

М.К. До, В.А. БОГАТЫРЕВ
**ПОЛУМАРКОВСКАЯ МОДЕЛЬ НАДЕЖНОСТИ
КОМПЬЮТЕРНОЙ СИСТЕМЫ С КОНТЕЙНЕРНОЙ
ВИРТУАЛИЗАЦИЕЙ**

До М.К., Богатырев В.А. Полумарковская модель надежности компьютерной системы с контейнерной виртуализацией.

Аннотация. В современных облачных инфраструктурах и микросервисных архитектурах критически важной задачей является обеспечение высокой надежности, отказоустойчивости и непрерывности функционирования приложений. Основным фактором, снижающим эффективность таких систем, выступает феномен программного старения – постепенная деградация системных ресурсов (утечки памяти, фрагментация данных) в процессе длительной эксплуатации, приводящая к нелинейному росту интенсивности отказов. Сложность прогнозирования надежности в таких условиях обусловлена иерархической зависимостью между общим программным обеспечением хоста и изолированными контейнерами, а также динамическим характером процессов восстановления. В данной статье рассматривается актуальная проблема обеспечения надежности программного обеспечения (ПО) в компьютерной системе с контейнерной виртуализацией, подверженной явлению программного старения. Научная новизна заключается в разработке комплексной математической модели на основе полумарковских процессов, которая учитывает иерархическую структуру системы (общее ПО хоста и изолированные контейнеры) и позволяет оптимизировать стратегии проактивного восстановления. Для описания процесса старения использовано распределение Вейбулла, что отражает возрастающую интенсивность отказов. Предложена полумарковская модель состояний и переходов, включающая механизмы проактивного перезапуска и миграции контейнеров. Для снижения вычислительной сложности разработана упрощенная модель, базирующаяся на допущении о пренебрежимо малой вероятности вторичных отказов в короткие интервалы восстановления. Расчеты стационарных вероятностей и коэффициента готовности реализованы в среде Mathcad. Проведенные численные эксперименты показали, что реализация стратегии проактивного восстановления в диапазоне интервалов 1200-2400 часов существенно повышает коэффициент готовности системы. Установлено, что погрешность упрощенной модели пренебрежимо мала по сравнению со строгой моделью. Предложенная модель позволяет определить максимальное значение коэффициента готовности в зависимости от числа развернутых контейнеров и интервалов проактивного перезапуска контейнеров. Сравнение разработанного подхода с традиционными реактивными моделями восстановления подтверждает преимущество проактивных методов в минимизации простоев, вызванных накоплением ошибок (утечки памяти, фрагментация ресурсов). Предложенная модель может быть применена при проектировании высоконадежных облачных инфраструктур и DevOps-конвейеров.

Ключевые слова: надежность, коэффициент готовности, программное обеспечение, контейнерная виртуализация, полумарковская модель, проактивное восстановление, перезапуск контейнеров.

1. Введение. При разработке современных инфокоммуникационных систем первостепенное внимание уделяется требованиям по обеспечению

высокой готовности, надёжности и отказоустойчивости [1 – 3]. Актуальность данных требований обусловлена критической значимостью непрерывности бизнес-процессов и необходимостью предотвращения потенциальных потерь данных в условиях цифровой трансформации.

Достижение заданных показателей качества обслуживания в современных децентрализованных компьютерных системах и распределенных сетях [4] неразрывно связано со сложной инженерной задачей – необходимостью поддержания непрерывности вычислений и минимизации задержек при обработке и передаче гетерогенного трафика. В условиях экспоненциального роста объемов обрабатываемой информации данная задача успешно решается за счет глубокой консолидации вычислительных ресурсов в рамках кластерных архитектур. Такие архитектуры поддерживают интеллектуальные механизмы динамической реконфигурации, базирующиеся на технологиях репликации данных, оперативной миграции виртуальных машин [5, 6] и легких контейнерах [7]. Эти механизмы активируются не только в случаях возникновения аппаратных отказов, но и в процессах планового технического обслуживания, а также при реализации процедур аварийного восстановления кластерной системы. Современные кластерные решения и платформы оркестрации обеспечивают интеллектуальное автоматическое перераспределение рабочей нагрузки между физическими и виртуальными узлами в режиме реального времени. Это позволяет не только минимизировать деструктивное влияние отказов отдельных аппаратных или программных компонентов на общую производительность и пропускную способность системы [8, 9], но и гарантировать пользователям бесперебойный доступ к сервисам независимо от текущего состояния базовой инфраструктуры.

Обоснование проектных решений в распределенных кластерных системах должно опираться на построение моделей, ориентированных на оптимизацию процессов обслуживания потоков функциональных задач с учетом обеспечения требований надежности [10 – 12], своевременности и непрерывности вычислений. Для современных распределенных систем заслуживают внимания модели, учитывающие особенности систем граничных, туманных и облачных вычислений [13, 14]. Для обоснования организации современных инфокоммуникационных систем представляют интерес модели надежности кластерных компьютерных систем, отражающие процесс накопления отказов и различных дисциплин восстановления после отказов при различных дисциплинах деградации и восстановления [15 – 17], а также модели надёжности, отражающие процесс контроля и мониторинга состояний компьютерных систем.

