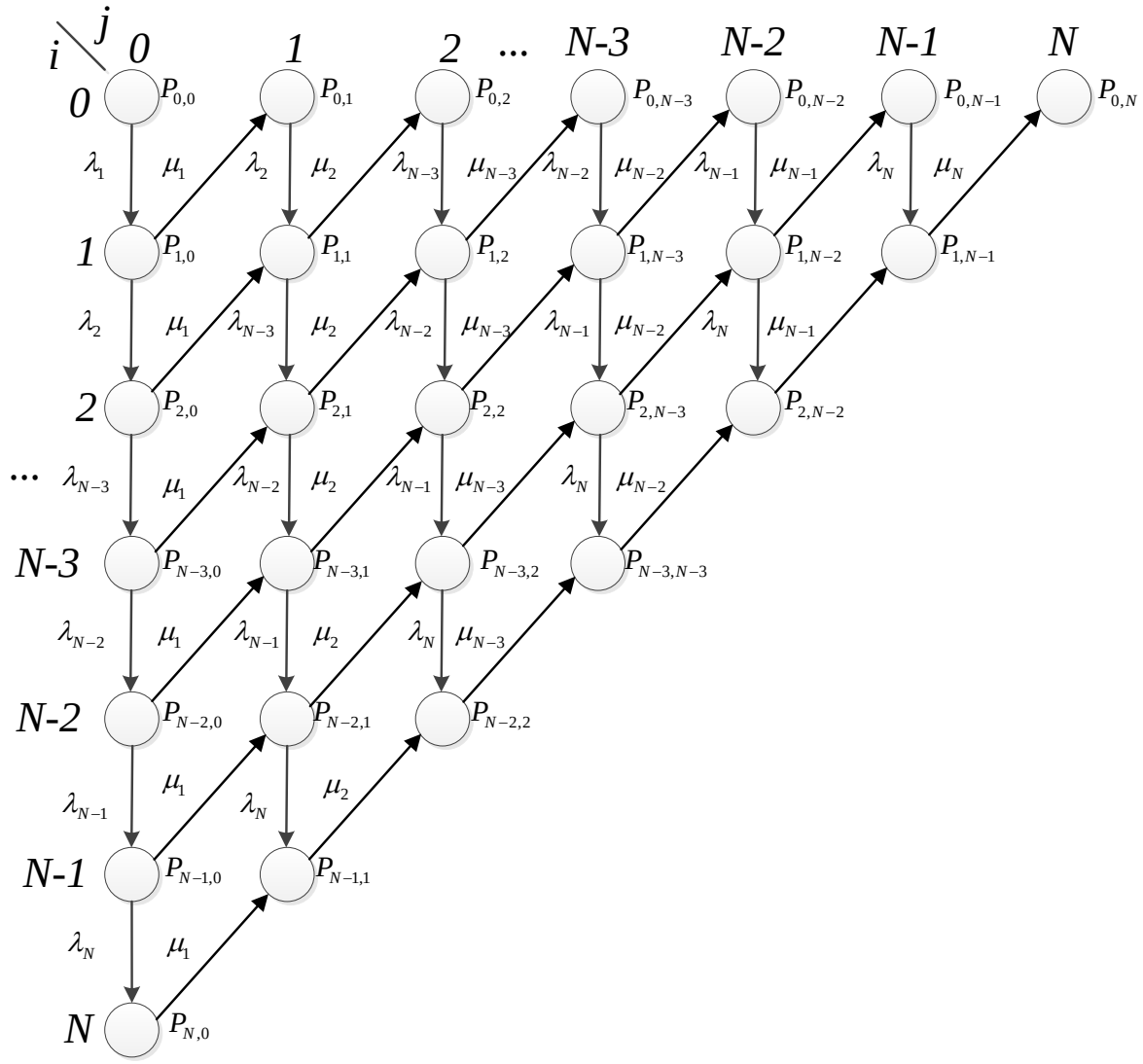


Рис. 1. Диаграмма переходов и состояний НСО

В качестве начальных условий выбирают обычно нахождение в состоянии $(0,0)$, то есть $P_{i,j}(0)=1-H(i+j)$. Для каждого момента времени t должно соблюдаться условие нормировки вида $\sum_{i=0}^{N}\sum_{j=0}^{N-i}P_{i,j}(t)=1$. Вероятность нахождения в НСО ровно i заявок в каждый момент времени $P_i(t)=\sum_{j=0}^{N-i}P_{i,j}(t)$ Значение вероятности обслуживания ровно j заявок $P_j(t)=\sum_{i=0}^{N-j}P_{i,j}(t)$, а значение вероятности обслуживания не менее q заявок может быть определено из выражения $P_q(t)=\sum_{j=q}^{N}P_j(t)$.

Марковский процесс, описывающий поведение НСО, имеет конечное число состояний. Есть начальное состояние и есть конечное (поглощающее) состояние. Все состояния невозвратные. Процесс однородный, не эргодический, для него не существует стационарного режима. Граф состояний и переходов для всех НСО ориентированный, плоский. Все пути простые и элементарные. Граф конечный, без петель, не имеет контуров. Существуют правила для любой НСО – невозможны переходы:

1) из состояния (i, j) в состояние $(i-k, j), k=\overline{1, N}$;
2) из состояния (i, j) в состояние $(i, j-k), k=\overline{1, N}$;
3) из состояния (i, j) в состояние (k, j), если $k > i+1$;
4) из состояния (i, j) в состояние (i, k), если $k > j+1$.

Алгоритм и программная реализация расчета вероятностей состояний НСО представлены в [20]. Алгоритм расчета вероятностей состояний НСО основывался на численном решении системы линейных однородных уравнений с постоянными коэффициентами при заданных начальных условиях методом типа Рунге – Кутты.

Введена система обозначения моделей НСО $G_k(i) / G_k(j) / N / L / P$, где G определяет закон распределения вероятностей временных интервалов между поступлениями заявок и их обслуживанием: M – экспоненциальный, E – Эрланга, H – гиперэкспоненциальный, C – Кокса; T – трасса поступления заявок в определенные моменты времени; k – число этапов в рассматриваемых E, H, C распределениях. Обозначения: (i) интенсивность поступления, зависит от номера заявки; (j) интенсивность обслуживания, зависит от номера заявки; k – число заявок в источнике; L – число каналов обслуживания; P – характеризует приоритетность; $I_{кр}$ – число заявок в очереди, после которых подключается резервный канал обслуживания или начинается обслуживание; t_α – время начала обслуживания; t_β – время окончания обслуживания; $\vec{l}$ – доступность каналов обслуживания; R – емкость накопителя очереди. В монографии [21] представлен ряд одноканальных НСО:

1) НСО с потерями (число мест в очереди меньше общего числа заявок) $M(i, R) / M(j) / N$;
2) НСО с двумя пакетами заявок $M(i) / M(j) / N_x$;
3) НСО с пороговым включением обслуживающего канала $M(i, I_{кр}) / M(j) / N$;
4) НСО с включением и выключением канала обслуживания в определенные моменты времени $M(i, t_\alpha) / M(j, t_\beta) / N$.

Там же можно найти многоканальные модели:

1) НСО с подключением и отключением второго канала обслуживания;
2) НСО с ненадежными каналами обслуживания;
3) НСО с доступностью каналов обслуживания и другие.

Выполнена аппаратная реализация разработанных моделей. Новизна которых подтверждена авторскими свидетельствами на изобретение [22–25].

2. Аппроксимация распределениями фазового типа

Для учета произвольного закона распределения вероятностей временных интервалов в моделях НСО проведено обобщение метода аппроксимации произвольной плотности распределения вероятностей, плотностью вида [26]:

$$f(t) \approx \sum_{i=1}^{k} \theta_i \mu_i e^{-\mu_i t}, \qquad (2)$$

где $f(t)$ – произвольная плотность распределения вероятностей; θ_i, μ_i, k – параметры аппроксимирующего распределения.

Известно, что плотности многих распределений, исключая некоторые (в частности, логарифмическое нормальное распределение), при определенных, обычно выполняемых условиях однозначно определяются своими начальными моментами. Потребовав равенства первых N начальных моментов плотности $f(t)$ и аппроксимирующей плотности, получим систему нелинейных алгебраических уравнений для определения неизвестных θ_i и μ_i:

$$\sum_{i=1}^{k} \frac{\theta_i}{\mu_i^{\,j}} = \frac{\nu_j}{j!}, j = \overline{0, N},$$

где ν_j – j-й начальный момент аппроксимируемой плотности распределения $f(t)$.

Если θ_i положительные числа, удовлетворяющие условиям $0 \le \theta_i \le 1$, $\sum_{i=1}^{k} \theta_i = 1$, то в правой части (2) плотность гиперэкспоненциального распределения. Коэффициент вариации такого распределения $\eta > 1$. Частный случай, когда μ_i равны между собой, а $\theta_i = \frac{1}{k}$ – экспоненциальное распределение $\eta = 1$.

Если $\theta_i = \prod_{l=1}^{k} \frac{\mu_l}{\mu_l - \mu_i}$; $l, i = \overline{1,k}$; $i \ne l$; $\mu_i \ne \mu_l$, то правая часть (2) представляет неоднородно-эрланговское распределение порядка k. Коэффициент вариации такого распределения $\frac{1}{\sqrt{k}} < \eta < 1$. В этом случае θ_i могут принимать отрицательное значение, однако условие $\sum_{i=1}^{k} \theta_i = 1$ должно сохраняться. Положив $\theta_i = \sum_{l=1}^{k} a_1 a_2 ... a_{l-1} b_l \prod_{n}^{l} 1 \frac{\mu_n}{\mu_n - \mu_i}$; $n \ne i$; $\mu_n \ne \mu_i$; $a_i + b_i = 1; i \ne l$, получим плотность распределения Кокса или гиперэрланговское распределение, как его иногда называют в литературе. Коэффициент вариации такого распределения $\eta > \frac{1}{\sqrt{k}}$.

Кокс показал [27], что параметры аппроксимирующего распределения могут быть комплексными, тогда чисто формально, можно исследовать процесс как марковский и составить уравнения Чепмена – Колмогорова обычным путем. В этом случае, несмотря на то, что значения вероятности, связанные с фиктивными фазами, могут быть комплексными, все же вероятности, связанные с реальными состояниями исследуемой системы, будут вещественными. Использование комплексно-сопряженных параметров позволяет аппроксимировать выше перечисленными распределениями произвольные плотности с коэффициентами вариации, находящимися в диапазоне $0 < \eta < \infty$. В качестве примера исследование влияния коэффициента вариации исходных распределений на полученные результаты надежности модели многопроцессорной вычислительной системы, полученные с помощью модели НСО, представлены в работах [28, 29].

3. Применение моделей НСО к оцениванию надежности программных средств и планированию их испытаний

Дальнейшее развитие модели НСО получили при их использовании для построения моделей надежности программных средств и планированию их испытаний.

Модели надежности программного обеспечения получили достаточно широкую проработку в научной литературе. При этом наибольшую известность получили модели Джелинского – Моранды, Мусы и Литлвуда. Систематическое описание этих и ряда других моделей надежности программ приводится в [30–33]. Авторами [2, 34] описываются нетрадиционные методы оценки надежности информационных систем с помощью вероятностного и надежностного подходов. В работах Половко А. М. и Гурова С. В. рассматриваются модели поведения программ, в которых учитываются интервалы работоспособного и неисправного состояний программы.

Сама по себе оценка надежностных характеристик программ представляет несомненный интерес. Однако следующий важный шаг, конечно, связан с выработкой практических рекомендаций по организации отладки программ, позволяющих улучшить характеристики процесса отладки. Здесь можно назвать работы по планированию испытаний сложных программных комплексов, например [35, 36]. В них делается существенное предположение о том, что характеристики надежности отдельных программных модулей известны.

Рассматривается вопрос обоснования стратегии отладки (отдельного модуля) программы на основе использования модели роста надежности программы, аналогичной модели Джелинского – Моранды [37]. Отметим, модели роста надежности программ часто используют вероятностные подходы и модели систем массового обслуживания с конечным и бесконечным числом каналов обслуживания и сетей массового обслуживания Джексона. К примеру, в работе [38] предлагается модель, основанная на Марковской модели систем массового обслуживания, с решением задачи появления и устранения ошибок в программе как Марковского процесса гибели и размножения с непрерывным временем и нахождением его характеристик. Недостатком названного решения является использование предположения о стационарности рассматриваемых процессов.

В основу предлагаемого решения предлагается положить базовую модель НСО, предложенную в настоящей статье. При соответствующей вероятностной интерпретации (аналогичной модели Джелинского – Моранды) модель позволяет описывать и рассчитывать характеристики надежности программы, а также обосновывать стратегию отладки программы. Модель в большей степени подходит для получения прогнозных оценок надежности отдельных программных модулей, число ошибок в которых может колебаться от единиц до двух-трех десятков.

Важным вопросом при моделировании надежности программного обеспечения является выбор исходных данных. Распространенным подходом [39, 40] является использование собранной в процессе разработки уже завершенных проектов статистики ошибок (интенсивности обнаружения ошибок при отладке, временные интервалы между моментами обнаружения ошибок). Однако, для учета особенностей конкретного проекта целесообразно также использовать его характеристики, влияющие на надежность – например, объектно-ориентированные метрики [32, 33, 41–44] (длина наибольшего пути иерархии наследования, взвешенное количество методов класса и др.). Метод планирования тестирования программ с использованием объектно-ориентированных метрик сложности предложен в [45].

3.1. Нестационарная модель надежности программных средств

Отладка программы представляет собой процесс, состоящий из двух этапов: тестирование (поиск ошибок) и разработка (устранение обнаруженных ошибок). Эти два типа деятельности, как правило, разделены и выполняются различными исполнителями (командами). Для упрощения будем считать, что при устранении ошибки новые ошибки не вносятся. Кроме того, считаем, что интенсивности обнаружения и исправления ошибок зависят от номера обнаруживаемой и устраняемой ошибки. Применение моделей НСО, описанных в первом разделе настоящей статьи, осуществляется интерпретацией потока заявок процессом нахождения ошибок при тестировании программных средств (ПС), а обслуживания заявок процессом исправления найденных ошибок. Показатели надежности ПС на примере базовой модели приведены ниже.

Вероятность $R_i(t)$ того, что в процессе испытаний было найдено ровно i ошибок, может быть вычислена по следующей формуле:

$$R_i(t) = \sum_{j=0}^{i} P_{i-j,j}(t). \qquad (3)$$

Тогда математическое ожидание числа обнаруженных ошибок $N_R(t)$ вычисляется по формуле:

$$N_R(t) = \sum_{i=1}^{N} iR_i(t). \qquad (4)$$

Вероятность $P_j(t)$ того, что в процессе испытаний было устранено ровно j ошибок, может быть вычислена по следующей формуле:

$$P_j(t) = \sum_{i=0}^{N-j} P_{i,j}(t). \quad (5)$$

Тогда математическое ожидание числа исправленных ошибок $N_p(t)$ вычисляется по формуле:

$$N_p(t) = \sum_{j=1}^{N} jP_j(t). \quad (6)$$

Соответственно, среднее число $N_{RL}(t)$ ошибок, не обнаруженных к моменту времени t, и среднее число $N_{PL}(t)$ ошибок, не исправленных к моменту времени t, определяются по формулам:

$$N_{RL}(t) = N - N_R(t), \quad (7)$$

$$N_{PL}(t) = N - N_P(t). \quad (8)$$

Вероятность безотказной работы $P(t,\tau)$ в течение интервала τ после испытаний в течение времени t:

$$P(t,\tau) = 1 - \sum_{j=1}^{N-1} P_j(t)(1 - e^{-\lambda_{j+1}\tau}). \quad (9)$$

Функция $F_i(t)$ распределения времени устранения i ошибок:

$$F_i(t) = \sum_{l=i}^{N} P_l(t), i = \overline{0,N}. \quad (10)$$

Модель позволяет оценить время испытаний $T_{N_{TP}P_{TP}}$, которое требуется, чтобы найти и исправить N_{TP} ошибок с заданной вероятностью P_{TP}, как:

$$T_{N_{TP}P_{TP}} = t : F_{N_{TP}}(t) \geq P_{TP}. \quad (11)$$

Вероятность безотказной работы $P(t,\tau)$ в течение интервала τ после испытаний в течение времени t:

$$P(t,\tau) = 1 - \sum_{j=1}^{N-1} \int_0^{\tau} P_{0,0,0,j}(t)(1 - e^{-\lambda_{j+1}x})(1 - e^{-\lambda'_{j+1}(\tau-x)})dx -$$

$$- \sum_{j=0}^{N-1} P_{0,1,0,j}(t)(1 - e^{-\lambda'_{j+1}\tau}). \quad (12)$$

Описанная модель надежности может применяться на различных этапах жизненного цикла. На этапе проектирования могут быть получены первые грубые прогнозы надежности на основе рассчитанных к этому моменту метрик сложности проекта и статистики о процессе обнаружения и исправления ошибок, собранных ранее при разработке подобных проектов. Такая оценка может быть уточнена позже на этапе тестирования, когда уже получены более точные данные о распределении интервалов времени обнаружения и исправления программных ошибок. Методика подготовки исходных данных для проведения моделирования представлена в работах [46–47]. Применение немарковских моделей НСО для расчета показателей надежности ПС нашло отражение в работах [48–53].

3.2. Варианты стратегии отладки

С точки зрения распределения времени между этапами тестирования и исправления ошибок можно рассматривать разные стратегии отладки программ [54].

Этап тестирования продолжается, пока не будет обнаружена одна ошибка. После обнаружения ошибки тестирование приостанавливается, ошибка исправляется. После исправления ошибки вновь начинается тестирование. Количество состояний системы:

$$k = 2N + 1. \tag{13}$$

Система дифференциальных уравнений базовой модели (1) примет следующий вид:

$$\begin{aligned}\frac{dP_{i,j}(t)}{dt} &= -\delta(N - j - iN)P_{i,j}(t)\lambda_{j+1} + \\ &+\delta(i)(P_{i-1,j}(t)\lambda_{j+1} - P_{i,j}(t)\mu_{j+1}) + \\ &+\delta(j(1-i))(P_{i+1,j-1}(t)\mu_j),\end{aligned} \tag{14}$$

где $i = \overline{0,1};\ j = \overline{0, N-i}$.

Этап тестирования продолжается, пока не будут обнаружены все ошибки. Затем обнаруженные ошибки исправляются. Количество состояний системы рассчитывается по формуле (13). Система дифференциальных уравнений (1) примет следующий вид:

$$\begin{aligned}\frac{dP_{i,j}(t)}{dt} &= -\delta(N - i - jN)P_{i,j}(t)\lambda_{j+1} + \\ &+\delta(i(1-j))P_{i-1,j}(t)\lambda_i - \\ &-\delta(i(i + j + 1 - N))P_{i,j}(t)\mu_{j+1} + \\ &+\delta(j(i + j + 1 - N))P_{i+1,j-1}(t)\mu_j,\end{aligned} \tag{15}$$

где $j = 0; i = \overline{0, N};\ j = \overline{1, N}; i = N - j$.

Этап тестирования продолжается, пока не будут обнаружены k ошибок. После обнаружения k ошибок тестирование приостанавливается, ошибки полностью исправляются. Затем тестирование продолжается. Количество состояний системы рассчитывается по формуле (13). Система дифференциальных уравнений (1) примет следующий вид:

$$\begin{aligned}\frac{dP_{i,j}(t)}{dt} &= \delta(1 - j \bmod k) \times \\ &\times(\delta(i)P_{i-1,j}(t)\lambda_{i+j} - \delta(k-i)\delta(N-i-j)P_{i.j}(t)\lambda_{i+j+1}) - \\ &-\delta(i)\delta(1-(i+j) \bmod k) + \delta(1-(i+j) \bmod N)) \times \\ &\times P_{i.j}(t)\mu_{j+1} + \delta(i+j)\delta(N \bmod k - i) \times \\ &\times(\delta(1-(i+j) \bmod k)\delta(k-i) + \\ &+\delta(1-(i+j) \bmod N)\delta(N \bmod k - i))P_{i.j}(t)\mu_j,\end{aligned} \tag{16}$$

где $i \bmod j$ – остаток от деления i / j.