Модели надежности кластеров должны отражать особенности процессов их физического и информационного восстановления при альтернативах разделения и совмещения этих процессов [15, 18], в том числе при миграции виртуальных машин [19, 20]. Модели двухузловых кластерных систем с учетом миграции виртуальных контейнеров при физических отказах узлов предложены в [21]. Для оценки надежности многоузловых кластерных систем из структурно сложных узлов в [22] предложен метод декомпозиции на основе построения марковской модели процессов отказов и восстановлений в одном узле. Однако следует заметить, что рассмотренные модели надежности кластерных систем ориентированы на отказы аппаратуры и не учитывают особенности миграции виртуальных контейнеров при отказах общего системного программного обеспечения узлов и программного обеспечения поддержки функционирования виртуальных контейнеров.

Современные тенденции развития информационных технологий характеризуются массовым переходом к микросервисной архитектуре и использованию контейнеризации (Docker, Kubernetes) [23]. По данным исследований последних лет, более 80% крупных IT-компаний внедрили контейнерные технологии в свою производственную инфраструктуру, а темпы роста рынка контейнеризации превышают 30% в год [24]. Глобальный рынок контейнерных технологий оценивается экспертами в миллиарды долларов и продолжает стремительно расти [25]. Контейнеризация обеспечивает ряд существенных преимуществ: быстрое развертывание приложений, эффективное использование ресурсов, портативность между различными средами исполнения и упрощение процессов непрерывной интеграции и доставки (CI/CD) [26]. Несмотря на гибкость и масштабируемость, такие системы сталкиваются с проблемой программного старения – постепенного снижения производительности и роста вероятности отказов из-за накопления ошибок в процессе непрерывной эксплуатации [27, 28]. Исследования показывают, что программное старение является одной из основных причин деградации облачных сервисов, приводящее к снижению качества обслуживания и нарушению соглашений об уровне сервиса [29]. В долгосрочной перспективе даже хорошо спроектированные системы подвержены эффектам старения, которые проявляются через увеличение времени отклика, рост потребления памяти и снижение пропускной способности [30].

В работе [31] вопросы надежности часто рассматриваются либо для физического оборудования, либо для монолитного программного

обеспечения без учета специфики контейнерной виртуализации и распределенных микросервисных архитектур. Традиционные подходы к восстановлению носят реактивный характер, т.е. активируются только после наступления отказа, что в критических системах ведет к значительным финансовым и репутационным потерям, снижению удовлетворённости клиентов и потере конкурентных преимуществ. Реактивные стратегии не способны предотвратить каскадные отказы в распределенных микросервисных архитектурах, где отказ одного компонента может привести к лавинообразному эффекту во всей системе из-за наличия сложных зависимостей между сервисами [32].

Анализ инцидентов в облачных инфраструктурах ведущих провайдеров [33] свидетельствует о том, что значительная доля отказов – от 40% до 60% по различным оценкам – вызвана не только дефектами кода, но и явлением программного старения. Данный феномен проявляется в постепенной деградации системных ресурсов – утечках оперативной памяти, фрагментации файловой системы, переполнении буферов, накоплении устаревших записей в таблицах маршрутизации и базах данных, что неизбежно приводит к снижению производительности и зависанию контейнеров. Исследования подтверждают, что даже при отсутствии логических ошибок в коде приложения, длительная эксплуатация контейнеров приводит к накоплению временных файлов, росту числа открытых дескрипторов файлов и сетевых соединений, увеличению фрагментации памяти и деградации производительности кэшей [8].

Применение традиционных моделей надежности программного обеспечения (ПО) к компьютерной системе с контейнерной виртуализацией сталкивается с серьезными ограничениями и зачастую не позволяет получить адекватные оценки характеристик надежности. Модель Джелинского-Моранды (JM) основывается на предположении, что интенсивность отказов пропорциональна числу оставшихся ошибок в программном коде, и после каждого исправления дефекта надежность системы мгновенно возрастает [34]. Однако в компьютерной системе с контейнерной виртуализацией отказы часто вызваны исчерпанием системных ресурсов в процессе длительной работы, даже при полном отсутствии логических ошибок в коде приложения. Исправление программных багов не останавливает процесс накопления «мусора» и деградации ресурсов операционной системы, что делает модель JM принципиально неприменимой для описания долгоживущих облачных сессий и непрерывно работающих сервисов [35].

Модели Мусы и Шумана рассматривают программную систему как единый независимый монолит с фиксированной архитектурой [36]. Контейнеры же, в отличие от виртуальных машин с полной изоляцией, используют общее ядро хостовой операционной системы и разделяют системные ресурсы. Это создает проблему «шумных соседей» и недостаточной изоляции, когда отказ или утечка ресурсов в одном контейнере оказывает существенный побочный эффект на другие сервисы и хост-систему в целом, приводя к снижению их производительности и увеличению вероятности отказов. Традиционные модели не учитывают эту корреляцию отказов и взаимное влияние компонентов в мультитенантных средах с разделяемыми ресурсами. Экспериментальные исследования показывают, что взаимное влияние может приводить к снижению производительности до 40% [37].