3.3. Результаты прикладных расчетов и обоснование стратегии отладки

При практическом использовании модели надежности $M(i)/M(j)/N$ для расчетов в качестве исходных данных могут быть использованы статистические данные по интервалам времени между обнаружением и исправлением ошибок. Общее число ошибок N при этом неизвестно, однако может быть спрогнозировано, исходя из эмпирических соображений на основе опыта разработки похожих проектов, с использованием существующих прогнозных моделей (Джелинского – Моранды и подобных).

Представляет интерес сравнение рассмотренных выше стратегий отладки. Данные по обнаруженным и устраненным ошибкам приведены в таблице 1.

Таблица 1 – Исходные данные о программных ошибках для расчетов

Время тестирования (час)	Число ошибок	Время устранения (час)	Интенсивность обнаружения (час$^{-1}$)	Интенсивность устранения (час$^{-1}$)
4	25	10	6,25	2,5
4	19	4	4,75	4,75
2	3	2	1,5	1,5
2	1	1	0,5	1
1	1	1	1	1

Используя представленные выше формулы можно определить математическое ожидание числа обнаруженных ошибок и математическое ожидание еще не исправленных ошибок для предлагаемой модели надежности – системы $M(i)/M(j)/N$ для различных стратегий.

Приведенные выше результаты могут быть использованы для прогнозирования числа остающихся в программе ошибок к моменту времени τ с момента начала отладки. Для этого в качестве исходных данных должна использоваться достаточно полная статистика, собранная в процессе разработки большого числа проектов. Пример такого прогноза для системы приведен в таблице 2.

Таблица 2 – Прогноз числа ошибок, не исправленных при отладке

Время τ (час)	Математическое ожидание числа не исправленных ошибок		
	Стратегия 1	Стратегия 2	Стратегия 3 (k=10)
10	29,2	46,2	31,3
15	19,4	38,1	24,3
20	8,5	24,7	18,8
25	2,1	9,2	6,6
30	0,5	1,7	1,1

Так как предложенная модель позволяет найти функции $F_i(t)$ распределения времени устранения i ошибок, возможно, оценить требуемое время отладки τ_{mp} для достижения необходимого уровня надежности, заданного в требованиях к программе (например, техническом задании) в виде вероятности $F_N(t=\tau_{mp}) > P_{mp}$ отсутствия ошибок в программе. Это время не зависит от выбора стратегии тестирования. Пример оценки времени τ_{mp} для системы $M(i)/M(j)/N$ приведен в таблице 3.

Таблица 3 – Прогноз требуемого времени отладки по заданной вероятности отсутствия ошибок

Вероятность P_{mp} отсутствия ошибок	Требуемое время отладки τ_{mp} (час)
0,8	31,4
0,9	33,3
0,95	35,0
0,99	38,6

Также модель надежности $M(i)/M(j)/N$ позволяет определить время тестирования до достижения с заданной вероятностью P_{mp} требуемого уровня надежности, определенного как количество N_{mp} остающихся в программе ошибок (из прогнозного общего числа ошибок N). Это может потребоваться в случае высокой стоимости процесса отладки или критичных сроков выпуска программы, например, при использовании Agile подхода к разработке: $F_{Nmp}(t=\tau_{mp}) > P_{mp}$. Пример оценки требуемого времени отладки τ_{mp} при $P_{mp} = 0{,}95$ для разного числа ошибок, выполненный с помощью предлагаемой модели надежности $M(i)/M(j)/N$, приведен в таблице 4.

Таблица 4 – Прогноз требуемого времени отладки по допустимому числу ошибок

Количество ошибок N_{mp}	Время отладки τ_{mp} (час)		
	Стратегия 1	Стратегия 2	Стратегия 3 (k=10)
5	25,5	31,3	30,1
3	26,4	31,8	30,5
1	32,6	33,8	32,5

Сравнение стратегий отладки показывает, что их выбор не влияет на распределение времени отладки до исправления всех N ошибок. Однако распределение времени обнаружения и исправления $N_{mp} < N$ ошибок сильно зависит от выбранной стратегии отладки. Так, стратегия 2 обеспечивает наименьшее время обнаружения ошибок, стратегия 1 наименьшее время исправления ошибок. Результаты расчетов стратегий 2 и 3 отчетливо выделяют различные этапы отладки: тестирование (кривая среднего числа найденных ошибок растет, а

среднего числа не исправленных близка к константе) и исправление ошибок (кривые ведут себя обратным образом).

3.4. Обобщенная модель надежности ПС

Обобщенная модель роста надежности [29, 58] представляет собой систему обслуживания с неэкспоненциальным распределением длин интервалов времени обнаружения и исправления ошибок, которое аппроксимируется двухэтапным распределением Кокса. Такая модель обозначается $C_2(i)/C_2(j)/N$.

Состояние (i,k,l,j) системы $C_2(i)/C_2(j)/N$ в каждый момент времени характеризуется количеством обнаруженных (еще не исправленных) ошибок $i(i=\overline{0,N})$, исправленных ошибок $j(j=\overline{0,N-i})$, фазой $k(k=\overline{0,\delta(N-i-j)})$ распределения Кокса длины интервалов времени между моментами обнаружения ошибок, фазой $l(l=\overline{0,\delta(i)})$ распределения Кокса времени исправления ошибок. Переход из состояния $(i,0,l,j)$ в состояние $(i+1,0,l,j)$ означает, что при тестировании была обнаружена ошибка под номером $(i+j+1)$. Переход из состояния $(i,k,0,j)$ в состояние $(i-1,k,0,j+1)$ означает, что была исправлена ошибка под номером $(j+1)$. Общее число состояний N_c вычисляется по формуле:

$$N_c = 2N^2 + 2N + 1.$$

Диаграмма модели $C_2(i)/C_2(j)/N$, представленной в [29] описывается системой дифференциальных уравнений:

$$\begin{aligned}
&\frac{dP_{i,k,l,j}(t)}{dt} = -\delta(1-k)\delta(N-i-j)P_{i,k,l,j}(t)\lambda_{i+j+1} - \\
&-\delta(k)P_{i,k,l,j}(t)\lambda'_{i+j+1} + \delta(k)P_{i,0,l,j}(t)p_{i+j+1}\lambda_{i+j+1} + \\
&+\delta(1-k)\delta(i)\left(P_{i,1,l,j}(t)\lambda'_{i+j} + P_{i-1,0,l,j}(t)(1-p_{i+j})\lambda_{i+j}\right) - \\
&-\delta(i)\delta(1-l)P_{i,k,l,j}(t)\mu_{j+1} - \delta(l)P_{i,k,l,j}(t)\mu'_{j+1} + \\
&+\delta(l)P_{i,k,0,j}(t)r_{j+1}\mu_{j+1} + \\
&+\delta(1-l)\delta(j)\left(P_{i+1,k,1,j-1}(t)\mu'_{j} + P_{i+1,k,0,j-1}(t)(1-r_j)\mu_j\right),
\end{aligned} \tag{17}$$

где $i=\overline{0,N}; j=\overline{0,N-i}; k=\overline{0,\delta(N-i-j)}; j=\overline{0,\delta(i)}$.

Для каждого момента времени *t* должно соблюдаться условие нормировки вида $\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{k=0}^{\delta(N-i-j)}\sum_{l=0}^{\delta(i)}P_{i,k,l,j}(t)=1$.

Задав начальные условия к системе в виде:

$$P_{i,k,l,j}(0) = \begin{cases} 0, \text{ если } i+k+l+j \neq 0; \\ 1, \text{ если } i+k+l+j = 0, \end{cases},$$

можно найти численное решение соответствующей задачи Коши для произвольного значения *t*.

Вероятность $R_i(t)$ того, что в процессе отладки было найдено ровно *i* ошибок, может быть вычислена по следующей формуле:

$$R_i(t) = \sum_{j=0}^{i} \sum_{k=0}^{\delta(N-i-j)} \sum_{l=0}^{\delta(i)} P_{i-j,k,l,j}(t).$$

Вероятность $P_j(t)$ того, что в процессе отладки было устранено ровно j ошибок, может быть вычислена по формуле:

$$P_j(t) = \sum_{i=0}^{N-j} \sum_{k=0}^{\delta(N-i-j)} \sum_{l=0}^{\delta(i)} P_{i,k,l,j}(t).$$

Вероятность безотказной работы $P(t,\tau)$ в течение интервала τ после отладки в течение времени t:

$$P(t,\tau) = 1 - \sum_{j=0}^{N-1} \int_0^{\tau} P_{0,0,0,j}(t)\left(1 - e^{-p_{j+1}\lambda_{j+1}x}\right)\left(1 - e^{-\lambda'_{j+1}(\tau-x)}\right)dx -$$

$$-\sum_{j=0}^{N-1} P_{0,0,0,j}(t)\left(1 - e^{-p'_{j+1}\lambda_{j+1}\tau}\right) - \sum_{j=0}^{N-1} P_{0,1,0,j}(t)\left(1 - e^{-\lambda'_{j+1}\tau}\right).$$

Результаты моделирования надежности с использованием описанных моделей могут быть использованы при планировании тестирования многомодульных ПС.

3.5. Комплекс программ расчета надежности и планирования испытаний ПС

На основе нестационарных моделей разработан комплекс программ расчета надежности и планирования испытаний ПС [55]. Программы написаны на языке Matlab и позволяют рассчитывать основные показатели надежности модулей ПС: среднее число неисправленных ошибок, вероятность отсутствия ошибок, вероятность того, что не исправлено k ошибок, вероятность безотказной работы.

Входными данными для расчета надежности являются: общее число ошибок, начальные моменты распределений длин временных интервалов обнаружения и исправления ошибок (в зависимости от используемой модели задаются от одного до трех начальных моментов), среднее время функционирования модулей и граф переходов между компонентами ПС. Используя рассчитанные показатели надежности и данные о структуре ПС, выполняется оценка надежности отдельных модулей и всего ПС, обоснование стратегий и планирование испытаний многомодульных ПС.

В комплексе программ реализованы следующие модели надежности ПС: $M_i / M_j / N$, $H_2(i) / M(j) / N$, $M(i) / E_2(j) / N$, $E_2(i) / E_2(j) / N$, $C_2(i) / C_2(j) / N$. Для всех моделей реализованы расчеты показателей надежности при использовании различных стратегий испытаний: как описанных выше, так и более сложных.

Программы содержат следующие основные функции:

1) *ErlangApproximation, CoxApproximation, H2Approximation* – функции расчета параметров аппроксимации двухэтапным распределением Эрланга (неоднородным), Кокса, гиперэкспоненциальным распределени-

ем соответственно. Параметры: первые два (три) начальных момента исходного распределения;

2) *StateQuantity* – функция расчета числа состояний модели. Параметр: общее число ошибок;
3) *PIndex* – функция расчета порядкового номера (индекса) состояния модели. Параметры: вектор, описывающий состояние системы (число обнаруженных, но не исправленных ошибок, число исправленных ошибок, фаза обнаружения и исправления ошибок);
4) *PIndex2ij* – функция выполняет обратное преобразование. Параметр: индекс состояния;
5) *DeSMO* – функция составления системы дифференциальных уравнений. Параметры: векторы интенсивностей обнаружения и исправления ошибок, матрица состояний (содержит индексы состояний, для которых производится расчет, это необходимо для моделирования различных стратегий испытаний), время моделирования, вектор вероятностей состояний системы в текущий момент времени;
6) *SummP* – функция расчета вероятностных характеристик процесса испытаний и показателей надежности (функции распределения времени исправления заданного числа ошибок, вероятности обнаружения и исправления заданного числа ошибок, среднее число обнаруженных и исправленных ошибок). Параметры: вектор вероятностей пребывания в различных состояниях системы в различные моменты времени моделирования;
7) *CalculateStates* – функция решения системы дифференциальных уравнений (вызов функций *ode45* и *DeSMO*) и расчета показателей надежности (вызов функции *SummP*);
8) *PFaultLess* – функция расчета вероятности безотказной работы в течение заданного времени после проведения испытаний. Параметры: вектор вероятностей пребывания в различных состояниях системы в моменты времени моделирования, время функционирования, интенсивности обнаружения ошибок;
9) *Main* – основная функция, готовит исходные данные, обращается к функции *CalculateStates* для решения системы дифференциальных уравнений и представляет результаты (графики, выходной файл).

Для решения системы дифференциальных уравнений использована стандартная функция *ode45* для численного интегрирования систем обыкновенных дифференциальных уравнений с использованием формул Рунге-Кутта 4-го и 5-го порядка.

Для каждой реализованной модели (как экспоненциальной, так и для общих моделей) существует возможность выполнить расчет надежности для различных стратегий испытаний.

Помимо описанных стратегий комплекс программ моделирует и более сложные стратегии испытаний. Для этого система дифференциальных уравнений такой модели генерируется функцией *DeSMO* на основе параметра – матрицы, описывающей возможные «пути» обнаружения и исправления ошибок в

графах переходов соответствующей модели и возможные варианты организации взаимодействия команд разработки и тестирования при испытаниях – последовательная, параллельная, комбинированная организация работы.

Для реализации модели планирования испытаний использован пакет символьных вычислений Matlab Symbolic Math Toolbox. Программа содержит следующие основные функции:

1) *CalculatePStay* – функция расчета вероятностей выполнения модулей ПС путем решения системы линейных уравнений баланса. Параметры: матрица вероятностей переходов между модулями и средние значения времени исполнения модулей;
2) *ProbabilityNoFaultSPK* – функция расчета вероятности безотказной работы многомодульного ПС в течение заданного времени после испытаний. Параметры: матрица вероятностей переходов между модулями и матрица безусловных вероятностей безотказной работы модулей;
3) *Inverse, Minor, Cofactor, Moments, MultyMV* – вспомогательные функции для выполнения символьных вычислений над матрицами (нахождение обратной матрицы, минора, алгебраического дополнения и др.).

Для обоснования достоверности расчетов выполнена проверка полученных результатов с использованием различных разработанных моделей НСО [48, 56–57]. Схема выполнения взаимной проверки результатов моделирования иллюстрируется на рис. 2. Реализованные модели показаны по степени общности, от базовой экспоненциальной (крайняя левая строка) до наиболее общей, с использованием двухэтапного распределения Кокса (крайняя правая строка).

Рис. 2. Схема взаимной проверки результатов моделирования

Модели, расположенные на рисунке правее, являются более общими и используются для проверки результатов частных моделей, расположенных на рисунке, соответственно, левее. Взаимная проверка моделей НСО показала совпадение полученных результатов.

3.6. Модели НСО надежности программных средств, учитывающие возможную вероятность обнаружения ошибок при тестировании и состоятельность используемых тестов

В работе [59] предложено дальнейшее развитие моделей НСО применительно к оцениванию характеристик надежности ПС и стратегий их испытаний.

Распределения временных интервалов между моментами обнаружения ошибок во время тестирования описываются экспоненциальными законами. Предполагается, что ПС могут содержать не более N ошибок, которые обнаруживаются с интенсивностями $\{\omega_1\lambda_1, \omega_2\lambda_2, ..., \omega_N\lambda_N\}$, зависящими от номера ошибки и значений соответствующих вероятностей их обнаружения. В случае подтверждения отсутствия ошибки, осуществляется переход по соответствующим дугам, с интенсивностями $\{(1-\omega_1)\lambda_1, (1-\omega_2)\lambda_2, ..., (1-\omega_N)\lambda_N\}$. Время устранения всех обнаруженных ошибок распределено по экспоненциальному закону с интенсивностью μ.

Состояния (i, j) такой системы в каждый момент времени будем характеризовать количеством обнаруженных, но ещё не исправленных ошибок i и числом устранённых (сумма отсутствующих и исправленных ошибок) $j (i, j = 0,1,..., N)$. Вероятности пребывания системы в этих состояниях обозначается $P_{i.j}(t)$.

Переход системы из состояния (i, j) в состояние $(i+1, j)$ означает, что при тестировании была обнаружена $(i + j + 1)$-я ошибка. Переход из состояния (i, j) в состояние $(i, j+1)$ означает, что при тестировании подтверждено отсутствие ошибки. Для последних состояний в каждом столбце графа переход системы из состояния (i, j) в состояние $(i-1, j+1)$ означает, что была исправлена обнаруженная ошибка с номером $(j+1)$.

В таком представлении общее число состояний системы вычисляется по формуле:

$$N_c = (N+1)(N+2)/2.$$

Процессы в рассматриваемой Марковской цепи описываются системой из N_c дифференциальных уравнений:

$$\begin{aligned}\frac{\mathrm{d}P_{i,j}(t)}{\mathrm{d}t} &= \delta(i)P_{i-1,j}(t)\omega_{i+j,}\lambda_{i+j} - \delta(N-i-j)P_{i,j}(t)\lambda_{i+j+1} + \\ &+\delta(j)P_{i,j-1}(t)(1-\omega_{i+j})\lambda_{i+j} + \\ &+\delta((i+j+1-N)j)P_{i+1,j-1}(t)\mu - \delta((i+j+1-N)i)P_{i,j}(t)\mu, \end{aligned} \quad (18)$$

где $\delta(m) = \begin{cases} 1, \text{ если } m > 0; \\ 0, \text{ если } m \le 0. \end{cases}$ функция Хевисайда; $i = 0,\ 1,\ \ldots,\ N$; $j = 0,1,\ \ldots,\ N-i$.

Для каждого момента времени t должно соблюдаться условие нормировки вида $\sum_{i=0}^{N}\sum_{j=0}^{N-i} P_{i,j}(t) = 1.$ Задав начальные условия к системе уравнений в виде:

$$P_{i,j}(0) = \begin{cases} 1, \text{ если } i+j = 0; \\ 0, \text{ если } i+j \ne 0, \end{cases}$$

можно найти численное решение соответствующей задачи Коши для произвольного значения времени *t*.

Используя решение предложенной системы (18), можно получить ряд важных вероятностных показателей процесса испытаний и состояния ПС (так же как и для базовой модели).

Вероятность $R_i(t)$ того, что в процессе испытаний было найдено ровно i ошибок, может быть вычислена по следующей формуле (сумма найденных, но не исправленных и устранённых ошибок):

$$R_i(t) = \sum_{j=0}^{i} P_{i-j,\,j}(t), i = \overline{0, N}.$$

Тогда математическое ожидание числа найденных ошибок $N_R(t)$ вычисляется по формуле:

$$N_R(t) = \sum_{i=1}^{N} iR_i(t).$$

Вероятность $P_j(t)$ того, что в процессе испытаний было устранено (исправлено или отсутствовало) ровно j ошибок, может быть вычислена по следующей формуле:

$$P_j(t) = \sum_{i=0}^{N-j} P_{i,\,j}(t),\ j = \overline{0, N}.$$

Математическое ожидание числа устранённых ошибок $N_P(t)$ вычисляется по формуле:

$$N_P(t) = \sum_{j=1}^{N} jP_j(t).$$

Соответственно, среднее число $N_{RL}(t)$ ошибок, не обнаруженных к моменту времени t, и среднее число $N_{PL}(t)$ ошибок, не устранённых к моменту времени t, определяются по формулам:

$$N_{RL} = N - N_R(t);$$
$$N_{PL} = N - N_P(t).$$

Вероятность безотказной работы $P(t,\tau)$ в течение интервала τ после испытаний в течение времени t:

$$P(t,\tau) = 1 - \sum_{j=0}^{N-1} R_j(t)(1 - e^{-\lambda_{i+1}\tau}).$$

Функция $F_i(t)$ распределения времени устранения не менее i ошибок:

$$F_i(t) = \sum_{l=i}^{N} P_l(t), i = \overline{0, N}.$$

Модель позволяет оценить время отладки $T_{N_{\text{ТР}}, P_{\text{ТР}}}$,которое требуется, чтобы устранить $N_{\text{ТР}}$ ошибок с заданной вероятностью $P_{\text{ТР}}$, как:

$$T_{N_{\text{ТР}}, P_{\text{ТР}}} = t \mid F_{N_{\text{ТР}}}(t) \geq P_{\text{ТР}}.$$

Подобные модели НСО применительно к расчету показателей надежности ПС и стратегий их тестирования при общих предположениях о законах распределения временных интервалов тестирования и устранения ошибки представлены в работах [58, 60–61].