Модель Миллса (Mills model, модель с посевом ошибок) предполагает статичное состояние среды выполнения и постоянную конфигурацию системы [38]. Контейнерные оркестраторы (например, Kubernetes, Docker Swarm, Apache Mesos) характеризуются высокой динамикой: экземпляры приложений постоянно масштабируются в соответствии с нагрузкой, перезапускаются при обнаружении отказов и мигрируют между узлами кластера для балансировки нагрузки и оптимизации использования ресурсов [39]. Использование фиксированной интенсивности отказов в таких динамических условиях приводит к значительным погрешностям прогнозирования, которые могут достигать порядка величины. Современные облачные системы требуют моделей, способных адаптироваться к изменяющимся условиям нагрузки, конфигурации инфраструктуры и топологии развертывания [40].

В связи с вышеизложенным, для моделирования надежности сервисов на базе контейнеров, подверженных эффектам старения, наиболее целесообразным признано использование математического аппарата полумарковских процессов (semi-Markov processes, SMP) [41]. В отличие от классических марковских цепей, строго ограничивающих время пребывания системы в каждом состоянии экспоненциальным законом распределения, SMP позволяют использовать произвольные функции распределения и описывать переходы между качественно различными состояниями системы: нормальное функционирование, постепенное старение, критическое состояние, отказ и процесс омоложения (rejuvenation) [42]. Это критически важно, так как вероятность отказа из-за накопленного старения увеличивается со временем непрерывной работы, что свидетельствует об отсутствии

свойства отсутствия памяти, характерного для экспоненциального распределения [43].

Для математического описания процесса деградации в данной работе применяется распределение Вейбулла [44]. Оно позволяет моделировать возрастающую интенсивность отказов, характерную для фазы износа ресурсов хостовой операционной системы и среды исполнения контейнеров, что обеспечивает высокую точность прогнозирования моментов наступления критических состояний и планирования профилактических мероприятий [45]. Параметр формы распределения Вейбулла позволяет гибко настраивать модель для различных сценариев деградации системы – от постепенного до резкого старения, что делает модель универсальной для широкого класса приложений [46].

Методологическим прототипом применения аппарата SMP к оценке надёжности ПО является работа [47], в которой предложено строить граф-схему полумарковского процесса по блок-схеме алгоритма вычислительного процесса, формировать матрицу вероятностей переходов и решать систему алгебраических уравнений в среде MathCAD для нахождения среднего времени до отказа. В работе [47] ПО рассматривается как единое целое без разделения на общее ПО и ПО виртуальных контейнеров, при этом не отражено возможное возрастание интенсивности отказов при программном старении. Исследование стратегий проактивного восстановления, предполагающего плановый перезапуск контейнеров или их миграцию до наступления отказа, в [47] не проводилось. Соответственно, не рассматривалась задача оптимизации числа контейнеров и интервалов их перезапуска для обеспечения максимального коэффициента готовности кластерной системы с контейнерной виртуализацией.

Цель данной работы заключается в разработке и исследовании математической модели надёжности компьютерной системы с контейнерной виртуализацией, учитывающей эффекты программного старения, для обоснования стратегий проактивного восстановления и проектирования оптимальной конфигурации системы по количеству используемых контейнеров.

Научная новизна данной работы заключается в создании многоуровневой математической модели надёжности, которая разделяет процессы старения общего (системного) программного обеспечения и прикладных контейнеров, учитывая их взаимное влияние. В отличие от существующих решений [47 – 49], предлагаемый подход интегрирует механизмы проактивного перезапуска и миграции контейнеров в единую полумарковскую

модель с возможностью оптимизации параметров обслуживания. Преимуществом разработанного метода является возможность аналитического определения оптимальных значений числа развернутых контейнеров и интервалов их проактивного перезапуска, обеспечивающих баланс между временными затратами на профилактику и коэффициентом готовности системы. Это позволяет осуществить переход от традиционной стратегии восстановления по факту отказа к стратегии превентивного оздоровления системы (proactive rejuvenation), что существенно повышает общую надежность и доступность сервисов.

2. Процесс старения программного обеспечения. Старение программного обеспечения (ПО) – это явление постепенного снижения производительности или увеличения вероятности отказа программной системы в процессе её непрерывной работы. В отличие от физического старения, старение ПО обусловлено не механическим износом, а истощением системных ресурсов, утечками памяти, фрагментацией данных и накоплением ошибок округления.

В теории надежности данное явление характеризуется функцией интенсивности отказов, возрастающей со временем. Это фундаментально отличает ПО от электронных компонентов, которые часто подчиняются экспоненциальному закону распределения (где $\lambda = \text{const}$). Многочисленные эмпирические исследования показывают, что распределение Вейбулла является наиболее подходящей моделью для описания этого этапа [44]. В частности, при параметре формы $\beta > 1$ интенсивность отказов монотонно возрастает, что отражает свойство старения системы.