4. Начальный подход к построению моделей НСО и расчету их вероятностей состояний

Во всех перечисленных выше публикациях настоящей статьи при построении моделей НСО использовался следующий подход:

- определялись параметры, характеризующие состояния модели НСО;
- строилась диаграмма переходов и состояний;
- для каждого состояния записывалось уравнение Чепмена – Колмогорова;
- выводилось общее уравнение системы однородных дифференциальных уравнений Чепмена – Колмогорова;
- задавались начальные условия для решения соответствующей задачи Коши;
- для решения системы однородных дифференциальных уравнений использовались методы численного решения типа Рунге – Кутты с заданным шагом интегрирования.

5. Численно-аналитический метод расчета моделей нестационарных систем обслуживания

На основе результатов работ [45–54] разработан комплекс программ [55] расчёта надежности и планирования испытаний программных средств, в котором системы обыкновенных дифференциальных уравнений (ОДУ) для различных моделей нестационарных систем обслуживания решаются с помощью численных методов. Популярным программным инструментом, например, является Matlab, предлагающий несколько процедур для решения систем ОДУ. Основными недостатками традиционных численных методов являются накапливаемая на каждом шаге интегрирования погрешность и время работы алгоритма. Причём, вычислительная погрешность может привести и к отсутствию физического смысла получаемого решения.

В частности, для решения систем, которые будут описаны в следующих разделах, была использована процедура *ode45* из пакета Matlab (при настройках по умолчанию), основанная на паре вложенных методов Рунге—Кутты порядков 4 и 5, разработанных Э. Фельбергом [62]. Глобальная погрешность этой процедуры имеет порядок $O(h^5)$, где h – максимальная величина шага, в данном случае, по времени. Положив число заявок на обслуживание, например равным 100 (что даёт 5151 дифференциальное уравнение в системе Чепмена – Колмогорова), при интенсивности поступления всех заявок 1, а интенсивности обработки 2, было замечено, что многие состояния имели отрицательную вероятность, в частности, вероятность состояния номер 5087 стала $-9,2719644052270 \cdot 10^{-9}$. Увеличение же точности расчётов значительно увеличило время работы программы.

В то же время, желание учёта большего числа факторов реальных процессов, подлежащих моделированию, приводит к увеличению числа уравнений Чепмена – Колмогорова относительно вероятностей состояний. Время числен-

ного решения такого рода систем уравнений, как показывают эксперименты, экспоненциально зависит от общего числа состояний систем обслуживания. Это накладывает серьезные ограничения на разрабатываемые модели НСО.

Высказанные соображения потребовали разработки численно-аналитического метода расчета вероятностей состояний моделей НСО, рассматриваемых в настоящей работе.

Одним из таких методов является численно-аналитический метод, предлагаемый в работах [63–64].

5.1. Алгоритм нумерации состояний НСО

Рассмотренная выше система ОДУ (1) является линейной однородной системой уравнений. Она может быть представлена в матричной форме:

$$x(t) = Ax(t), \qquad (19)$$

где $x(t)$ – вектор неизвестных функций размерности K; A — квадратная матрица.

Для указанной системы существует явное аналитическое решение, если только известны собственные числа матрицы A. Очевидным является тот факт, что в случае треугольного вида матрицы A (для определённости матрица нижнетреугольная) её собственные числа выписаны в явном виде на диагонали. Таким образом, аналитическое решение системы вида (19) можно найти, если только матрица A — треугольная.

Если пронумеровать состояния по возрастанию числа обработанных заявок, а внутри этих групп по возрастанию числа поступивших заявок, то ни одно из состояний полученного списка не будет иметь зависимости от последующих. На рис. 1 это будет выглядеть как нумерация сверху вниз с перемещением по столбцам слева направо. Соответственно, полученная матрица A системы (19) будет нижнетреугольной.

Система (19), учитывая треугольность матрицы A:

$$\begin{cases} \dfrac{dx_1}{dt} = a_{11}x_1; \\ \dfrac{dx_2}{dt} = a_{21}x_1 + a_{22}x_2; \\ \dfrac{dx_i}{dt} = a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ii}x_i; \\ \dfrac{dx_K}{dt} = a_{K1}x_1 + a_{K2}x_2 + \ldots + a_{KK}x_K; \end{cases} \qquad (20)$$

где a_{ij} — интенсивность перехода из состояния с номером j в состояние с номером i (в векторе $x(t)$).

5.2. Решение системы ОДУ

Решение системы (20) разбивается на последовательное решение скалярных уравнений, первое из которых будет однородным, а все последующие бу-

дут включать в себя решения предыдущих уравнений в качестве неоднородности. Это позволяет сформулировать следующий алгоритм нахождения решения.

Решением первого уравнения (20) очевидно является:

$$x_1(t) = x_1(0)e^{a_{11}t}, \tag{21}$$

и ему соответствует фундаментальное решение $x_1(t) = e^{a_{11}t}$.

Предполагается, что решения первых $i-1$ уравнений найдены в форме:

$$x_j(t) = \sum_{w=1}^{j} k_{iw} x_w(t), \;\; j = \overline{1, i-1}, \tag{22}$$

причём $x_j(t) = t^{Vj-1}e^{a_{jj}t}$, где Vj — кратность собственного числа a_{jj} в системе первых j уравнений. Тогда, после подстановки (22) в i - е уравнение системы (20), последнее можно записать в виде

$$\frac{dx_i}{dt} = a_{ii}x_i(t) + \sum_{j=1}^{i-1} b_{ij}x_j(t), \tag{23}$$

где $b_{ij} = \sum_{w=j}^{i-1} a_{iw}k_{wj}(t)$. Решение (23) будет иметь вид:

$$x_i(t) = \sum_{j=1}^{i} k_{ij}x_j(t),\; x_i(t) = t^{Vj-1}e^{a_{ii}t}. \tag{24}$$

Пусть a_{ii} — собственное число, кратность которого в системе первых i уравнений равна v_i. Собственные числа $a_{i_1i_1} = a_{i_2i_2} = \cdots = a_{i_{v_i}i_{v_i}}$ пронумерованы в порядке нахождения на диагонали матрицы A. Тогда:

$$k_{ii_{v_i}} = \frac{b_{ii_{v_i-1}}}{v_i}, v_i > 1. \tag{25}$$

Коэффициенты k_{ij} при тех $x_j(t)$, для которых $a_{ii} \neq a_{jj}$:

$$k_{ij} = \begin{cases} \dfrac{b_{ij} - k_{iw}(v_j - i)}{a_{jj} - a_{ii}}, \; v_j > 1, \\ \dfrac{b_{ij}}{a_{jj} - a_{ii}}, \; v_j = 1, \end{cases} \tag{26}$$

где w такая, что $a_{ww} = a_{jj}$ и $v_w = v_j - 1$.

После нахождения всех коэффициентов (25) и (26), через начальные данные оставшиеся находятся по формуле:

$$k_{ii_1} = x_i(0) - \sum_{\substack{j=1 \\ v_j=0}}^{i-1} k_{ij}. \tag{27}$$

Следует отметить, что нахождение решения системы ОДУ в данном случае, в отличие от численного метода, дающего конечный набор точек, представляет собой построение процедуры, позволяющей определить вероятности состояний НСО в произвольный момент времени. Это, во-первых,

даёт возможность вывести решение ОДУ с произвольной степенью детальности, а во-вторых, скорость нахождения решения в любой сколь угодно удалённый момент времени одинакова и не требует расчёта многих предыдущих временных состояний, как это потребовалось бы численному методу, имеющему к тому же и методическую погрешность, оценка которой дополнительно увеличивает сложность решения.

5.3. Особенности программной реализации метода

Описанный в предыдущем параграфе метод был реализован на объектно-ориентированном языке программирования Java [66]. Первая версия реализации данного метода значительно выигрывала в скорости у Matlab. В таблице 5 приведено сравнение времени работы методов. Для всех заявок была установлена интенсивность поступления $\lambda = 1$, интенсивность обработки $\mu = 2$, рассчитывались вероятности состояний для значений $t = 100$.

Таблица 5 – Сравнение времени работы методов

Количество заявок (N)	Время расчёта на Java, мс	Время расчёта в Matlab, мс
3	1	24
5	3	29
10	21	49
15	22	60

Тестирование метода при больших значений N показало, что используемая реализация непригодна к практическому применению, в связи с появляющейся погрешностью. Это объясняется особенностями хранения чисел с плавающей точкой в памяти ЭВМ. Стандартный тип хранения таких данных в Java — тип *double*, размерность которого равна 64 бита. На всех этапах алгоритма происходит округление переменных, вследствие чего накапливается погрешность и при больших N эта погрешность значительно возрастает.

Для борьбы с погрешностью вычислений было принято решение использовать числа с практически неограниченной размерностью, реализованные в классе *BigDecimal* [65]. Эти числа представляют собой сочетание двух значений. Первое — неограниченное значение строкового типа (*String, S*), второе — 32-битное целое число (*Integer, I*). Таким образом, значение числа типа *BigDecimal* представляется в виде: $s \cdot 10^I$.

Для контроля погрешности в программу была добавлена возможность задания точности чисел, то есть определения номера десятичного знака после запятой, который округляется в арифметических операциях и все числа правее которого отбрасываются. Необходимо учитывать тот факт, что чем большая точность чисел используется, тем больше времени требуется на совершение арифметических операций с ними и, соответственно, тем дольше производится расчёт.

В классе *BigDecimal* реализованы все необходимые в аналитическом методе арифметические операции кроме нахождения экспоненты. Данная функциональность была реализована в ходе разработки, в её основе лежит ряд Тейлора:

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}. \quad (28)$$

После реализации алгоритма с числами типа *BigDecimal*, его точность и, к сожалению, время работы значительно возросли. Так, например, при $N = 100$, $\lambda = 1$, $\mu = 2$ с использованием точности чисел в 1000 знаков расчёт вероятностей состояний занимал полтора часа.

5.4. Оптимизация реализации численно-аналитического метода

В ходе анализа времени выполнения программы, реализующей численно-аналитический метод, было выяснено, что основное время занимает расчёт коэффициентов k по формулам (25)–(27). Для ускорения этой части программы было сделано несколько изменений в коде [67]:

1) изначально при вычислении коэффициентов по (26) выполнялся цикл от $j = 0$ до $j = i$. Такая реализация, при очередном j, таком, что $v_j > 1$, требовала пересчитывать все предыдущие k, кратные k_{ij}, кратность которых больше 1. Это приводило к многократному пересчёту одних и тех же значений и значительно увеличивало время выполнения программы. После оптимизации цикл стал выполняться от $j = i$ к $j = 0$. Таким образом, каждое значение k вычисляется только один раз;
2) было установлено, что большинство элементов матрицы A равны 0, поэтому в (23) было добавлено условие

$$b_{ij} = \sum_{w=j}^{i-1} a_{iw} k_{wj}, \; a_{iw} \neq 0, \quad (29)$$

которое позволило значительно сократить время работы алгоритма, за счёт отбрасывания операций, не влияющих на результат;
3) изначально в программе выделялась оперативная память для хранения всех значений b, но было замечено, что при вычислении значений k_{ij}, используются только значения b_{is}, с тем же первым индексом i. Было принято решение выделять память только для хранения набора коэффициентов b_{ij} при фиксированном i, а при переходе к вычислению значений при новом i освобождать использованную память. Такая оптимизация позволила сократить требуемый размер памяти для хранения коэффициентов b в $(N+1)(N+2)/2$ раз и, как следствие, увеличить скорость работы программы;
4) большая часть алгоритма может выполняться в многопоточном режиме, поэтому были определены точки, в которых потоки должны синхронизироваться, и реализована многопоточная структура алгоритма. Этими точками при вычислении k для любого i являются:
   а) окончание расчёта коэффициентов b_{ij} (29);

б) окончание расчёта коэффициентов k_{ij}, для которых $a_{ii} = a_{jj}$ и $v_j > 1$ (25) и для которых $a_{ii} \neq a_{jj}$ (11);

в) окончание расчёта оставшихся коэффициентов k_{ij} (26), (27), выполняющийся в один поток вычислений.

Кроме того была проведена оптимизация расчёта вероятностей состояний при вычисленных коэффициентах k:

1) из формулы (22) видно, что значения $x_j(t)$ для двух собственных чисел таких, что $a_{i_j} = a_{i_j+1, i_j+1}$, будут удовлетворять уравнению $x_{i_j+1}(t) = t x_{i_j}(t)$. Таким образом, при использовании этого уравнения в (22), удаётся значительно сократить время выполнения алгоритма;
2) было замечено, что при вычислении вероятностей состояний на промежутке времени от t_1 до t_2 с шагом h в (22) целесообразно сохранять значения экспоненты в отдельный вектор E размерности K. Действительно, если значения экспоненты для момента времени t_1 сохранены, тогда для вычисления вероятностей состояний в момент времени $t_1 + h$ можно воспользоваться выражением $x_j(t+h) = t^{v_j - 1} e^{a_{jj} h} E_j$. Данная модификация метода значительно уменьшает время работы алгоритма;
3) алгоритм вычисления вероятностей состояний по (22) так же был разделён на потоки.

Все перечисленные в данном пункте изменения программы значительно сократили время её выполнения.

5.5. Оценка точности

Одним из способов проверки точности работы метода является суммирование вероятностей всех состояний в какой-либо момент времени, так как суммарная вероятность всегда остаётся равной 1. В рассматриваемой модели НСО для всех заявок была установлена интенсивность поступления $\lambda = 1$, интенсивность обработки $\mu = 2$, число заявок взято $N = 100$, точность чисел — 1000 знаков. Рассчитывались вероятности состояний для $t = 50$. Сумма всех вероятностей состояний, рассчитанных численно-аналитическим методом составила $1 + 616856 \cdot 10^{-678}$. При этом, ни одна вероятность не была отрицательной. В Matlab сумма всех состояний была равна 1, но были состояния, вероятность которых меньше нуля, в частности состояние номер 5087 имело вероятность $-9,2719644052270 \cdot 10^{-9}$.

Для дополнительной проверки точности на Java был реализован метод метод Рунге—Кутты четвёртого порядка для решения систем дифференциальных уравнений с постоянным шагом [68]. В таблице 7 приведены значения вероятности поглощающего состояния с номером 5150 для $t = 50, 100, 150$. Интенсивность поступления заявок $\lambda = 1$, интенсивность обработки $\mu = 2$, $N = 100$. Шаг по времени в методе Рунге—Кутты обозначен h.

Таблица 6 – Сравнение точности

	$t=50$	$t=100$	$t=150$
Matlab	$4,24408126\cdot10^{-8}$	0,47343**46**	0,9999919**7951**
Аналитический метод	$2,24023344\cdot10^{-11}$	0,47343**80**	0,9999919**9164**
Метод Рунге—Кутты ($h=0,4$)	$3,64428442\cdot10^{-11}$	0,47343**53**	0,9999919**9068**
Метод Рунге—Кутты ($h=0,2$)	$2,23763751\cdot10^{-11}$	0,47343**78**	0,9999919**9158**

Из таблицы видно, что чем h меньше, тем ближе значение вероятности к результату, полученному аналитическим методом и наоборот, чем шаг больше, тем значение ближе к значению, полученному с использованием Matlab.

5.6. Сравнительная оценка скорости работы программной реализации метода

Был проведён эксперимент по сравнению скорости работы метода с Matlab. Интенсивность поступления заявок $\lambda=1$, интенсивность обработки заявок $\mu=2$, количество заявок $N=100$, размерность чисел в аналитическом методе равна 1000. В таблице 7 приведено время работы метода и Matlab при расчёте вероятностей состояний в различные моменты времени.

Особенность данного метода такова что, рассчитав значения коэффициентов k, можно получить значения вероятностей состояний в любой момент времени. Поэтому был проведён ещё один эксперимент с теми же исходными данными, в котором аналитический метод использовал для расчёта вероятностей заранее сохранённые значения коэффициентов k. Результаты эксперимента приведены в таблице 8.

Таблица 7 – Сравнение скорости с учётом расчёта коэффициентов k

	Нахождение решения в $t=50$, с	Нахождение решения в $t=100$, с	Нахождение решения в $t=150$, с	Нахождение решения в $t=200$, с
Matlab	08,370605	14,19324	20,578149	25,971104
Аналитический метод	55.961	59,98	58,637	61,616

Таблица 8 – Сравнение скорости при использовании уже найденных k

	Нахождение решения в $t=50$, с	Нахождение решения в $t=100$, с	Нахождение решения в $t=150$, с	Нахождение решения в $t=200$, с
Matlab	8,370605	14,19324	20,578149	25,971104
Аналитический метод	4,782	4,298	4,673	4,457

В третьем эксперименте проверялось влияние размерности чисел на скорость работы алгоритма с заранее сохранёнными значениями коэффициентов k. Интенсивность поступления заявок $\lambda=1$, интенсивность обработки заявок

$\mu = 2$, количество заявок $N = 100$, момент времени $T = 100$. При 500 знаках расчёт занимает 3,086 с, а при 1000 – 4,680 с.

Из таблиц 7 и 8 видно, что использование численно-аналитического метода позволяет получить расчет за одинаковое время, независимо от момента времени, для которого необходимо произвести расчёт. Кроме того, рассчитав и сохранив матрицу коэффициентов *k* для конкретной НСО, можно получать вероятности состояний для любого момента времени намного быстрее по сравнению с MatLab.

6. Алгоритмы формирования матрицы коэффициентов системы однородных дифференциальных уравнений Чепмена – Колмогорова

6.1. Рекурсивный алгоритм генерации матрицы коэффициентов системы однородных дифференциальных уравнений Чепмена – Колмогорова

Когда известно общее уравнение системы ОДУ, описывающей НСО, реализовать алгоритм генерации матрицы A не составляет труда. Но, при добавлении в модель НСО новых характеристик, таких как несколько каналов обслуживания, различный приоритет заявок, очереди к каналам и так далее, общее уравнение системы значительно усложняется. Для примера приведём общее уравнение системы ОДУ, описывающей двухканальную систему обслуживания, приведённую в [22]. Построить диаграмму состояний и переходов, а затем вывести такое уравнение и не допустить в нём ошибки очень сложно. При этом существуют куда более сложные системы, вывод общего уравнения для которых не представляется возможным. Например, сетевая модель, описанная в [70]. Для возможности расчёта вероятностно временных характеристик таких моделей, авторами работ [71–72], было принято решение разработать алгоритм генерации матрицы A без вывода общего уравнения системы ОДУ.

Суть его заключается в том, что необходимо вывести правила перехода НСО из одного состояния в другое и ввести некоторое начальное состояние. Под переходом из одного состояния в другое понимается поступление новой заявки на обслуживание, либо обработка заявки, которая в исходном состоянии обрабатывалась (возможен вариант окончания этапа аппроксимирующего распределения, при аппроксимации распределением, имеющим 2 и более этапов, в приведённых примерах он рассматривался). Начальным состоянием может быть состояние системы, в котором $n = 0$ и $m = 0$. Определяется номер начального состояния и его номер в списке состояний (Num). Для начального состояния $Num = 1$. Далее генерируется одно из возможных состояний по правилам перехода, проверяется было ли сгенерировано данное состояние до этого и если нет, то добавляется оно в список состояний (xT). Затем записывает изменения в матрицу A по следующей формуле:

$$A_{xT.len,Num} = x;$$
$$A_{Num,Num} = A_{Num,Num} - x, \quad (30)$$

где $xT.len$ – длина списка состояний, сгенерированных на данный момент; x – интенсивность перехода из исходного состояния в новое. Ею может быть интенсивность поступления новой заявки, либо интенсивность обработки очередной заявки. Если полученное состояние уже было сгенерировано и находится в списке состояний под номером $Num1$, то формула (30) принимает вид:

$$A_{Num1,Num} = x;$$
$$A_{Num,Num} = A_{Num,Num} - x. \quad (31)$$

Если сгенерированное состояние не было получено ранее, рекурсивно генерируются состояния, в которые НСО может перейти из только что полученного. При этом, в рекурсию передаётся $Num = xT.len$. Если для какого либо состояния сгенерированы все состояния, в которые может перейти НСО, то этот этап рекурсии завершается. Алгоритм прекращает работу, когда завершатся все этапы рекурсии. Данный алгоритм представлен на рис. 3.