Принимается допущение, что аппаратное обеспечение является абсолютно надежным. В отказоустойчивых кластерах, использующих технологию контейнерной виртуализации, надежность ПО рассматривается как композиция двух составляющих: общего ПО и прикладного ПО контейнеров. Отказ общего ПО приводит к полной потере работоспособности всех запущенных контейнеров. Рассмотрим программную систему, состоящую из общего ПО и одного изолированного контейнера. Процессы возникновения отказов общего ПО и контейнера описываются функциями распределения Вейбулла $F_1(t)$ и $F_2(t)$ соответственно:

$$F_1(t) = 1 - e^{-\left(\frac{t}{\eta_1}\right)^{\beta_1}}, F_2(t) = 1 - e^{-\left(\frac{t}{\eta_2}\right)^{\beta_2}}, \\ R_1(t) = 1 - F_1(t), R_2(t) = 1 - F_2(t).$$

$R_1(t)$ и $R_2(t)$ представляют собой вероятности безотказной работы общего ПО и контейнера соответственно. Параметр масштаба распределения Вейбулла η представляет собой наработку, в течение которой в среднем отказывает 63,2% наблюдаемых объектов. Диаграмма состояний и переходов ПО, построенная на основе полумарковской модели, представлена на рисунке 1. В рассматриваемой модели состояние S_0 соответствует работоспособности общего ПО и контейнера. В процессе непрерывной работы программная среда контейнера подвергается старению, что со временем приводит к возникновению отказов. При отказе контейнера система переходит в состояние S_1 (восстановления контейнера) и S_2 (миграции контейнера). В случае отказа общего ПО фиксируется переход в состояние S_3 , характеризующееся необходимостью полного восстановления как хостовой платформы, так и запущенного в ней контейнера.

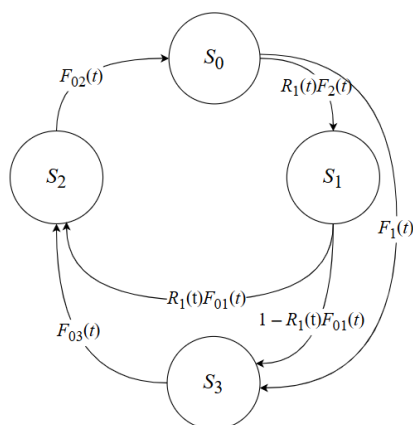


Рис. 1. Диаграмма состояний и переходов общего ПО и контейнера в условиях естественного старения

Предполагается, что процессы восстановления в состояниях S_1 , S_2 и S_3 описываются экспоненциальными законами распределения $F_{01}(t)$, $F_{02}(t)$ и $F_{03}(t)$ соответственно:

$$F_{01}(t) = 1 - e^{-\mu_1 t}, F_{02}(t) = 1 - e^{-\mu_2 t}, F_{03}(t) = 1 - e^{-\mu_3 t},$$

где μ_1 , μ_2 , μ_3 – интенсивности восстановления контейнера, миграции и полного восстановления хостовой платформы соответственно. Для расчета надежности системы строится матрица вероятностей

переходов полумарковского процесса $M_1(t)$ и вектор вероятностей состояний $v(t) = (v_0(t), v_1(t), v_2(t), v_3(t))$. Согласно графу состояний, матрица имеет вид:

$$M_1(t) = \begin{pmatrix} 1 - F_1(t) - R_1(t)F_2(t) & R_1(t)F_2(t) & 0 & F_1(t) \\ 0 & 0 & R_1(t)F_{01}(t) & 1 - R_1(t)F_{01}(t) \\ F_{02}(t) & 0 & 1 - F_{02}(t) & 0 \\ 0 & 0 & F_{03}(t) & 1 - F_{03}(t) \end{pmatrix}.$$

Вектор $v(t) = (v_0(t), v_1(t), v_2(t), v_3(t))$ является единственным решением системы линейных алгебраических уравнений:

$$\begin{cases} v(t) = v(t)M_1(t), \\ \sum_{i=0}^3 v_i(t) = 1. \end{cases}$$

Расчет вектора стационарных вероятностей был выполнен в программной среде Mathcad. Полученные значения компонентов вектора вероятностей состояний представлены на рисунке 2(а). На основе этих данных коэффициент готовности системы рассчитывается по следующей формуле:

$$K_1(\tau) = \frac{\int_0^{\tau} e^{-\left(\frac{t}{\eta_1}\right)^{\beta_1}} e^{-\left(\frac{t}{\eta_2}\right)^{\beta_2}} dt}{\int_0^{\tau} e^{-\left(\frac{t}{\eta_1}\right)^{\beta_1}} e^{-\left(\frac{t}{\eta_2}\right)^{\beta_2}} dt + v_1(\tau)t_1 + v_2(\tau)t_2 + v_3(\tau)t_3}.$$

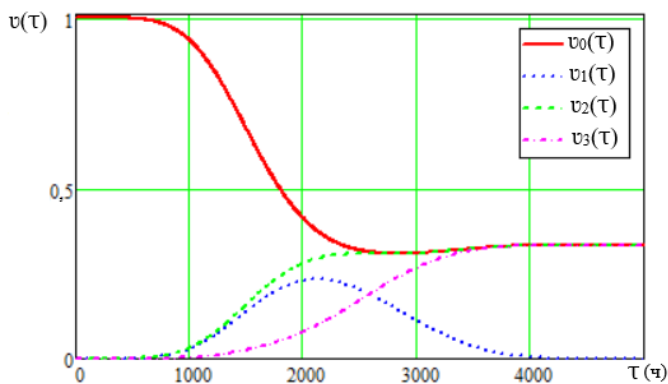
В данной формуле использование определенного интеграла в числителе обусловлено процессом программного старения, поскольку вероятность безотказной работы общего ПО $R_1(t)$ и контейнера $R_2(t)$ непрерывно снижается, а интеграл позволяет рассчитать математическое ожидание времени полезной работы системы. Здесь t – текущая наработка (внутренняя переменная интегрирования), а τ – заданная наработка системы до момента оценки её состояния (интервал наблюдения). Параметры t_1 , t_2 и t_3 представляют собой средние времена восстановления, соответствующие вероятностям пребывания системы в состояниях

S_1, S_2 и S_3 соответственно. Таким образом, знаменатель отражает полную среднюю продолжительность цикла, включая как работу, так и суммарное время восстановления. Математически коэффициент готовности представляет собой отношение среднего времени пребывания системы в работоспособном состоянии к суммарному среднему времени полного цикла «работа – отказ – восстановление».

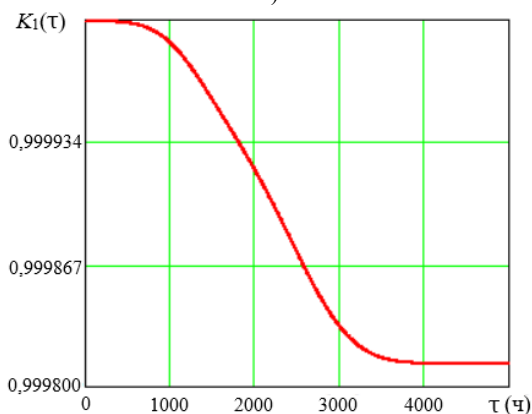
При проведении численных экспериментов были приняты следующие значения параметров. Значения параметров формы $\beta_1 = \beta_2 = 5$ выбраны на основе следующих соображений: согласно теории надёжности, значения $\beta > 4$ характерны для систем, демонстрирующих ярко выраженный износ ресурсов – режим, при котором интенсивность отказов стремительно возрастает в конце жизненного цикла [45]. Применительно к контейнерной виртуализации данный режим соответствует фазе накопленной деградации: утечкам памяти, росту числа открытых дескрипторов и истощению буферов – явлениям, экспериментально подтверждённым для платформ Docker и Kubernetes в работах [30]. Равенство $\beta_1 = \beta_2$ обусловлено тем, что общее ПО и контейнер функционируют на базе единого общего ядра и подвержены схожим механизмам деградации. Значения параметра масштаба $\eta_1 = 3000$ ч для общего ПО и $\eta_2 = 2000$ ч для контейнера отражают типичное для производственных облачных систем соотношение сроков службы. Общее ПО хоста, как правило, функционирует дольше без перезапуска, тогда как контейнер чаще подвергается накоплению ошибок вследствие интенсивных рабочих нагрузок. Конкретные числовые значения выбраны в качестве репрезентативных для демонстрации метода; для реальных систем их следует определять методом максимального правдоподобия по статистике времён до отказа, получаемой из систем мониторинга (например, из открытых наборов данных Google Cluster Data или Alibaba Cluster Data).

Значения временных параметров приняты на основе экспериментальных данных для типичных контейнеризированных систем. Время восстановления контейнера после отказа $t_1 = 0,25$ ч включает диагностику и освобождение ресурсов, что соответствует оценкам, полученным при экспериментальном исследовании программного старения в контейнерной виртуализации [30]. Время миграции контейнера $t_2 = 1/120$ ч описывает не перемещение между физическими серверами, а процесс внутриузлового переразвертывания и миграции контекста, характерный для современных платформ контейнеризации [8]. Этот процесс включает удаление старых пространств имен, которые могли быть загрязнены эффектами старения,

и инициализацию полностью нового контейнера из эталонного образа для обеспечения чистого состояния. Время восстановления общего ПО $t_3 = 1$ ч соответствует полному перезапуску хостовой ОС и среды исполнения контейнера при критическом отказе, вызванном накоплением эффектов программного старения [28, 30]. Результаты расчета коэффициента готовности системы в зависимости от длительности интервала наблюдения τ представлены на рисунке 2(б).



а)



б)

Рис. 2. Зависимость компонентов вектора вероятностей состояний $v(\tau)$ и коэффициента готовности системы $K_1(\tau)$ от длительности интервала наблюдения τ : а) компоненты вектора $v(\tau)$; б) коэффициент готовности $K_1(\tau)$

Анализ временной динамики коэффициента готовности позволяет выделить три характерные фазы. В начальный момент

времени значение K_1 равно 1, что объясняется идеальным состоянием системы, при котором все программные компоненты полностью работоспособны и ресурсы не исчерпаны. В последующий период наблюдается постепенное снижение показателя, вызванное кумулятивным эффектом старения ПО. Согласно закону Вейбулла, с течением времени интенсивность отказов возрастает, что приводит к более частому переходу системы в состояния восстановления S_1, S_2 и S_3 . Наконец, при стремлении времени к бесконечности график выходит на плато, стремясь к своему стационарному значению. Эта стабилизация объясняется достижением динамического равновесия между процессами возникновения отказов и процессами восстановления в полумарковской модели. Данное стационарное значение является ключевым индикатором долгосрочной надежности, определяющим эффективность системы в реальных условиях эксплуатации.