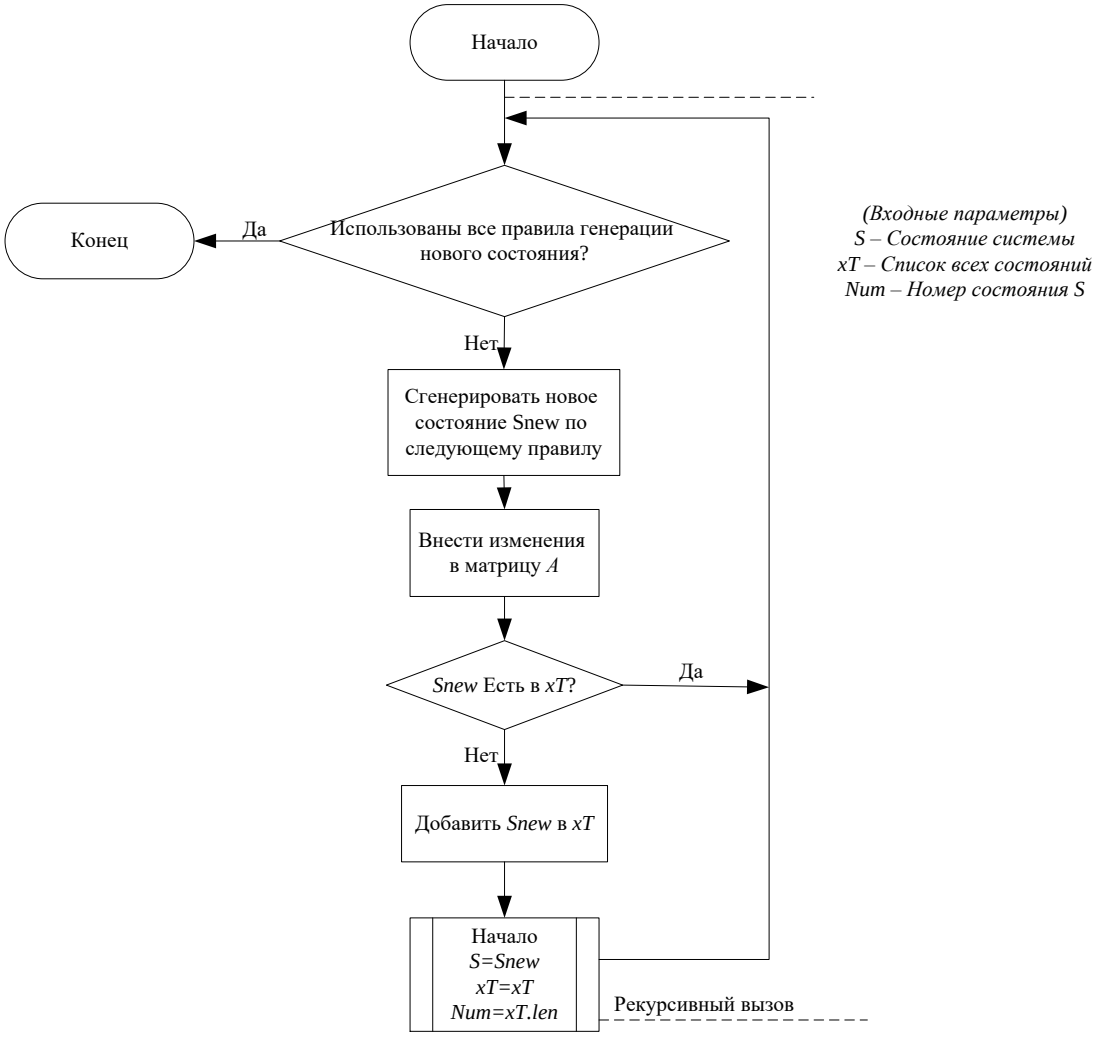


Рис. 3. Рекурсивный алгоритм формирования матрицы коэффициентов

В следующих параграфах приведены детальные алгоритмы генерации матрицы коэффициентов для некоторых многопараметрических моделей НСО.

6.2. Рекурсивный алгоритм формирования матрицы коэффициентов для двухканальной НСО

Детальная демонстрация рекурсивного алгоритма генерации матрицы A приводится на примере модели двухканальной системы, описанной в [21].
Для возможности определения номера заявки, которую канал обслуживает в данный момент, в модели НСО вектор $\vec{l}$ был преобразован в такой что, $\vec{l}_j = 0$ если j-й канал в данный момент свободен, $\vec{l}_j = i$, если j-й канал в данный момент обрабатывает i-ю заявку.

Так как состояния НСО характеризуются значениями $(n, m, \vec{l})$, то переходы между состояниями происходят по следующим правилам:

1) Из состояния $P_{(n,m,\vec{l})}$, где $n + m < K$, система переходит в состояние $P_{(n1,m1,\vec{l}1)}$, где $n1 = n, m1 = m$, с интенсивностью λ_{n+m+1}. При этом может выполняться одно из условий:

   а) если $\vec{l}_1 = 0$, то $\vec{l}1_1 = \begin{cases} n+1, \vec{l}1_2 = 0 \\ n+2, \vec{l}1_2 > 0 \end{cases}$, $\vec{l}1_2 = \vec{l}_2$;

   б) если $\vec{l}_2 = 0$ и $\vec{l}_1 \neq 0$, то $\vec{l}1_1 = \vec{l}_1$, $\vec{l}1_2 = \begin{cases} n+1, \vec{l}1_1 = 0 \\ n+2, \vec{l}1_1 > 0 \end{cases}$;

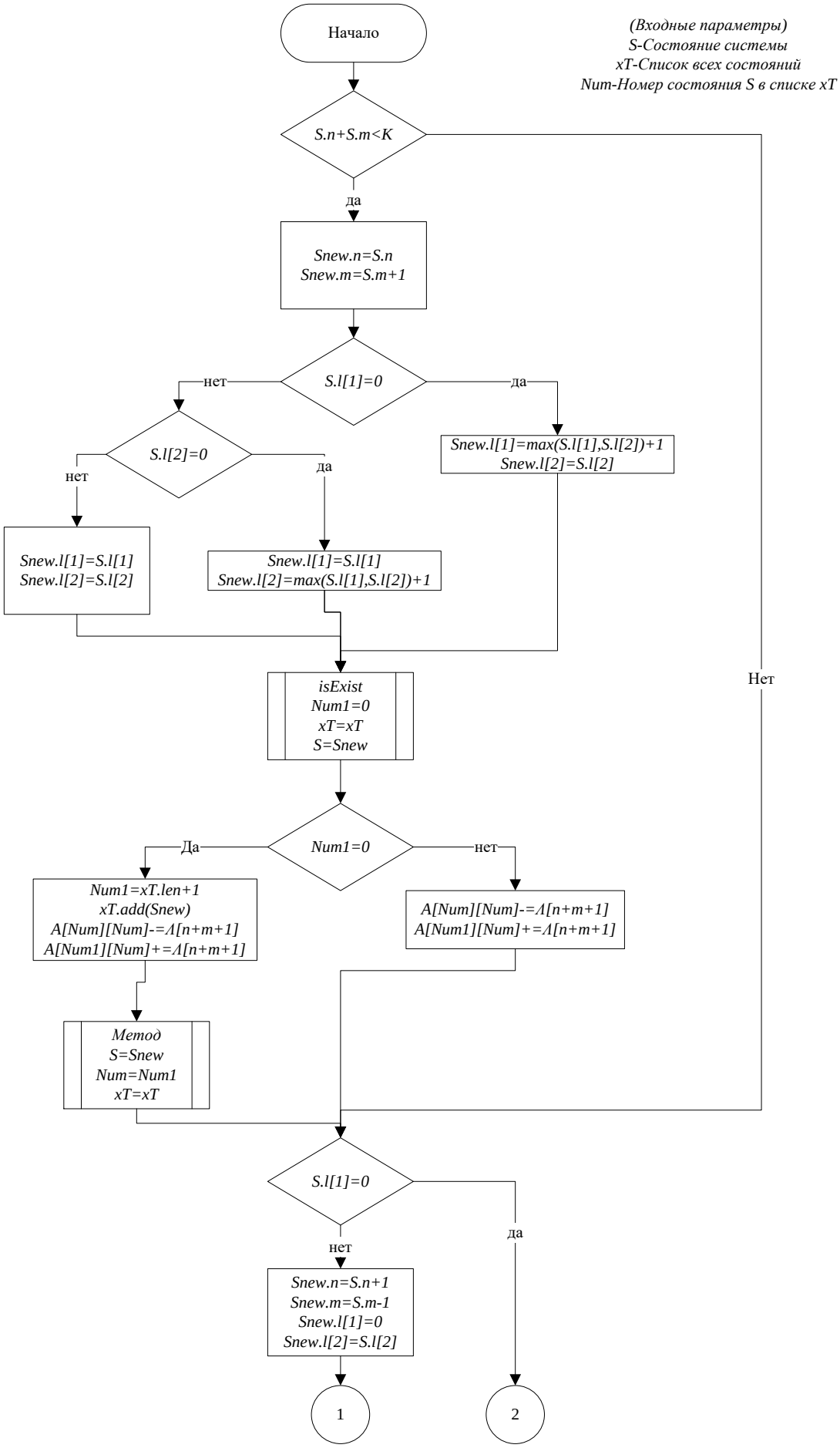
2) Из состояния $P_{(n,m,\vec{l})}$,где $\vec{l}_1 \neq 0$, система переходит в состояние $P_{(n1,m1,\vec{l}1)}$, где $n1 = n, m1 = m-1$, $\vec{l}_1 = 0$, $\vec{l}1_2 = \vec{l}_2$ с интенсивностью $\mu_{l_1,1}$;

3) Из состояния $P_{(n,m,\vec{l})}$,где $\vec{l}_2 \neq 0$, система может перейти в состояние $P_{(n1,m1,\vec{l}1)}$, где $n1 = n, m1 = m-1$, $\vec{l}1_1 = \vec{l}_1$ $\vec{l}_2 = 0$ с интенсивностью $\mu_{l_2,2}$.

В качестве начального состояния было выбрано состояние $P_{(n,m,\vec{l})}$, где $n = 0, m = 0$, $\vec{l} = [0,0]$.

Алгоритм для двухканальной НСО с интенсивностью поступления заявок и интенсивностью их обслуживания, зависящей от номера заявки и номера канала обслуживания, представлен на рис. 4. При этом в алгоритме используется дополнительная функция *isExist*, которая возвращает номер состояния *S* в списке состояний *xT* или 0, если состояния в списке нет. Её параметрами являются *S* – состояние, номер которого ищется, *xT* – список состояний и *Num1* – номер в списке состояний, который требуется найти. Алгоритм данной функции не приводиться, так как это стандартный алгоритм поиска объекта в списке.

Как видно из примера выше, предложенный алгоритм значительно упрощает процесс составления программы для генерации матрицы коэффициентов A для системы дифференциальных уравнений. Результаты работы модели были проверены другой моделью, где матрица коэффициентов A формировалась с помощью общего уравнения системы ОДУ [21].


Начало
(Входные параметры)
S-Состояние системы
xT-Список всех состояний
Num-Номер состояния S в списке xT
S.n+S.m<K
да
Нет
Snew.n=S.n
Snew.m=S.m+1
S.l[1]=0
нет
да
S.l[2]=0
нет
да
Snew.l[1]=max(S.l[1],S.l[2])+1
Snew.l[2]=S.l[2]
Snew.l[1]=S.l[1]
Snew.l[2]=S.l[2]
Snew.l[1]=S.l[1]
Snew.l[2]=max(S.l[1],S.l[2])+1
isExist
Num1=0
xT=xT
S=Snew
Num1=0
Да
нет
Num1=xT.len+1
xT.add(Snew)
A[Num][Num]-=Λ[n+m+1]
A[Num1][Num]+=Λ[n+m+1]
A[Num][Num]-=Λ[n+m+1]
A[Num1][Num]+=Λ[n+m+1]
Метод
S=Snew
Num=Num1
xT=xT
S.l[1]=0
нет
да
Snew.n=S.n+1
Snew.m=S.m-1
Snew.l[1]=0
Snew.l[2]=S.l[2]
1
2

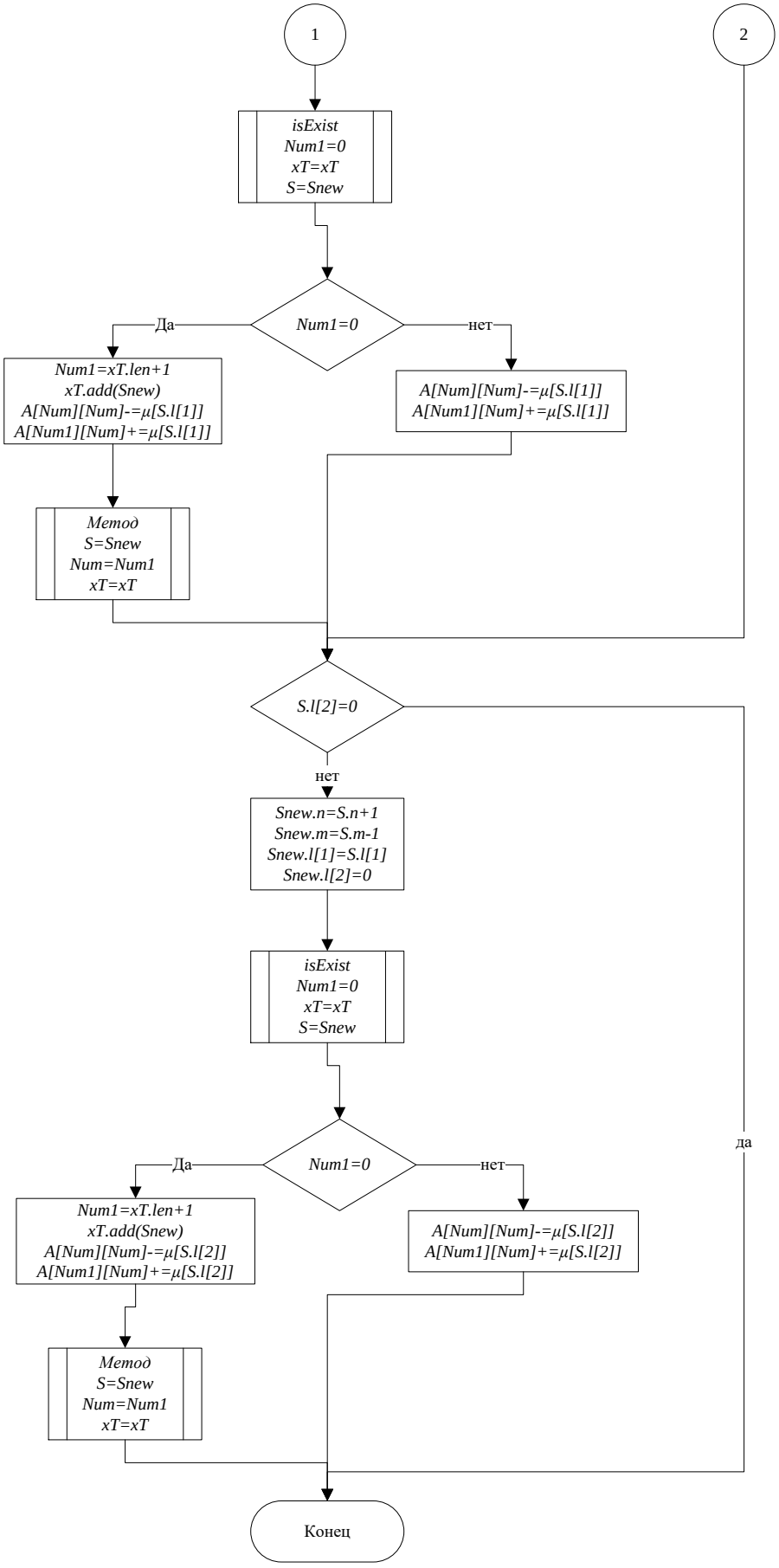


Рис. 4. Рекурсивный алгоритм для двухканальной модели НСО

6.3. Рекурсивный алгоритм формирования матрицы коэффициентов для сетевой модели НСО

В работе [70] представлена сетевая модель НСО, разработка и программная реализация которой, стала возможной с использованием предложенных методов и алгоритмов. В модели число заявок конечно. Заявки каждого типа имеют относительный приоритет. Каждый канал обслуживания имеет свою очередь. Длина очереди на обслуживание не ограничена. Заявки поступают из источника заявок в соответствии с матрицей доступности каналов (зависит от технологического графика управления) на обслуживание или в очередь к определенному каналу. После обслуживания заявки покидают сеть обслуживания. Графическая интерпретация представлена на рис. 5.

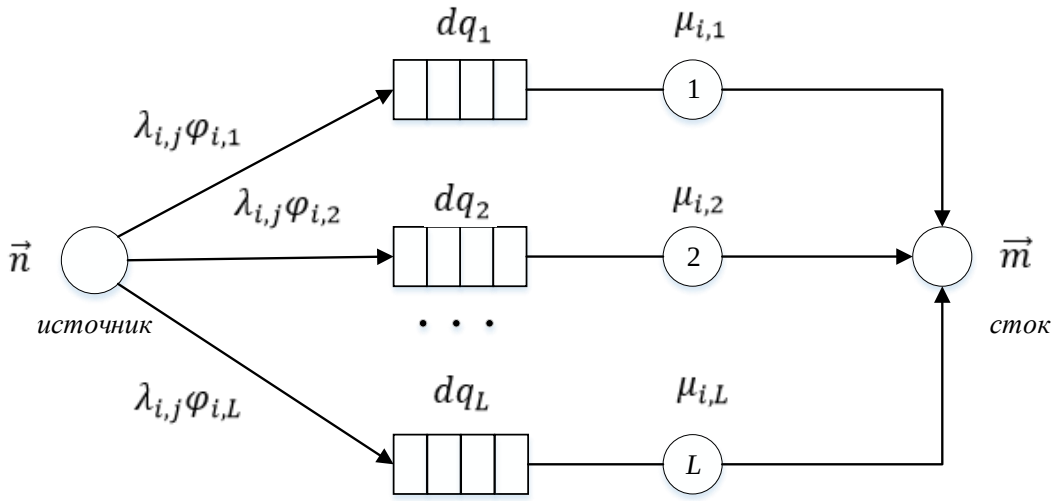


Рис. 5. Графическая интерпретация модели

Модель характеризуется следующими параметрами:

1) M – количество типов заявок;
2) $KT_i(i=\overline{1,M})$ – количество заявок i-го типа, поступающих в НСО за интервал моделирования;
3) λ_{ij} – элемент матрицы интенсивности поступления заявки i-го типа с номером j;
4) L – число каналов обслуживания;
5) $\mu_{ij}(i=\overline{1,M}, j=\overline{1,L})$ – элемент матрицы интенсивности обслуживания каналом с номером j заявки i-го типа;
6) $\varphi_{ij}(i=\overline{1,M}, j=\overline{1,L})$ – элемент матрицы доступности каналов, $\varphi_{ij}=1$ заявке типа i доступен канал с номером j, $\varphi_{ij}=0$ в противоположном случае;
7) $\vec{n}=\{n_i\}(i=\overline{1,M})$ – вектор, определяющий количество заявок i-го типа, находящихся в НСО;
8) $\vec{m}=\{m_i\}(i=\overline{1,M})$ – вектор, определяющий количество заявок каждого типа, уже получивших обслуживание и покинувших НСО;

9) $\vec{l} = \{l_j\}(j = \overline{1,L})$ – вектор состояний каналов обслуживания, $l_j = 0$, если j-й канал свободен, $l_j = i$, если j-й канал обслуживает заявку типа i;
10) dq_i – длина очереди к i-му каналу обслуживания, $i = \overline{1,L}$;
11) $\vec{q_i} = \{q_{ij}\}, j = \overline{1,dq_i}$ – вектор очереди к i-му каналу обслуживания, при этом j-я компонента i-го вектора равны номеру типа заявки, стоящей в очереди к i-му каналу обслуживания на j-ом месте;
12) $\overrightarrow{pr} = \{pr_i\}, i = \overline{1,M}$ – i-я компонента вектора определяет приоритет заявок i-го типа.

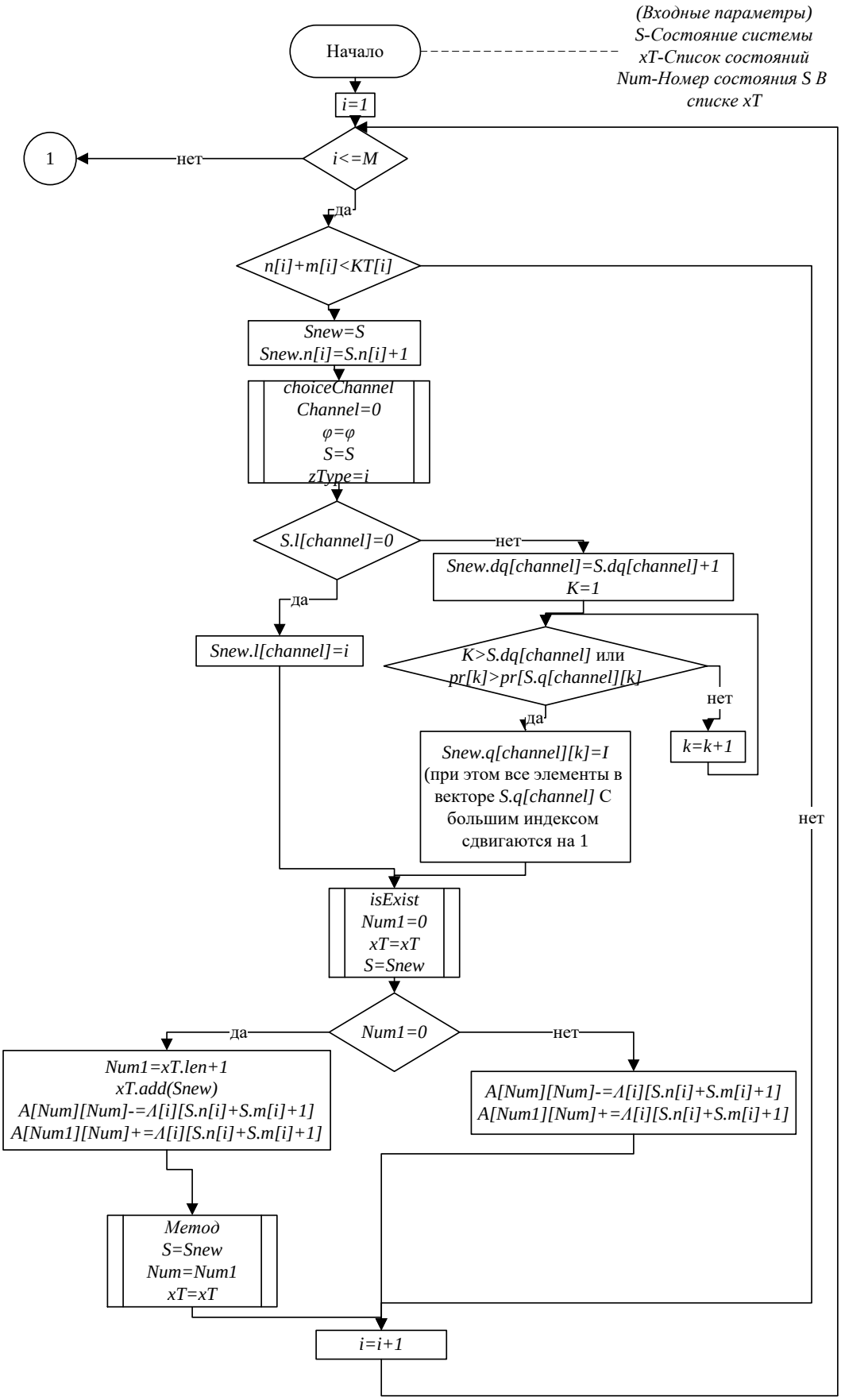
Состояние НСО в каждый момент времени характеризуются упорядоченным набором векторов: $(\vec{n}, \vec{m}, \vec{l}, \overrightarrow{dQ}, \vec{Q})$.

Алгоритм также использует дополнительную функцию *isExist*. Кроме того, для возможности гибкого распределения задач между узлами (каналами обслуживания) было введено понятие алгоритма выбора канала обслуживания. Алгоритм выбора канала обслуживания реализован в дополнительной функции *choiceChannel*. Эта функция имеет 4 параметра: φ – матрица доступности каналов; *S* – текущее состояние системы; *zType* – номер типа заявки; *channel* – переменная, в которую запишется результат работы функции.

Переходы между состояниями системы происходят по следующим правилам:

1) из состояния $P_{\vec{n},\vec{m},\vec{l},\overrightarrow{dQ},\vec{Q}}$ в котором $n_i + m_i < KT_i$ система может перейти в состояние $P_{\vec{n1},\vec{m1},\vec{l1},\overrightarrow{dQ1},\vec{Q1}}$ с интенсивностью λ_{i,n_i+m_i+1}. При этом номер канала (j), на который попадёт заявка, определяется с помощью функции *choiceChannel*, если $l_j > 0$, то $dq1_j = dq_j + 1$, а i добавляется в $\vec{q_j}$ перед всеми заявками, имеющими меньший приоритет. Если $l_j = 0$, то $l1_j = i$, вектор dq и матрица q остаются без изменения. А $n1_i = n_i + 1$. Вектор $\vec{m}$ остаётся без изменений;
2) из состояния $P_{\vec{n},\vec{m},\vec{l},\overrightarrow{dQ},\vec{Q}}$ в котором $l_i > 0$ система может перейти в состояние $P_{\vec{n1},\vec{m1},\vec{l1},\overrightarrow{dQ1},\vec{Q1}}$ с интенсивностью $M_{l_i i}$, где $n1_i = n_i - 1$, $m1_i = m_i + 1$, $\vec{l1}_i = \begin{cases} 0, dq_i = 0; \\ q_{i,1}, dq_i > 0 \end{cases}$, при этом необходимо удалить $q_{i,1}$.

Кроме того, в алгоритме используется функция проверки наличия состояния *isExist*, такая же, как и для двухканальной системы, описанной выше. В качестве начального состояния выбрано состояние, в котором 0 обслуженных заявок и 0 поступивших. Алгоритм генерации матрицы коэффициентов для сетевой НСО приведен на рис. 6.


Начало
(Входные параметры)
S-Состояние системы
xT-Список состояний
Num-Номер состояния S В списке xT
i=1
i<=M
нет
1
да
n[i]+m[i]<KT[i]
нет
Snew=S
Snew.n[i]=S.n[i]+1
choiceChannel
Channel=0
φ=φ
S=S
zType=i
S.l[channel]=0
нет
Snew.dq[channel]=S.dq[channel]+1
K=1
да
Snew.l[channel]=i
K>S.dq[channel] или pr[k]>pr[S.q[channel][k]
нет
да
k=k+1
Snew.q[channel][k]=I (при этом все элементы в векторе S.q[channel] С большим индексом сдвигаются на 1
нет
isExist
Num1=0
xT=xT
S=Snew
Num1=0
да
нет
Num1=xT.len+1
xT.add(Snew)
A[Num][Num]-=Λ[i][S.n[i]+S.m[i]+1]
A[Num1][Num]+=Λ[i][S.n[i]+S.m[i]+1]
A[Num][Num]-=Λ[i][S.n[i]+S.m[i]+1]
A[Num1][Num]+=Λ[i][S.n[i]+S.m[i]+1]
Метод
S=Snew
Num=Num1
xT=xT
i=i+1

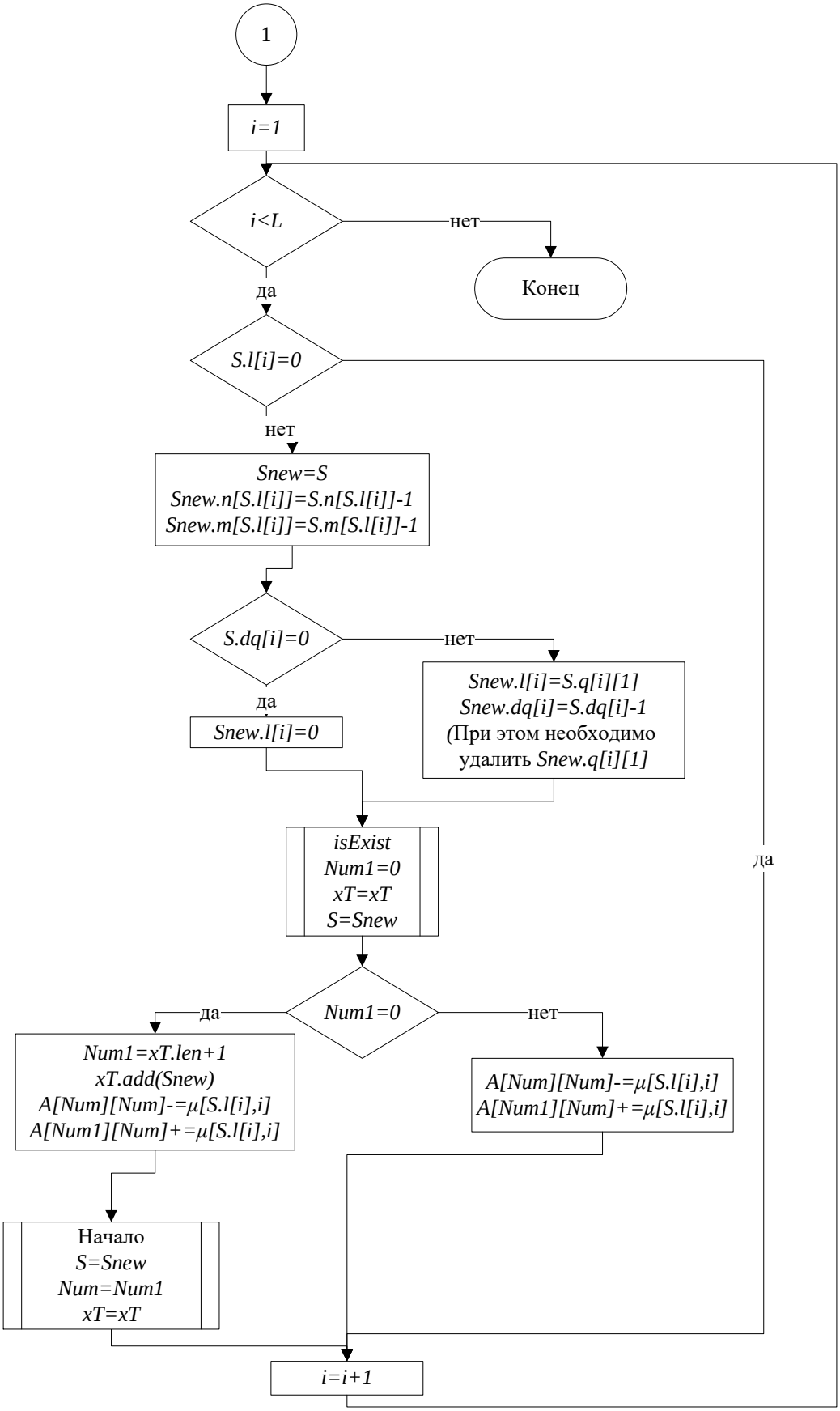


Рис. 6. Рекурсивный алгоритм формирования матрицы коэффициентов сетевой модели НСО

Данный алгоритм был интегрирован в программную реализацию численно-аналитического метода, описанного выше. Кроме того, была реализованная имитационная сетевая модель НСО [71]. В таблице 9 приведены вероятности поглощающего состояния системы в различные моменты времени, полученные двумя моделями.

Таблица 9 – Сравнение результатов моделирования

Момент времени	Имитационная модель	Аналитическая
1	0,010411	0,0105599357
2	0,224826	0,225337103263
3	0,564251	0,56480951
4	0,776363	0,776569477849
5	0,869929	0,87000275886

6.4. Сортировка элементов матрицы

Как было указано в [64], для системы (19) существует явное аналитическое решение, если только известны собственные числа матрицы A. Очевидным является тот факт, что в случае треугольного вида матрицы A (для определённости будем считать её нижнетреугольной), её собственные числа выписаны в явном виде на диагонали. Таким образом, аналитическое решение системы можно найти, если только матрица A — треугольная.

Если пронумеровать состояния по возрастанию числа обработанных запросов, а внутри этих групп по возрастанию числа поступивших запросов то ни одно из состояний полученного списка не будет иметь зависимости от последующих, а матрица A примет треугольный вид. Например, в сетевой модели в одной подгруппе одной группы будет более одного состояния, но нумерация останется верной, так как эти состояния не будут зависеть друг от друга.

На выходе предлагаемый алгоритм генерации матрицы выдаёт результаты в виде массива *xT*, в котором хранятся все состояния системы. Необходимо отсортировать элементы массива так, как это описано выше. Для этого можно воспользоваться любым алгоритмом сортировки. При этом, при перестановке двух состояний с номерами *x1* и *x2* необходимо поменять местами 2 строки и 2 столбца в матрице A с теми же номерами. Пример такой перестановки приведён в таблицах 10 и 11.

В таблице 10 приведён фрагмент матрицы состояний A. Первый столбец и первая строка содержат условные номера состояний. В ячейках таблицы содержится информация вида: *x-y*, обозначающая интенсивность перехода из состояния x в состояние y. Допустим в процессе сортировки появилась необходимость поменять местами состояния с номерами *x1* и *x2*. Матрица коэффициентов A после такой перестановки представлена в таблице 11. Как видно из таблицы 11 зависимости между состояниями не нарушены, что доказывает адекватность предлагаемого метода сортировки.

Таблица 10 – Фрагмент матрицы до перестановки

	1	2	...	x1	x2	x3
1	1–1	2–1	...	x1–1	x2–1	x3–1
2	1–2	2–2	...	x1–2	x2–2	x3–2
...	...	...	...	...	...	...
x1	1-x1	2-x1	...	x1-x1	x2-x1	x3-x1
x2	1-x2	2-x2	...	x1-x2	x2-x2	x3-x2
x3	1-x3	2-x3	...	x1-x3	x2-x3	x3-x3

Таблица 11 – Фрагмент матрицы после перестановки

	1	2	...	x2	x1	x3
1	1–1	2–1	...	x2–1	x1–1	x3–1
2	1–2	2–2	...	x2–2	x1–2	x3–2
...	...	...	...	...	...	...
x2	1-x2	2-x2	...	x2-x2	x1-x2	x3-x2
x1	1-x1	2-x1	...	x2-x1	x1-x1	x3-x1
x3	1-x3	2-x3	...	x2-x3	x1-x3	x3-x3

Какой бы сложной не была модель НСО, предложенный подход позволяет синтезировать для неё алгоритм генерации матрицы коэффициентов. Увеличение числа параметров модели НСО ведёт к увеличению числа состояний и, иногда, правил переходов, но сложность вывода этих правил практически не меняется. С использованием рекурсивного алгоритма генерации матрицы коэффициентов ОДУ Чепмена – Колмогорова был разработан ряд новых моделей НСО представленных в [73].

6.5. Последовательный алгоритм генерации матрицы коэффициентов

Представленный выше рекурсивный алгоритм, по мнению авторов статьи [74], также имеет недостатки, а именно: на выходе алгоритм предоставляет неотсортированный список состояний, состояния в нем находятся в порядке их рекурсивного генерирования; на выходе алгоритм предоставляет массив нужной размерности вместо нижней треугольной матрицы; исходя из первых двух пунктов, вытекает необходимость сортировки списка состояния и матрицы коэффициентов для приведения их в нужное состояние, при замене местами значений в списке состояний, необходимо поменять местами строки и столбцы для тех же номеров состояний в матрице, а при больших размерах матрицы это является трудоемкой операцией. Для устранения указанных недостатков авторы предлагают последовательный алгоритм генерации матрицы коэффициентов, использующий совершенно иной подход.

Суть последовательного алгоритма заключается в разделении списка состояний на подгруппы с одинаковыми свойствами и дальнейшем использовании этих свойств. Для примера рассматривается базовая модель НСО представленная в первом разделе настоящей статьи. Все состояния делятся на группы, такие, что в каждой группе при постоянном j, значение i будет расти до $N - j$.

Отмечается несколько важных фактов: количество групп всегда будет равно $N+1$; длина каждой последующей группы всегда будет на 1 меньше текущей; в рамках одной группы состояния могут переходить друг в друга последовательно и только с интенсивностью λ_i; (4) переход с интенсивностью μ_j может происходить только из состояния *i* группы *j* в состояние следующей группы *j+1*, при этом номер нового состояния внутри группы всегда будет *i-1*, интенсивность такого перехода всегда будет μ_j.

Следующим шагом генерируется список состояний длиной $N_c = (N+1)(N+2)/2$. Список генерируется простым перебором *j* во внешнем цикле, *i* во внутреннем, до тех пор, пока $i + j \le N$. При достижении этого порога происходит увеличение *j* на единицу. На выходе алгоритм предоставляет упорядоченный список состояний и делает это в один проход. Важным компонентом является структура данных для хранения этого списка. Состояния хранятся в списке со списками. Основной список содержит в себе несколько групп, каждая из которых является отдельным списком. Внутри группы представлены состояния, каждое из которых хранится как список, содержащий номер состояния и список с его описанием. Такая структура хранения упрощает дальнейшее заполнение матрицы коэффициентов и позволяет не перебирать список состояний при поиске нужного состояния, а сразу обращаться в нужное место в списке. Например:

```
Все состояния [
        Группы [
          Состояние [
                Номер состояния,
                [заявок в системе,
                 обработанных заявок]
          ]
        ]
].
```

Следующим шагом заполняется матрица коэффициентов. При начальных условиях это квадратная нулевая матрица размерности *Nc*. Последовательно перебирается список состояний и для каждого состояния последовательно применяются 3 краевых условия, благодаря которым применяются все необходимые изменения матрицы коэффициентов, описывающих это состояние. Обозначим матрицу коэффициентов как *A*, номер текущего состояния как *Num*, μ и λ – интенсивности поступления и обслуживания заявки с номером *Num*.

1) если это не первое состояние внутри своей группы:
$$A_{\text{Num,Num}} = A_{\text{Num,Num}}\mu;$$
$$A_{\text{Num,Num}-1} = A_{\text{Num,Num}-1} + \lambda;$$
2) если это не последнее состояние внутри своей группы:
$$A_{\text{Num,Num}} = A_{\text{Num,Num}} - \lambda;$$
3) если это не первая группа:
$$A_{\text{Num,Prev\_Num}} = A_{\text{Num,Prev\_Num}} + \mu,$$

где *Prev_num* – это номер состояния из которого можно попасть в состояние с номером *Num* обработав заявку, в списке состояний всегда находится в группе *j*-1 и имеет номер в своей группе *i*+1.

После обработки всех состояний такой алгоритм на выходе предоставляет готовую нижнетреугольную матрицу коэффициентов, что избавляет от необходимости её сортировки.

Последовательный алгоритм формирования матрицы коэффициентов представлен на рис. 7.