Для практического применения модели в конкретных реальных системах параметры распределения Вейбулла должны определяться на основе статистических данных мониторинга (например, из открытых наборов данных, таких как Google Cluster Data или Alibaba Cluster Data). Наиболее достоверным методом оценки является метод максимального правдоподобия [45]. Если имеется выборка времен до отказа $\tau_1, \tau_2, \dots, \tau_m$ – функция правдоподобия для распределения Вейбулла имеет вид:

$$F(\eta, \beta) = \prod_{i=1}^m \frac{\beta}{\eta} \left(\frac{\tau_i}{\eta} \right)^{\beta-1} e^{-\left(\frac{\tau_i}{\eta} \right)^{\beta}}.$$

Процедура нахождения точечных оценок параметров η и β для конкретного типа контейнера или общего ПО заключается в максимизации логарифмической функции правдоподобия. Для двухпараметрического распределения Вейбулла логарифм функции правдоподобия имеет вид:

$$\ln F(\eta, \beta) = m \ln \beta - m \beta \ln \eta + (\beta - 1) \sum_{i=1}^m \ln \tau_i - \sum_{i=1}^m \left(\frac{\tau_i}{\eta} \right)^{\beta}.$$

Приравнявая к нулю частные производные, получают систему уравнений метода максимального правдоподобия:

$$\begin{cases} \frac{\partial \ln F(\eta, \beta)}{\partial \beta} = \frac{m}{\beta} - m \ln \eta + \sum_{i=1}^m \ln \tau_i - \sum_{i=1}^m \left(\frac{\tau_i}{\eta} \right)^\beta \ln \left(\frac{\tau_i}{\eta} \right) = 0, \\ \frac{\partial \ln F(\eta, \beta)}{\partial \eta} = -\frac{m\beta}{\eta} + \frac{\beta}{\eta^{\beta+1}} \sum_{i=1}^m \tau_i^\beta = 0, \end{cases}$$

$$\Leftrightarrow \begin{cases} \frac{\sum_{i=1}^m \tau_i^\beta \ln \tau_i}{\sum_{i=1}^m \tau_i^\beta} - \frac{1}{\beta} = \frac{1}{m} \sum_{i=1}^m \ln \tau_i, \\ \eta = \left(\frac{1}{m} \sum_{i=1}^m \tau_i^\beta \right)^{\frac{1}{\beta}}. \end{cases}$$

Моделирование подтвердило, что в условиях естественного старения и при отсутствии профилактических мер коэффициент готовности системы неуклонно снижается, особенно при высоких значениях параметра формы распределения Вейбулла β . Это свидетельствует о быстром накоплении критических ошибок и утечек ресурсов. Установлено, что общая надежность инфраструктуры в значительной степени лимитируется временем безотказной работы общего ПО, отказ которого приводит к полной потере работоспособности всех запущенных контейнеров. Выявленная динамика деградации показателей обосновывает необходимость перехода от пассивного ожидания отказов к разработке стратегий проактивного восстановления, которые будут подробно рассмотрены в последующих разделах работы.

3. Механизмы противодействия старению ПО на основе перезапуска контейнеров. Для повышения рассчитанного ранее коэффициента готовности и минимизации негативного влияния программного старения необходимо внедрение механизмов проактивного восстановления. Одной из наиболее эффективных стратегий является проактивная перезагрузка и обновление контейнера. Суть данной стратегии заключается в плановом превентивном перезапуске контейнера до момента наступления критического отказа. Такой подход позволяет сбросить накопленные в процессе эксплуатации ошибки, устранить утечки памяти и высвободить системные ресурсы, возвращая программную среду в исходное чистое состояние. На рисунке 3 приведена полумарковская модель состояний системы при реализации предложенной стратегии

проактивного восстановления. В отличие от базовой модели, здесь учитывается переход в состояние активного восстановления до момента возникновения отказа.

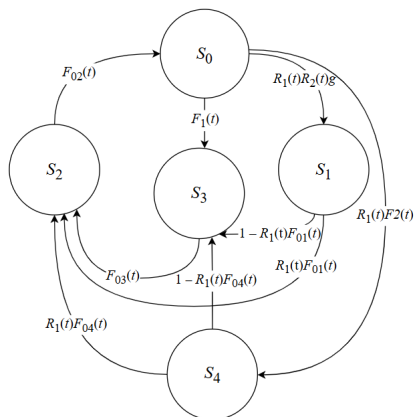


Рис. 3. Диаграмма состояний и переходов системы с проактивным восстановлением

В предложенной модели выделяются следующие состояния системы: S_0 – работоспособность при возможности нормального функционирования общего ПО и контейнера; S_1 – проактивный перезапуск контейнера; S_2 – миграция контейнера; S_3 – восстановление контейнера после отказа, S_4 – отказ системного (общего) ПО. При этом процедуры проактивного обновления и восстановления контейнера реализуются только при условии работоспособности общего ПО. В случае отказа общего ПО (состояние S_4) требуется восстановление системной среды, перезапуск контейнера и его последующая миграция. Процессы восстановления в состояниях S_1 , S_2 , S_3 и S_4 описываются экспоненциальными законами распределения $F_{01}(t)$, $F_{02}(t)$, $F_{03}(t)$ и $F_{04}(t)$ соответственно.

Для описания стратегии проактивного перезапуска контейнера вводится управляемый параметр $g \in [0, 1]$, определяющий вероятность принятия решения о проведении проактивного восстановления при достижении детерминированного интервала проактивного перезапуска контейнера t_0 . Параметр g представляет собой управляемую характеристику стратегии проактивного перезапуска контейнера и определяет вероятность того, что при достижении наработки $t = t_0$ система перейдет в состояние проактивного восстановления S_1 , а не продолжит работу в состоянии S_0 .