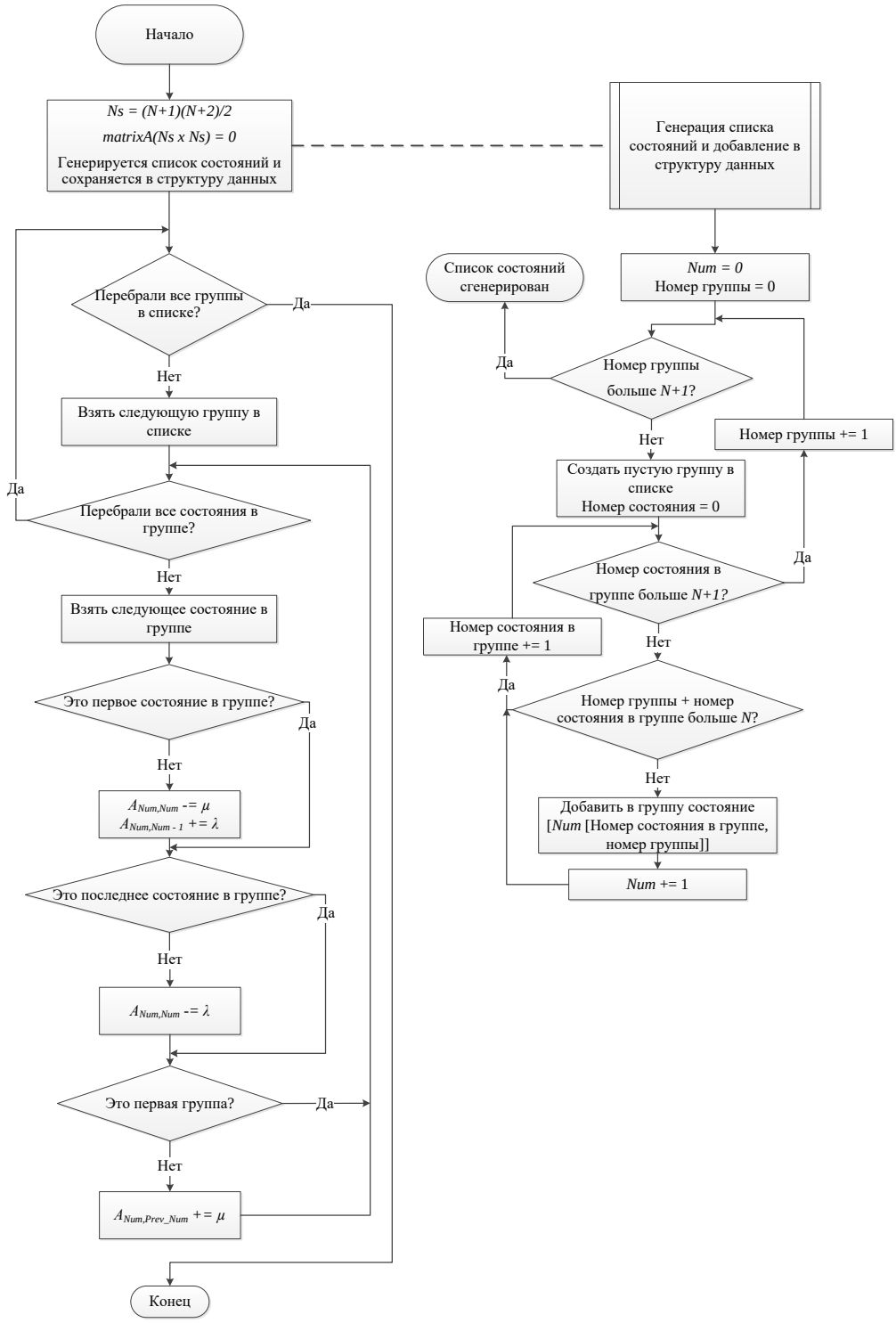


Рис. 7. Последовательный алгоритм формирования матрицы коэффициентов

№3. 2020
ISSN 2410–9916

Оба алгоритма генерации матрицы коэффициентов системы ОДУ без вывода общего уравнения системы были реализованы на языке Python3. Результаты сравнительного анализа представлены в таблице 12.

Таблица 12 – Сравнение времени выполнения последовательного и рекурсивного алгоритмов

Заявок	Состояний	Время последовательного, с	Время рекурсивного, с	Генерирование, без сортировки (для рекурсивного), с
3	10	0,0027289390564	0,000235080718994	0,000102996826172
5	21	0,000874042510986	0,000907897949219	0,000251054763794
10	66	0,000315189361572	0,00605607032776	0,000638008117676
20	231	0,00100684165955	0,129016876221	0,00578188896179
30	496	0,00470900535583	0,592235088348	0,0215079784393
40	861	0,00546598434448	2,48418688774	0,0802609920502
50	1326	0,0105609893799	12,4112920761	0,151191949844
60	1891	0,0128128528595	83,0276899338	0,318285942078
70	2556	0,0225200653076	224,342078924	0,576465129852
80	3321	0,0305080413818	568,904677153	0,847626924515
90	4186	0,0393660068512	1163,46502709	1,33484387398
100	5151	0,105370044708	2274,09017706	1,87042212486

Как видно из таблицы 12, количество возможных состояний системы, а, следовательно, и количество уравнений в системе ОДУ растет экспоненциально от количества заявок, которые могут поступить в НСО. График зависимости количества состояний от количества заявок представлен на рис. 8.

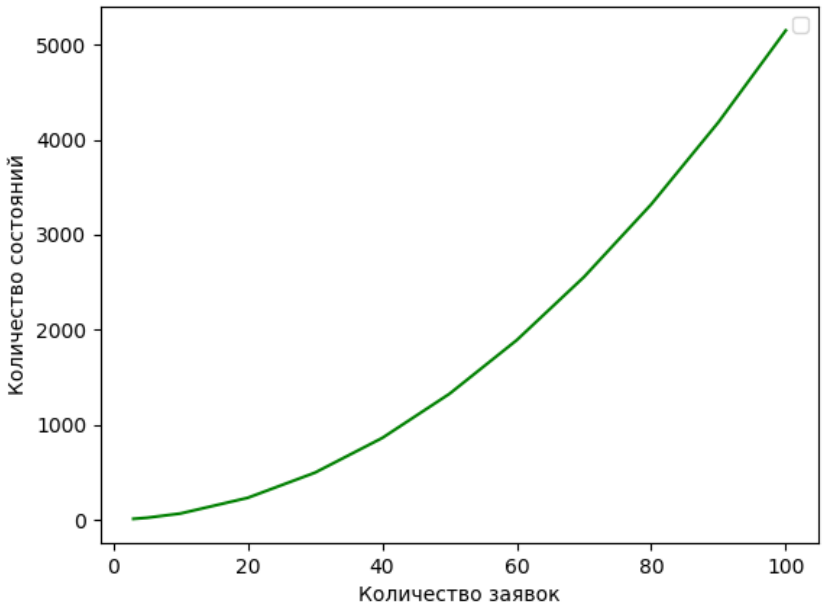


Рис. 8. Зависимость количества состояний от количества заявок

DOI: 10.24411/2410–9916–2020–10303
URL: https://sccs.intelgr.com/archive/2020–03/03-Bubnov.pdf

№3. 2020
ISSN 2410–9916

На рис. 9 представлено сравнение двух частей работы рекурсивного алгоритма: генерации списка состояний и матрицы коэффициентов и их последующая сортировка. Как видно из рис. 9, при увеличении количества состояний, большую часть времени рекурсивный алгоритм занят сортировкой полученных результатов.

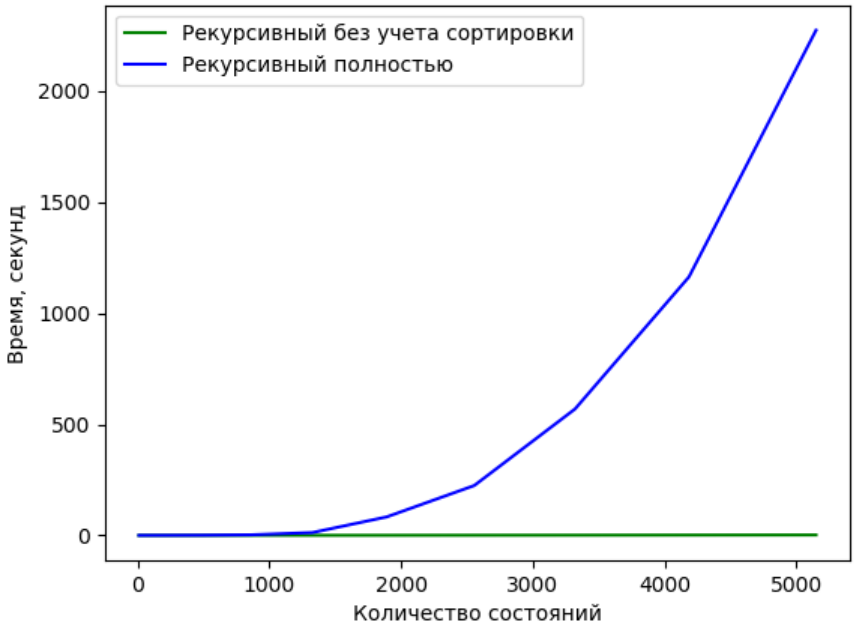


Рис. 9. Время выполнения обеих частей рекурсивного алгоритма

На рис. 10 представлено сравнение полного времени работы рекурсивного и последовательного алгоритмов генерации матрицы коэффициентов. Время выполнения рекурсивного алгоритма растет экспоненциально при увеличении количество состояний, исходя из рис. 10 можно заключить, что большая часть этого времени тратится на сортировку.

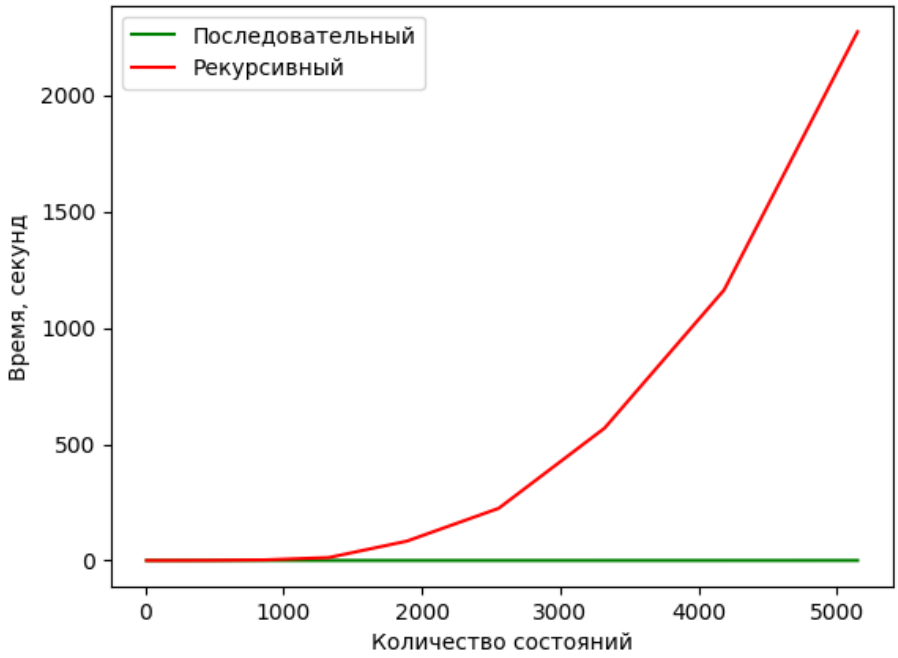


Рис. 10. Время выполнения последовательного и рекурсивного алгоритмов

DOI: 10.24411/2410–9916–2020–10303
URL: https://sccs.intelgr.com/archive/2020–03/03-Bubnov.pdf

№3. 2020
ISSN 2410–9916

На рис. 11 представлено время выполнения последовательного алгоритма и время выполнения рекурсивного алгоритма, исключая трудоемкий этап сортировки. Как можно заметить, даже в этом случае рекурсивный алгоритм уступает последовательному в скорости работы.

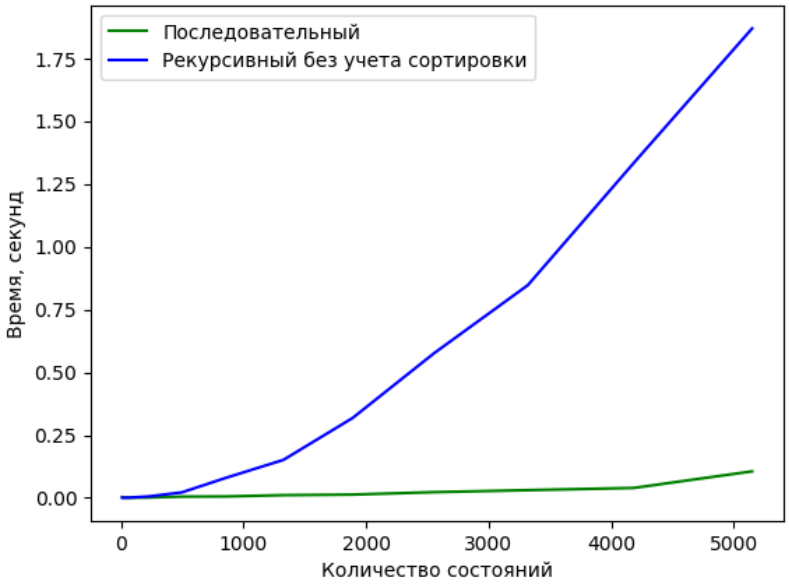


Рис. 11. Время выполнения последовательного алгоритма
и рекурсивного алгоритма без сортировки

На рис. 12 представлен график зависимости времени работы последовательного алгоритма от количества возможных состояний системы. Сняты 100 значений при количестве поступающих заявок от 1 до 100. Как можно заметить, время выполнения этого алгоритма имеет обыкновенную линейную зависимость от количества поступающих в систему заявок и количества состояний этой системы.

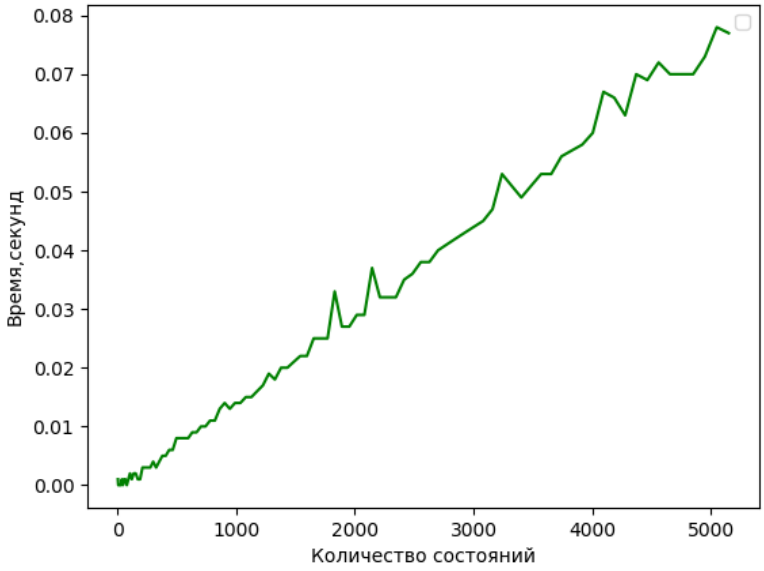


Рис. 12. Зависимость времени работы последовательного алгоритма
от количества возможных состояний системы

DOI: 10.24411/2410–9916–2020–10303
URL: https://sccs.intelgr.com/archive/2020–03/03-Bubnov.pdf

7. Новый подход к построению моделей НСО и расчету их вероятностей состояний

Для построения моделей НСО, учитывающих сетевые структуры, многоканальность, доступность каналов обслуживания для разного типа заявок, приоритетность, отличие законов распределения вероятностей временных интервалов поступления и обслуживания заявок от экспоненциального и так далее, подход, представленный в начале настоящей статьи будет затруднителен, а в некоторых случаях просто невозможен. Так как матрица коэффициентов системы ОДУ Чепмена – Колмогорова однозначно определяет необходимую модель НСО (диаграмму состояний и переходов, уравнения вероятностей каждого состояния, общее уравнение системы), то основной задачей исследователя становится разработка алгоритма формирования всех возможных состояний НСО и правил перехода между ними в соответствии с материалом настоящей статьи. А для расчета вероятностно-временных показателей модели НСО целесообразно использовать программный комплекс, основанный на численно-аналитическом методе.

Заключение

Целью настоящего исследования было демонстрация развития процесса разработки моделей НСО от подхода по построению диаграммы переходов и состояний для конкретной модели, записи дифференциальных уравнений Чепмена – Колмогорова для каждого состояния, вывода общего уравнения ОДУ и численного решения соответствующей задачи Коши до подхода по формированию матрицы коэффициентов A ОДУ и решения с помощью численно-аналитического метода. В настоящей статье также показаны возможные способы применения моделей НСО для различных прикладных задач по оценки пропускной способности аппаратно-программных комплексов и их компонентов, а также оценки показателей надежности программных средств. Авторы статьи подробно рассмотрели особенности программной реализации предложенных моделей НСО и методов расчета их характеристик. Показаны достоинства и недостатки каждой конкретной программной реализации, что позволит исследователям выбирать наиболее удобную для решения своих задач. Авторы считают, что предложенный инструментарий позволит как разработать новые модели НСО, так и применять предлагаемые для исследования широкого круга инженерных задач.

Исследования, выполненные по данной тематике, проводились в рамках бюджетной темы № 0073–2019–0004.

Литература

1. Хинчин А. Я Работы по математической теории массового обслуживания. – М.: ФИЗМАТЛИТ, 1963. – 236 с.

2. Takacs K. Investigation of Waiting Time Problems by Reduction to Orkov processes // Acta Mathematica Academiae Scientarium Hungaicae, 1955. Vol. 6. P. 101–129.

3. Гнеденко Б. В., Коваленко И. Н. Введение в теорию массового обслуживания. – М.: Наука, 1966. – 432 с.

4. Коваленко И. Н. О системе массового обслуживания со скоростью обслуживания, зависящей от числа требований в системе, и периодическим отключением каналов // Проблемы передачи информации. 1971. № 2. С. 108–114.

5. Коваленко И. Н. О восстановлении характеристик системы по наблюдениям над выходящим потоком // Доклады Академии Наук СССР. 1965. № 5. С. 979–981.

6. Ежов И. И., Корнейчук М. Т., Олийнык И. Д. Распределение количества каналов системы ремонта, когда интенсивность потока изменяется специальным образом // Кибернетика. 1976. № 3. С. 92–97.

7. Абольников Л. М. Нестационарная задача массового обслуживания для систем с бесконечным числом каналов при групповом поступлении требований // Проблемы передачи информации. 1968. № 3. С. 99–102.

8. Арсенишвили Г. Л. Однолинейная система массового обслуживания с зависящей от величины очереди интенсивностью входящего потока // Сообщения Академии наук Грузинской ССР. 1974. № 2. С. 285–288.

9. Conolly B. W. Generalized State Dependent Eriangian Queues (speculation about calculating easure of effectiveness) // Applied Probability. 1975. № 2. P. 358–363.

10. Hadidi N. A. A queueing model with variable arrival rates // Periodica Mathematica Hungarica. 1975. № 1. P. 39–47.

11. Сиголов Г. Г., Люперсольский А. М. Метод расчета переходных процессов в сетевых моделях массового обслуживания // Сборник тезисов всесоюзной школы-семинара по распределенным автоматизированным системам массового обслуживания (Рига, 15–20 октября 1988 г.). – Рига, 1988. – С. 353–354.

12. Сиголов Г. Г., Люперсольский А. М. Метод приближенного расчета переходных процессов в сетевых моделях массового обслуживания // Автоматика и вычислительная техника. 1990. № 3. С. 40–43.

13. Скляревич Ф. А. Характеристика пребывания фрагмента вычислительной сети в фиксированном подмножестве состояний // Автоматика и вычислительная техника. 1985. № 2. С. 14–20.

14. Головко Н. И., Каретник В. О., Пелешок А. В. СМО с бесконечным накопителем и скачкообразной интенсивностью входного потока // Автоматика и телемеханика. 2009. № 10. С. 75–96.

15. Зейфман А. И. О нестационарной модели Эрланга // Автоматика и телемеханика. 2009. № 12. С.71–80.

16. Gelenbe E., Mitrani I. Analysis and Synthesis of computer systems. – Academic Press, 1980. – 239p.

17. Zeifman A., Korotysheva A., Satin Y. On stability for Mt/Mt/N/N queue // Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2010 International Congress (Moscow, 18–20 October 2010). – Moscow, 2010. – P. 1102–1105.

18. Бубнов В. П., Сафонов В. И., Смагин В. А. О загрузке вычислительной системы с изменяющейся интенсивностью поступления заданий // Автоматика и вычислительная техника. 1987. № 6. С.19–22.

19. Бубнов В. П., Сафонов В. И. Алгоритм исследования модели нестационарных систем обслуживания // Сборник типовых алгоритмов и программ. 1987. № 8. С. 172–178.

20. Бубнов В. П., Сафонов В. И. Разработка динамических моделей нестационарных систем обслуживания. – СПб.: Издательство “Лань”, 1999. – 64 с.

21. Бубнов В. П., Михайлов А. В., Сафонов В. И. Устройство для моделирования процесса обслуживания заявок // Авторское свидетельство SU 1341648 A1, 30.09.1987. Заявка № 4063068 от 29.04.1986.

22. Пучков В. В., Смагин В. А., Бубнов В. П., Сафонов В. И. Устройство для моделирования систем массового обслуживания // Авторское свидетельство SU 1399756 A1, 30.05.1988. Заявка № 4155331 от 02.12.1986.

23. Бубнов В. П., Михайлов А. В., Сафонов В. И., Хапалов И. Л. Устройство для моделирования систем массового обслуживания // Авторское свидетельство SU 1405071 A1, 23.06.1988. Заявка № 4155357 от 02.12.1986.

24. Сафонов В. И., Бубнов В. П., Ткачев С. Б., Белугин Г. П. Устройство для моделирования систем массового обслуживания // Авторское свидетельство SU 1652979 A1, 30.05.1991. Заявка № 4708900 от 24.04.1989.

25. Бубнов В. П., Сергеев А. С. Аппроксимационный метод исследования немарковских систем: учебное пособие. – СПб.: Петербургский государственный университет путей сообщения императора Александра I, 2014. – 37 с.

26. Cox D.R. A use of complex probabilities in the theory of stochastic processes // Proceedings of the Cambridge Philosophical. 1955. № 2. P. 313–319.

27. Бубнов В. П., Бурцева К. И. Нестационарная модель для оценки функциональной безопасности средств железнодорожной автоматики // Проблемы математической и естественно научной подготовки в инженерном образовании. Исторический опыт, современные вызовы: сб. тр. Международной научно-методической конференции (11–12 ноября 2010 г.). – СПб., 2011. – С. 25- 33.

28. Тырва А. В. Модели надежности и планирования испытаний программных средств: дис. … к.т.н.: 05.13.18. – СПб., 2010. – 147 с.

29. Darcy D. P., Kemerer C. F. OO Metrics in Practice // Software. 2005. № 6. P. 17–19.

30. Dick S., Bethel C. L., Kandel A. Software-Reliability Modeling: The Case for Deterministic Behavior // IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans. 2007. Vol. 37. № 1. P. 106–119.

31. El-Emam K. Object-Oriented Metrics: A Review of Theory and Practice // Advances in software engineering. 2002. № 1. P. 23–50.

32. El-Emam K., Melo W., Machado J. C. The prediction of faulty classes using object-oriented design metrics // Journal of Systems and Software. 2001. № 1. P. 63–75.

33. Musa J. D. The Measurement and Management of Software Reliability // Proceedings of the IEEE. 1980. № 9. P. 1131–1143.

34. Тырва А. В., Хомоненко А. Д. Метод планирования тестирования сложных программных комплексов на этапах проектирования и разработки // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. 2009. № 4. С. 125–131.

35. Тырва А. В., Хомоненко А. Д. Планирование проведения тестирования комплексов программ на основе проектных метрик сложности // Труды Всероссийской научно-практической конференции «Транспорт-2009» (Ростов-на-Дону, 31 марта – 1 апреля, 2009 г.). – Ростов-на-Дону, 2009. – С. 80–82.

36. Jelinski Z., Moranda P. B. Software Reliability Research // Proceedings of the Statistical Methods for the Evaluation of Computer System Performance. – Academic Press, 1972. – P. 465–484.

37. Huang C. Y., Huang W. C. Software Reliability Analysis and Measurement Using Finite and Infinite Server Queuing Models // IEEE Transactions On Reliability. 2008. №1. P. 192–203.

38. Genero M., Piattini M., Caleron C. A survey of metrics for UML class diagrams // Journal of Object Technology. 2005. Vol. 4. № 9. P. 59–92.

39. Guo P., Lyu M. R. Software Quality Prediction Using Mixture Models with EM Algorithm // Proceedings of the First Asia-Pacific Conference on Quality Software. 2000. P. 69–78.

40. Bansiya J., Davis C. A Hierarchical Model for Object-Oriented Design Quality Assessment // IEEE Transactions on Software Engineering. 2002. Vol. 28. P. 4–17.

41. Basili V. R., Briand L. C. A validation of object-oriented design metrics as quality indicators // IEEE Transactions on Software Engineering. 1996. Vol. 22. P. 751–761.

42. Briand L., Devanbu P., Melo W. An Investigation into Coupling Measures for C++ // Proceedings of the 1997 (19th) International Conference on Software Engineering. 1997. P. 412–421.

43. Chidamber S. R., Kemerer C. F. A metrics suite for object oriented design // IEEE Transactions on Software Engineering. 1994. Vol. 20. P. 476–493.

44. Тырва А. В., Хомоненко А. Д. Метод планирования тестирования сложных программных комплексов на этапах проектирования и разработки // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. 2009. № 4. С. 125–131.

45. Тырва А. В. Методика задания исходных данных для моделей надежности программных средств железнодорожного транспорта // Известия Петербургского университета путей сообщения. 2010. № 2. С. 250–261.

46. Тырва А. В. Методика сертификационных испытаний программного обеспечения системы «Горочная автоматическая централизация микропроцессорная с ведением накопления вагонов в сортировочном парке (ГАЦ МН) 86246294.50 5200 020–01. – СПб.: Петербургский государственный университет путей сообщения императора Александра I, 2009. – 18 с.

47. Bubnov V. P., Tyrva A. V., Khomonenko A. D. Model of reliability of the software with Coxian distribution of length of intervals between the moments of detection of errors // Proceedings of 34th Annual IEEE Computer Software and Applications Conference (COMPSAC 2010). – Seoul, 2010. – P. 238–243.

48. Bubnov V. P., Tyrva A. V., Khomonenko A. D. Software Reliability Model with Coxian Distribution of Length of Intervals Between Errors Detection and Fixing Moments // Proceedings of 35th Annual IEEE Computer Software and Applications Conference (COMPSAC 2011). – Munich, 2011. – 2011. – P. 310–314.

49. Тырва А. В., Хомоненко А. Д., Бубнов В. П. Модели надежности программного обеспечения: учебное пособие. – СПб.: ПГУПС, 2010. – 39 с.

50. Бубнов В. П., Тырва А. В., Бурцева К. И. Нестационарная модель надежности программных средств с распределением Кокса длин интервалов времени исправления ошибок // Вестник Всероссийского научно-исследовательского и проектно-конструкторского института электровозостроения. 2010. № 1. С. 143–152.

51. Бубнов В. П. Комплекс моделей надежности программных средств с распределениями фазового типа // Вестник Всероссийского научно-исследовательского и проектно-конструкторского института электровозостроения. 2011. № 1. С. 185–193.

52. Бубнов В. П., Тырва А. В., Еремин А. С. Комплекс моделей нестационарных систем обслуживания с распределениями фазового типа // Труды СПИИРАН. 2014. № 6. С. 61–71.

53. Бубнов В. П., Тырва А. В., Хомоненко А. Д. Обоснование стратегии отладки программ на основе нестационарной модели надежности // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. 2010. № 2. С. 85–92.

54. Тырва А. В., Хомоненко А. Д., Бубнов В. П. Комплекс программ расчета надежности и планирования испытаний программных средств // Свидетельство о государственной регистрации программ для ЭВМ № 2010615617.

55. Дашонок В. Л., Тырва А. В. Методика сертификационных испытаний программного обеспечения «Тестирующий комплекс систем микропроцессорных централизаций и систем микропроцессорных автоблокировок с централизованным размещением оборудования (ТК МПЦ/АБТМПЦ). Программное обеспечение типовое» 01095505.42 5210–01. – СПб.: Петербургский государственный университет путей сообщения императора Александра I, 2010. – 32 с.

56. Тырва А. В. Методика сертификационных испытаний программного обеспечения комплексной системы автоматизации управления компрессорной станцией (ПО КСАУКС) 86246294.50 5200 002–01. – СПб.: Петербургский государственный университет путей сообщения императора Александра I, 2010. – 25 с.

57. Хомоненко А. Д., Бубнов В. П., Басыров А. Г. Модели и методы исследования информационных систем: Монография. – СПб.: Издательство “Лань”, 2019. – 204 с.

58. Хомоненко А. Д., Данилов А. А., Данилов А. И. Нестационарные модели стратегий испытаний программных средств при вероятностных параметрах обнаружения ошибок // Информационно-управляющие системы. 2015. № 4. С. 50–58.

59. Хомоненко А. Д., Данилов А. И., Данилов А. А. Динамические модели испытаний программных средств // Труды Международной конференция по мягким вычислениям и измерениям (SCM-2015) (Санкт-Петербург, 19–21 мая 2015 г.). – Санкт-Петербург, 2015. – С. 239–242.

60. Данилов А. А. Комплекс программ оценивания надежности и планирования разработки программных средств на основе динамических моделей // Интеллектуальные технологии на транспорте. 2017. № 4. С. 18–24.

61. Fehlberg E. Low-order classical Runge—Kutta formulas with step size control and their application to some heat transfer problems // NASA Technical Report 315 (1969), extract published in Computing. 1970. Vol. 6. P. 61–71.

62. Бубнов В. П. Алгоритм аналитического расчета вероятностей состояний нестационарных систем обслуживания // Известия Петербургского университета путей сообщения. 2011. № 4. С. 90–97.

63. Бубнов В. П., Еремин А. С., Сергеев С. А. Особенности программной реализации численно-аналитического метода расчёта моделей нестационарных систем обслуживания // Труды СПИИРАН. 2015. № 1. С. 218–232.

64. Class BigDecimal // Java Documentation [Электронный ресурс]. 2020. – URL: http://docs.oracle.com/javase/7/docs/api/java/math/BigDecimal.html (дата обращения: 10.05.2020).

65. Сергеев С. А., Бубнов В. В., Ерёмин А. С. Программа для расчёта вероятностей состояний нестационарных систем обслуживания // Свидетельство о государственной регистрации программы для ЭВМ № 2014662753. 2014.

66. Сергеев С. А. Нестационарные модели компонентов системы автоматизированного мониторинга технического состояния искусственных сооружений: дис. … к.т.н.: 05.13.18. – СПб, 2016. – 385 с.

67. Сергеев С. А., Бубнов В. П., Бубнов В. В. Программный комплекс аналитико-имитационных моделей для расчёта вероятностно-временных характеристик нестационарных систем обслуживания // Свидетельство о государственной регистрации программы ЭВМ № 201561725. 2015.

68. Сергеев С. А., Бубнов В. П., Бубнов В. В. Программа для имитационного моделирования нестационарных систем обслуживания // Свидетельство о государственной регистрации программы ЭВМ № 201561735. 2015.

69. Бубнов В. П., Ходаковский В. А., Сергеев С. А., Соловьева В. Г. Нестационарная сетевая модель управляющего аппаратно-программного комплекса // Автоматика на транспорте. 2018. Том 4. № 2. С. 208–222.

70. Бубнов В. П., Хомоненко А. Д., Сергеев С. А. Рекурсивный метод генерации матрицы коэффициентов системы однородных дифференциальных уравнений, описывающих нестационарную систему обслуживания // Труды

Международной конференция по мягким вычислениям и измерениям (SCM-2015) (Санкт-Петербург, 19–21 мая 2015 г.). – СПб., 2015. – С. 164–166.

71. Сергеев С. А. Метод составления систем однородных дифференциальных уравнений для расчёта вероятностно-временных характеристик нестационарных систем обслуживания // Интеллектуальные технологии на транспорте. 2015. № 2. С. 32–42.

72. Бубнов В. П., Сергеев С. А. Нестационарные модели локального сервера автоматизированной системы мониторинга искусственных сооружений // Труды СПИИРАН. 2016. № 2. С. 102–115.

73. Shardakov, K. S., Bubnov V. P., Pavlov A. N. Generating of the Coefficient Matrix of the System of Homogeneous Differential Equations. // Workshop Computer Science and Engineering in the framework of the 5th International Scientific-Methodical Conference "Problems of Mathematical and Natural-Scientific Training in Engineering Education". 2018. P. 42–47.

References

1. Khinchin A. Ia. *Raboty po matematicheskoi teorii massovogo obsluzhivaniia* [Works on the mathematical theory of queuing]. Moscow, Fizmatlit Publ., 1963. 236 p. (in Russian).

2. Takacs K. Investigation of Waiting Time Problems by Reduction to orkov processes. *Acta Mathematica Academiae Scientarium Hungaicae*, 1955, vol. 6, pp. 101–129 (in Russian).

3. Gnedenko B. V., Kovalenko I. N. *Vvedenie v teoriiu massovogo obsluzhivaniia* [Introduction to Queuing Theory]. Moscow, Nauka Publ., 1966. 432 p. (in Russian).

4. Kovalenko I. N. O sisteme massovogo obsluzhivaniia so skorost'iu obsluzhivaniia, zavisiashchei ot chisla trebovanii v sisteme, i periodicheskim otkliucheniem kanalov [About the queuing system with the speed of service, depending on the number of requirements in the system, and periodic shutdown of channels]. *Problems of Information Transmission*, 1971, vol. 7, no. 2, pp. 108–114 (in Russian).

5. Kovalenko I. N. O vosstanovlenii kharakteristik sistemy po nabliudeniiam nad vykhodiashchim potokom [About restoration of system characteristics from observations of the outgoing stream]. *Doklady Akademii Nauk SSSR*, 1965, vol. 164, no. 5, pp. 979–981(in Russian).

6. Ezhov I. I., Korneichuk M. T., Oliinyk I. D. Raspredelenie kolichestva kanalov sistemy remonta, kogda intensivnost' potoka izmeniaetsia spetsial'nym obrazom *obrazom* [Distribution of the number of repair system channels when the flow rate changes in a special way]. *Kibernetika*, 1976, no. 3, pp. 92–97 (in Russian).

7. Abol'nikov L. M. Nestatsionarnaia zadacha massovogo obsluzhivaniia dlia sistem s beskonechnym chislom kanalov pri gruppovom postuplenii trebovanii [Non-stationary queuing problem for systems with an infinite number of channels with group receipt of requirements]. *Problems of Information Transmission*, 1968, vol. 4, no. 3, pp. 99–102 (in Russian).

8. Arsenishvili G. L. Odnolineinaia sistema massovogo obsluzhivaniia s zavisiashchei ot velichiny ocheredi intensivnost'iu vkhodiashchego potoka [Single-line queuing system with queue-dependent incoming flow rate]. *Soobshcheniia Akademii nauk Gruzinskoi SSR*, 1974, vol. 76, no. 2, pp. 285–288 (in Russian).

9. Conolly B. W. Generalized State Dependent Eriangian Queues (speculation about calculating easure of effectiveness). *Applied Probability*, 1975, no. 2. pp. 358–363.

10. Hadidi N. A. A queueing model with variable arrival rates. *Periodica Mathematica Hungarica*, 1975, no. 1, pp. 39–47.

11. Sigolov G. G., Liupersol'skii A. M. Metod rascheta perekhodnykh protsessov v setevykh modeliakh massovogo obsluzhivaniia *obsluzhivaniya* [Method for calculating transients in network queuing models]. *Sbornik tezisov vsesoiuznoi shkoly-seminara po raspredelennym avtomatizirovannym sistemam massovogo obsluzhivaniia* [Proceedings of the All-Union School-Seminar on Distributed Automated Queuing Systems]. Riga, 1988, pp.80–82 (in Russian).

12. Sigolov G. G., Liupersol'skii A. M. Metod priblizhennogo rascheta perekhodnykh protsessov v setevykh modeliakh massovogo obsluzhivaniia [Method for calculating transients in network queuing models]. *Avtomatika i vychislitel'naia tekhnika*, 1990, no. 3, pp. 40–43 (in Russian).

13. Skliarevich F. A. Kharakteristika prebyvaniia fragmenta vychislitel'noi seti v fiksirovannom podmnozhestve sostoianii [Characteristics of the stay of a fragment of a computer network in a fixed subset of states]. *Automatic Control and Computer Sciences*, 1985, no. 2, pp. 14–20 (in Russian).

14. Golovko N. I., Karetnik V. O. Peleshok A. V. SMO s beskonechnym nakopitelem i skachkoobraznoi intensivnost'iu vkhodnogo potoka [QS with infinite drive and spasmodic input stream intensity]. *Automation and Remote Control*, 2009, no. 10, pp.75–96 (in Russian).

15. Zeifman A. I. O nestatsionarnoi modeli Erlanga [About the non-stationary Erlang model]. *Automation and Remote Control*, 2009, no. 12, pp.71–80 (in Russian).

16. Gelenbe E., Mitrani I. *Analysis and Synthesis of computer systems*. Academic Press, Mitrani Israel, 1980. 239p.

17. Zeifman A., Korotysheva A., Satin Y. On stability for Mt/Mt/N/N queue. *Proceedings of the International Conference on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*. Moscow, 2010, pp. 1102–1105.

18. Bubnov V. P., Safonov V. I, Smagin V. A. O zagruzke vychislitel'noi sistemy s izmeniaiushcheisia intensivnost'iu postupleniia zadanii [About the loading of a computing system with a varying intensity of receipt of tasks]. *Automatic Control and Computer Sciences*, 1987, no. 6, pp. 19–22 (in Russian).

19. Bubnov V. P., Safonov V. I. Algoritm issledovaniia modeli nestatsionarnykh sistem obsluzhivaniia [An algorithm for studying the model of non-stationary queuing systems]. *Sbornik tipovykh algoritmov i program*, 1987, no. 8, pp. 172–178 (in Russian).

20. Bubnov V. P., Safonov V. I. *Razrabotka dinamicheskikh modelei nestatsionarnykh sistem obsluzhivaniia* [Development of dynamic models of non-

stationary queuing systems]. Saint-Petersburg, Izdatel'stvo “Lan'”, 1999. 64 p. (in Russian).

21. Bubnov V. P., Mikhailov A. V., Safonov V. I. *Ustroistvo dlia modelirovaniia protsessa obsluzhivaniia zaiavok* [Device for modeling the process of servicing applications]. Avtorskoe svidetel'stvo SSSR, no. SU1341648A1, 1986.

22. Puchkov V. V., Smagin V. A., Bubnov V. P., Safonov V. I. *Ustroistvo dlia modelirovaniia sistem massovogo obsluzhivaniia* [Device for modeling queuing systems]. Avtorskoe svidetel'stvo SSSR, no. SU1399756A1, 1988.

23. Bubnov V. P., Mikhailov A. V., Safonov V. I., Khapalov I. L. *Ustroistvo dlia modelirovaniia sistem massovogo obsluzhivaniia* [Device for modeling queuing systems]. Avtorskoe svidetel'stvo SSSR, no. SU1405071A1, 1988.

24. Safonov V. I., Bubnov V. P., Tkachev S. B., Belugin G. P. *Ustroistvo dlia modelirovaniia sistem massovogo obsluzhivaniia* [Device for modeling queuing systems]. Avtorskoe svidetel'stvo SSSR, no. SU1652979A1, 1991.