Параметр g определяет режим функционирования системы. При $g = 0$ реализуется полностью реактивная стратегия восстановления, при которой переход в состояния восстановления S_3 или S_4 осуществляется исключительно по факту отказа общего ПО или контейнера соответственно. При $g = 1$ система гарантированно переходит в состояние проактивного восстановления S_1 при достижении интервала проактивного перезапуска контейнера t_0 . При промежуточных значениях $0 < g < 1$ реализуется смешанная стратегия проактивного перезапуска контейнера, при которой с вероятностью g осуществляется проактивное восстановление в момент $t = t_0$, а с вероятностью $(1 - g)$ система продолжает эксплуатацию до наступления отказа. На основе представленной (Рис. 3) диаграммы состояний и переходов матрица вероятностей переходов полумарковского процесса $M_2(t, g)$ записывается в виде:

$$M_2(t, g) = \begin{pmatrix} R_1(t)R_2(t)(1-g) & R_1(t)R_2(t)g & 0 & F_1(t) & R_1(t)F_2(t) \\ 0 & 0 & R_1(t)F_{01}(t) & 1-R_1(t)F_{01}(t) & 0 \\ F_{02}(t) & 0 & 1-F_{02}(t) & 0 & 0 \\ 0 & 0 & F_{03}(t) & 1-F_{03}(t) & 0 \\ 0 & 0 & R_1(t)F_{04}(t) & 1-R_1(t)F_{04}(t) & 0 \end{pmatrix}.$$

Произведение вероятностей $R_1(t)R_2(t)$ определяет вероятность безотказной работы системы к моменту наработки t , т.е. вероятность того, что ни общее ПО, ни контейнер не отказали в течение интервала $[0, t]$. При достижении детерминированного интервала проактивного перезапуска контейнера $t = t_0$ и выполнении условия безотказной работы $R_1(t_0)R_2(t_0)$ с вероятностью g принимается решение о проведении проактивного перезапуска контейнера (переход в состояние S_1), а с вероятностью $(1 - g)$ система продолжает функционировать в работоспособном состоянии S_0 . Стационарное распределение вероятностей $v(t, g) = (v_0(t, g), v_1(t, g), v_2(t, g), v_3(t, g), v_4(t, g))$ находится из решения системы линейных уравнений:

$$\begin{cases} v(t, g) = v(t, g)M_2(t, g), \\ \sum_{i=0}^4 v_i(t, g) = 1. \end{cases}$$

Решение данной системы уравнений позволило получить динамику компонентов вектора $v(t, g)$ в зависимости от наработки

и параметра g . На рисунке 4(а) представлена динамика составляющих вектора $v(t, g)$ в зависимости от наработки при фиксированном значении $g = 0,5$, а также на рисунке 4(б) приведено семейство кривых для компонента $v_3(t, g)$, отражающего вероятность пребывания системы в состоянии отказа S_3 , при различных значениях g .

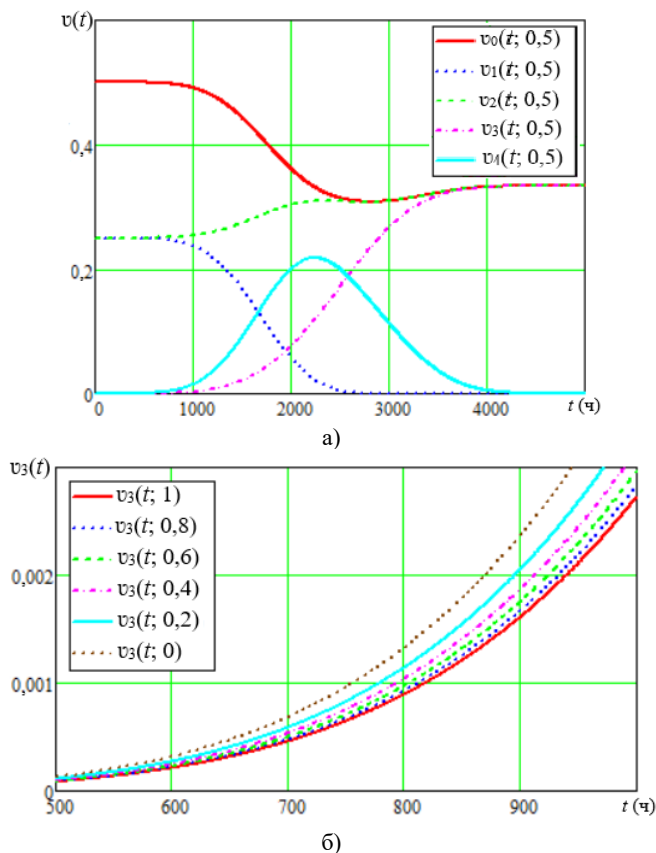


Рис. 4. Результаты компонентов вектора вероятностей состояний $v(t)$ при $g = 0,5$ и семейство кривых $v_3(t)$ при различных значениях вероятности проактивного восстановления g : а) компоненты вектора $v(t)$ при $g = 0,5$; б) семейство кривых $v_3(t)$ при различных значениях g

Анализ полученных зависимостей показывает, что увеличение вероятности проактивного восстановления g приводит к сокращению среднего времени пребывания системы в состоянии отказа общего ПО.