25. Bubnov V. P., Sergeev A. S. *Approximation method for studying non-Markov systems*. Saint-Petersburg, Emperor Alexander I St. Petersburg State Transport University Publ., 2014. 37 p. (in Russian).

26. Cox D. R. A use of complex probabilities in the theory of stochastic processes. *Proceedings of the Cambridge Philosophical*, 1955, vol. 51, no. 2, pp. 313–319.

27. Bubnov V. P., Burtseva K. I. *Nestatsionarnaia model' dlia otsenki funktsional'noi bezopasnosti sredstv zheleznodorozhnoi avtomatiki* [Non-stationary model for assessing the functional safety of railway automation]. *Problemy matematicheskoi i estestvenno nauchnoi podgotovki v inzhenernom obrazovanii. Istoricheskii opyt, sovremennye vyzovy: sbornik trudov Mezhdunarodnoi nauchno-metodicheskoi konferentsii* [Problems of mathematical and naturally scientific training in engineering education. Historical experience, modern challenges. Proceedings of International Scientific and Methodological Conference]. Saint-Petersburg, 2011, pp. 25–33. (in Russian).

28. Tyrva A. V. *Modeli nadezhnosti i planirovaniia ispytanii programmnykh sredstv.* Diss. kand. tehn. nauk [Reliability Models and Software Test Planning. Ph.D. Tesis]. Saint-Petersburg, 2010. 147 p. (in Russian).

29. Darcy D. P., Kemerer C. F. OO Metrics in Practice. *Software*, 2005, vol. 22, no. 6, pp. 17–19.

30. Dick S., Bethel C. L., Kandel A. Software-Reliability Modeling: The Case for Deterministic Behavior . *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, 2007, vol. 37, no. 1, pp. 106–119.

31. El-Emam K. Object-Oriented Metrics: A Review of Theory and Practice. *Advances in software engineering*, 2002, no. 1, pp. 23–50.

32. El-Emam K., Melo W., Machado J. C. The prediction of faulty classes using object-oriented design metrics. *Journal of Systems and Software*, 2001, no. 1, pp. 63–75.

33. Musa J. D. The Measurement and Management of Software Reliability. *Proceedings of the IEEE*, 1980, vol. 68, no. 9, pp. 1131–1143.

34. Tyrva A. V., Khomonenko A. D. Metod planirovaniia testirovaniia slozhnykh programmnykh kompleksov na etapakh proektirovaniia i razrabotki [Test planning method for complex software systems at the design and development stages]. *St. Petersburg State Polytechnical University Journal*, 2009, no. 4, pp. 125–131 (in Russian).

35. Tyrva A. V., Khomonenko A. D. Planirovanie provedeniia testirovaniia kompleksov programm na osnove proektnykh metrik slozhnosti [Planning for testing software packages based on project complexity metrics]. *Trudy Vserossiiskoi nauchno-prakticheskoi konferentsii «Transport-2009»* [Proceedings of All-Russian scientific-practical conference "Transport-2009]. Rostov-on-Don, 2009, pp. 80–82 (in Russian).

36. Jelinski Z., Moranda P. B. Software Reliability Research. *Proceedings of the Statistical Methods for the Evaluation of Computer System Performance*, 1972, pp. 465–484.

37. Huang C. Y., Huang W. C. Software Reliability Analysis and Measurement Using Finite and Infinite Server Queuing Models. *IEEE Transactions On Reliability*, 2008, vol. 57, no. 1, pp. 192–203.

38. Genero M., Piattini M., Caleron C. A survey of metrics for UML class diagrams. *Journal of Object Technology*, 2005, vol. 4, no. 9, pp. 59–92.

39. Guo P., Lyu M. R. Software Quality Prediction Using Mixture Models with EM Algorithm. *Proceedings of the The First Asia-Pacific Conference on Quality Software*, 2000, pp. 69–78.

40. Bansiya J., Davis C. A Hierarchical Model for Object-Oriented Design Quality Assessment. *IEEE Transactions on Software Engineering*, 2002, vol. 28, no. 1, pp. 4–17.

41. Basili V. R., Briand L. C. A validation of object-oriented design metrics as quality indicators. *IEEE Transactions on Software Engineering*, 1996, vol. 22, no. 10, pp. 751–761.

42. Briand L., Devanbu, P., Melo. W. An Investigation into Coupling Measures for C++. *Proceedings of the 1997 (19th) International Conference on Software Engineering*, 1997, pp. 412–421.

43. Chidamber S. R., Kemerer C. F. A metrics suite for object oriented design. *IEEE Transactions on Software Engineering*, 1994, vol. 20, no. 6, pp. 476–493.

44. Tyrva A. V., Khomonenko A. D. Metod planirovaniia testirovaniia slozhnykh programmnykh kompleksov na etapakh proektirovaniia i razrabotki [Test planning method for complex software systems at the design and development stages]. *St. Petersburg State Polytechnical University Journal*, 2009, no. 4, pp. 125–131 (in Russian).

45. Tyrva A. V. Metodika zadaniia iskhodnykh dannykh dlia modelei nadezhnosti programmnykh sredstv zheleznodorozhnogo transporta [The methodology for setting the source data for the reliability models of software for railway transport]. *Proceedings of Petersburg Transport University*, 2010, no. 2, pp. 250–261 (in Russian).

46. Tyrva A. V. *Metodika sertifikatsionnykh ispytanii programmnogo obespecheniia sistemy «Gorochnaia avtomaticheskaia tsentralizatsiia*

mikroprotsessornaia s vedeniem nakopleniia vagonov v sortirovochnom parke (GATs MN)» 86246294.50 5200 020–0 [The methodology of certification tests of the software of the system "Automatic centralization microprocessor with the accumulation of cars in the sorting fleet (GAC MN) 86246294.50 5200 020–0"]. Saint-Petersburg, Emperor Alexander I St. Petersburg State Transport University Publ., 2009. 18p. (in Russian).

47. Bubnov V. P., Tyrva A. V., Khomonenko A. D. Model of reliability of the software with Coxian distribution of length of intervals between the moments of detection of errors. *Proceedings of 34th Annual IEEE Computer Software and Applications Conference (COMPSAC 2010).* Seoul, 2010, pp. 238–243.

48. Bubnov V. P., Tyrva A. V., Khomonenko A. D. Software Reliability Model with Coxian Distribution of Length of Intervals Between Errors Detection and Fixing Moments. *Proceedings of 35th Annual IEEE Computer Software and Applications Conference (COMPSAC 2011).* Munich, 2011, pp. 310–314.

49. Tyrva A. V., Khomonenko A. D., Bubnov V. P. *Modeli nadezhnosti programmnogo obespecheniia* [Software Reliability Models]. Saint-Petersburg, Emperor Alexander I St. Petersburg State Transport University, 2010. 39 p. (in Russian).

50. Bubnov V. P., Tyrva A. V., Burtseva K. I. Nestatsionarnaia model' nadezhnosti programmnykh sredstv s raspredeleniem Koksa dlin intervalov vremeni ispravleniia oshibok *oshibok* [Non-staitonary model of software reliability with Cox distribution of error correction time intervals]. *Vestnik Vserossiiskogo nauchno-issledovatel'skogo i proektno-konstruktorskogo instituta elektrovozostroeniia*, 2010, no. 1, pp. 143–152 (in Russian).

51. Bubnov V. P. Kompleks modelei nadezhnosti programmnykh sredstv s raspredeleniiami fazovogo tipa [A complex of software reliability models with phase-type distributions]. *Vestnik Vserossiiskogo nauchno-issledovatel'skogo i proektno-konstruktorskogo instituta elektrovozostroeniia*, 2011, no. 1, pp. 185–193 (in Russian).

52. Bubnov V. P., Tyrva A. V., Eremin A. S. Kompleks modelei nestatsionarnykh sistem obsluzhivaniia s raspredeleniiami fazovogo tipa [A complex of models of non-stationary queuing systems with phase-type distributions]. *SPIIRAS Proceedings*, 2014, no. 6, pp. 61–71 (in Russian).

53. Bubnov V. P., Tyrva A. V., Khomonenko A. D. Obosnovanie strategii otladki programm na osnove nestatsionarnoi modeli nadezhnosti [Justification of a strategy for debugging programs based on a non-stationary reliability model]. *St. Petersburg State Polytechnical University Journal. Computer Science. Telecommunications and Control Systems*, 2010, no. 2, pp. 85–92 (in Russian).

54. Tyrva A. V., Khomonenko A. D., Bubnov V. P. *Kompleks programm rascheta nadezhnosti i planirovaniia ispytanii programmnykh sredstv. Svidetel'stvo ob ofitsial'noi registratsii programm dlya EVM* [A set of programs for calculating reliability and test software planning. The Certificate on Official Registration of the Computer Program]. No. 2010615617, 2010.

55. Dashonok V. L., Tyrva A. V. *Metodika sertifikatsionnykh ispytanii programmnogo obespecheniia «Testiruiushchii kompleks sistem mikroprotsessornykh*

tsentralizatsii i sistem mikroprotsessornykh avtoblokirovok s tsentralizovannym razmeshcheniem oborudovaniia (TK MPTs/ABTMPTs). Programmnoe obespechenie tipovoe» 01095505.42 5210–01 [Software certification test methodology "Testing complex of microprocessor centralization systems and microprocessor self-locking systems with centralized equipment placement (TC MPC / ABTMPC). Typical software" 01095505.42 5210–01]. Saint-Petersburg, Emperor Alexander I St. Petersburg State Transport University Publ., 2010. 32 p. (in Russian).

56. Tyrva A. V. *Metodika sertifikatsionnykh ispytanii programmnogo obespecheniia kompleksnoi sistemy avtomatizatsii upravleniia kompressornoi stantsiei (PO KSAUKS) 86246294.50 5200 002–01* [Methodology for certification testing of software for an integrated compressor station control automation system (software KSAUKS) 86246294.50 5200 002–01]. Saint-Petersburg, Emperor Alexander I St. Petersburg State Transport University Publ., 2010. 25 p. (in Russian).

57. Khomonenko A. D., Bubnov V. P., Basyrov A G. *Modeli i metody issledovaniia informatsionnykh system. Monografiia* [Models and methods of researching information systems. Monography]. Saint-Petersburg, Izdatel'stvo "Lan'", 2019. 204 p. (in Russian).

58. Khomonenko A. D., Danilov A. A., Danilov A. I. Nestatsionarnye modeli strategii ispytanii programmnykh sredstv pri veroiatnostnykh parametrakh obnaruzheniia oshibok [Non-stationary models of software testing strategies with probabilistic error detection parameters]. *Informatsionno-upravliaiushchie sistemy*, 2015, no. 4, pp. 50–58 (in Russian).

59. Khomonenko A. D., Danilov A. I., Danilov A. A. Dinamicheskie modeli ispytanii programmnykh sredstv [Dynamic Software Testing Models]. *Proceedings of 2015 XVIII International Conference on Soft Computing and Measurements (SCM-2015)*. Saint-Petersburg, 2015, vol. 1, pp. 239–242 (in Russian).

60. Danilov A. A. Kompleks programm otsenivaniia nadezhnosti i planirovaniia razrabotki programmnykh sredstv na osnove dinamicheskikh modelei modelej [A set of programs for assessing reliability and planning software development based on dynamic models]. *Intellectual Technologies on Transport*, 2017, no. 4, pp. 18–24 (in Russian).

61. Fehlberg E. Low-order classical Runge—Kutta formulas with step size control and their application to some heat transfer problems. *NASA Technical Report 315 (1969), extract published in Computing*, 1970, vol. 6, no. 1–2, pp. 61–71.

62. Bubnov V. P. Algoritm analiticheskogo rascheta veroiatnostei sostoianii nestatsionarnykh sistem obsluzhivaniia obsluzhivaniya [Algorithm for the analytical calculation of the probabilities of non-stationary queuing systems]. *Proceedings of Petersburg Transport University*, 2011, no. 4, pp. 90–97 (in Russian).

63. Bubnov V. P., Eremin A. S., Sergeev S. A. Osobennosti programmnoi realizatsii chislenno-analiticheskogo metoda rascheta modelei nestatsionarnykh sistem obsluzhivaniia [Features of software implementation of a numerical-analytical method for calculating models of non-stationary queuing systems]. *SPIIRAS Proceedings*, 2015, no. 1, pp. 218–232 (in Russian).

64. Class BigDecimal. *Java Documentation.* Available at: http://docs.oracle.com/javase/7/docs/api/java/math/BigDecimal.html (accessed at 09.09.2016).

65. Sergeev S. A., Bubnov V. V., Eremin A. S. *Programma dlia rascheta veroiatnostei sostoianii nestatsionarnykh sistem obsluzhivaniia. Svidetel'stvo ob ofitsial'noi registratsii programm dlya EVM* [The program for calculating the probabilities of non-stationary queuing systems. The Certificate on Official Registration of the Computer Program]. No. 2014662753, 2014.

66. Sergeev S. A. Nestatsionarnye modeli komponentov sistemy avtomatizirovannogo monitoringa tekhnicheskogo sostoianiia iskusstvennykh sooruzhenii. Diss. kand. tehn. nauk [Non-stationary models of components of a system for automated monitoring of the technical condition of artificial structures. Ph.D. Tesis]. Saint-Petersburg, 2016, 385 p. (in Russian).

67. Sergeev S. A., Bubnov V. P., Bubnov V. V. *Programmnyi kompleks analitiko-imitatsionnykh modelei dlia rascheta veroiatnostno-vremennykh kharakteristik nestatsionarnykh sistem obsluzhivaniia. Svidetel'stvo ob ofitsial'noi registratsii programm dlya EVM* [The software package of analytical and simulation models for calculating the probability-time characteristics of non-stationary queuing systems. The Certificate on Official Registration of the Computer Program]. No. 201561725, 2015.

68. Sergeev S. A., Bubnov V. P., Bubnov V. V. *Programma dlia imitatsionnogo modelirovaniia nestatsionarnykh sistem obsluzhivaniia*. *Svidetel'stvo ob ofitsial'noi registratsii programm dlya EVM* [The program for simulation of non-stationary queuing systems. The Certificate on Official Registration of the Computer Program]. No. 201561735, 2015.

69. Bubnov V. P., Khodakovskii V. A., Sergeev S. A., Solov'eva V. G. Nestatsionarnaia setevaia model' upravliaiushchego apparatno-programmnogo kompleksa [Non-stationary network model of the controlling hardware-software complex]. *Avtomatika na transporte*, 2018, vol. 4, no. 2, pp. 208–222 (in Russian).

70. Bubnov V. P., Khomonenko A. D., Sergeev S. A. Rekursivnyi metod generatsii matritsy koeffitsientov sistemy odnorodnykh differentsial'nykh uravnenii, opisyvaiushchikh nestatsionarnuiu sistemu obsluzhivaniia [A recursive method for generating a matrix of coefficients of a system of homogeneous differential equations describing a non-stationary queuing system]. *Proceedings of 2015 XVIII International Conference on Soft Computing and Measurements (SCM-2015)*. Saint-Petersburg, 2015, vol. 1, pp. 164–166 (in Russian).

71. Sergeev S. A. Metod sostavleniia sistem odnorodnykh differentsial'nykh uravnenii dlia rascheta veroiatnostno-vremennykh kharakteristik nestatsionarnykh sistem obsluzhivaniia [A method for compiling systems of homogeneous differential equations for calculating the probability-time characteristics of non-stationary queuing systems]. *Intellectual Technologies on Transport*, 2015, no. 2, pp. 32–42.

72. Bubnov V. P., Sergeev S. A. Nestatsionarnye modeli lokal'nogo servera avtomatizirovannoi sistemy monitoringa iskusstvennykh sooruzhenii [Non-stationary models of a local server of an automated monitoring system for artificial structures]. *SPIIRAS Proceedings*, 2016, no. 2, pp. 102–115 (in Russian).

73. Shardakov K. S., Bubnov V. P., Pavlov A. N. Generating of the Coefficient Matrix of the System of Homogeneous Differential Equations. *Workshop Computer Science and Engineering in the framework of the 5th International Scientific-Methodical Conference "Problems of Mathematical and Natural-Scientific Training in Engineering Education"*. Saint-Petersburg, 2018, pp. 42–47.

Статья поступила 28 мая 2020 г.

Информация об авторах

Бубнов Владимир Петрович – доктор технических наук, профессор. Профессор кафедры «Информационные и вычислительные системы». Петербургский государственный университет путей сообщения императора Александра I. Область научных интересов: вероятностные модели, решение систем дифференциальных уравнений. E-mail: bubnov1950@yandex.ru

Адрес: 190031, Россия, г. Санкт-Петербург, Московский пр., д. 9.

Сафонов Владимир Иванович – доктор технических наук. Эксперт в федеральном бюджетном учреждении «Регистр сертификации на федеральном железнодорожном транспорте». Область научных интересов: моделирование систем обслуживания, архитектура систем управления. E–mail: safonov1953@yandex.ru

Адрес: 128365, Россия, г. Москва, ул. Барышиха, дом 25–1.

Шардаков Кирилл Сергеевич – соискатель ученой степени кандидата технических наук. Аспирант кафедры «Информационные и вычислительные системы». Петербургский государственный университет путей сообщения императора Александра I. Область научных интересов: нестационарные системы обслуживания. E-mail: k.shardakov@gmail.com

Адрес: 190031, Россия, г. Санкт-Петербург, Московский пр., д. 9.

Overview of existing models of non-stationary queuing systems and methods for their calculation

V. P. Bubnov, V. I. Safonov, K. S. Shardakov

Relevance. Currently, hardware-software complexes (HSC) included in the control loop of moving objects and technological processes are increasingly operating under peak loads. This is due, on the one hand, to an increase in threats caused by technogenic, natural, and human factors, and, on the other hand, the desire to use existing HSC to solve new more complex problems. To determine the feasibility of all operations related to the control technology cycle, mathematical modeling is used at a given time interval. However, classical stationary models of queuing theory do not allow modeling in the case of peak changes in the workload at a given (directive) time interval, which required the development of models of non-stationary queuing systems (NSQS). **The purpose** of the work is to analyze the main publications aimed at developing the theoretical foundations, models and methods for calculating the models of NSQS. **Results.** The article presents the results of the systematization and analysis of NSQS models with a finite number of applications, the rules for their construction and methods for calculating the probability-time characteristics.

Elements of the novelty of the work are two approaches to the construction and calculation of NSQS models, a complete list of NSQS models developed to date, a comparative analysis of the methods for calculating probability-time characteristics and a discussion of the features of their software implementation. **Practical significance.** The material of the article can be effectively used in the problems of analyzing the implementation of HSC management operations located in the control loop of technological processes and moving objects, calculating the indicators of their functional reliability. It is applicable to increase the accuracy of calculating the reliability indicators of software tools, to reduce the time of their testing based on the choice of a debugging strategy. And also for the further construction of new NSQS models and methods for their calculation.

Key words: non-stationary queuing system, NSQS, transition and state diagram, system of homogeneous differential equations, Chapman-Kolmogorov equation, Markov process, application, software reliability growth model.

Information about Authors

Vladimir Petrovich Bubnov – Dr. habil. of Engineering Sciences, Full Professor. Professor at the Department of Information and Computing Systems. Emperor Alexander I St. Petersburg State Transport University. Field of research: probabilistic models, solving systems of differential equations. E-mail: bubnov1950@yandex.ru

Address: Russia, 190031, Saint-Petersburg, Moskovskiy avenue, 9.

Vladimir Ivanovich Safonov – Dr. habil. of Engineering Sciences. Federal Budgetary Organization «Register of Certification on the Federal Railway Transport». Field of research: service system modeling, management system architecture. E-mail: safonov1953@yandex.ru

Address: Russia, 128365, Moscow, Baryshiha Street., 25–1.

Kirill Sergeevich Shardakov – postgraduate student. Postgraduate at the Department of Information and Computing Systems. Emperor Alexander I St. Petersburg State Transport University. Field of research: non-stationary queuing systems. E-mail: k.shardakov@gmail.com

Address: Russia, 190031, Saint-Petersburg, Moskovskiy avenue, 9.