Физический смысл данного явления заключается в том, что реализация стратегии омоложения позволяет превентивно устранять накопленные программные дефекты и утечки ресурсов до момента их перехода в критическую стадию. Таким образом, при максимальной вероятности реализации стратегии ($g = 1$) система чаще переходит в режим контролируемого планового проактивного восстановления, что минимизирует длительность нахождения в состояниях реактивного восстановления. Это подтверждает, что активное управление жизненным циклом системы эффективно подавляет негативные последствия программного старения, предотвращая длительные простои инфраструктуры.

С учетом найденного распределения вероятностей $v(t, g)$ итоговая формула для коэффициента готовности принимает вид:

$$K_2(t_0, g) = \frac{\int_0^{t_0} e^{-\left(\frac{t}{\eta_1}\right)^{\beta_1}} e^{-\left(\frac{t}{\eta_2}\right)^{\beta_2}} dt}{\int_0^{t_0} e^{-\left(\frac{t}{\eta_1}\right)^{\beta_1}} e^{-\left(\frac{t}{\eta_2}\right)^{\beta_2}} dt + v_1(t_0, g)t_1 + v_2(t_0, g)t_2 + v_3(t_0, g)t_3 + v_4(t_0, g)t_4}. \quad (1)$$

Для проведения численного анализа и верификации модели были использованы следующие исходные данные: время перезагрузки контейнера $t_1 = 1/30$ ч, время миграции контейнера $t_2 = 1/120$ ч, время восстановления общего ПО $t_3 = 1$ ч и время восстановления контейнера $t_4 = 0,25$ ч. Значения остальных параметров приняты идентичными использованным выше. Графические зависимости коэффициента готовности от интервала проактивного перезапуска контейнера t_0 при различных значениях g приведены на рисунке 5.

Для оценки устойчивости модели был проведен анализ чувствительности коэффициента готовности к вариации параметров распределения Вейбулла. Результаты расчетов представлены на рисунке 6. Увеличение параметра формы $\beta_1 = \beta_2 = \beta$ (кривые 2, 1, 3 соответствуют $\beta = 3; 5; 7$) при $\eta_1 = 3000$ ч и $\eta_2 = 2000$ ч приводит к росту максимального значения коэффициента готовности и смещению оптимального интервала проактивного восстановления t_0 вправо. Это объясняется тем, что при высоких значениях β программное старение проявляется позже, но более интенсивно: система дольше остается стабильной в начале цикла, что позволяет отложить перезапуск контейнера без риска отказа. Чем выше β , тем дольше система сохраняет высокую надежность в начале цикла,

что позволяет увеличить интервал между перезапусками контейнера без риска возникновения отказа. При фиксированном значении $\beta_1 = \beta_2 = \beta = 5$ изменение параметра масштаба η (кривые 4 и 5) демонстрирует практически прямо пропорциональную зависимость. Снижение показателей надежности ($\eta_1 = 2400$ ч; $\eta_2 = 1600$ ч – кривая 4) сокращает безопасный период работы ПО, что требует более раннего вмешательства и смещает t_0 влево. Напротив, при увеличении запаса стабильности ПО ($\eta_1 = 3600$ ч; $\eta_2 = 2400$ ч – кривая 5) время до наступления фазы старения увеличивается, вследствие чего оптимальная точка t_0 смещается вправо.

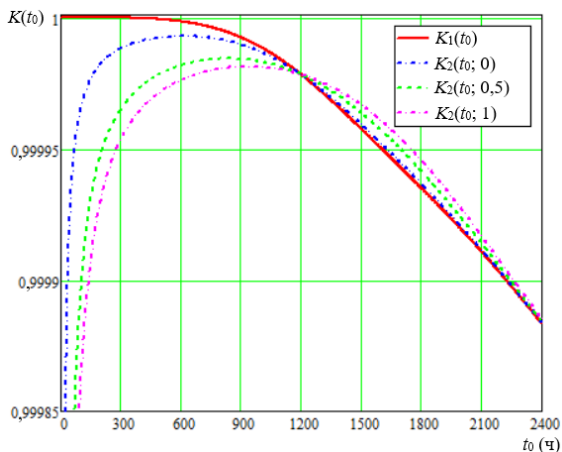


Рис. 5. Зависимость коэффициента готовности от интервала проактивного перезапуска контейнера t_0 при различных значениях g

В результате исследования установлено, что интервал проактивного перезапуска контейнера t_0 является ключевым фактором управления надежностью системы. Сравнительный анализ показал, что реализация стратегии проактивного восстановления в диапазоне от 1200 до 2400 часов обеспечивает существенный прирост коэффициента готовности, достигая максимума при полной вероятности проактивного восстановления ($g = 1$). Выявлено, что чрезмерно частое проведение проактивного восстановления ведет к неоправданному простоям и снижению производительности. В то же время увеличение интервала t_0 нивелирует эффект профилактики, переводя систему в режим реактивного восстановления после критических отказов. Полученные результаты позволяют

оптимизировать стратегию проактивного перезапуска контейнера, обеспечивая своевременное устранение накопленных ошибок, высвобождение избыточных ресурсов и предотвращение критических отказов, требующих значительных временных и финансовых затрат на восстановление.

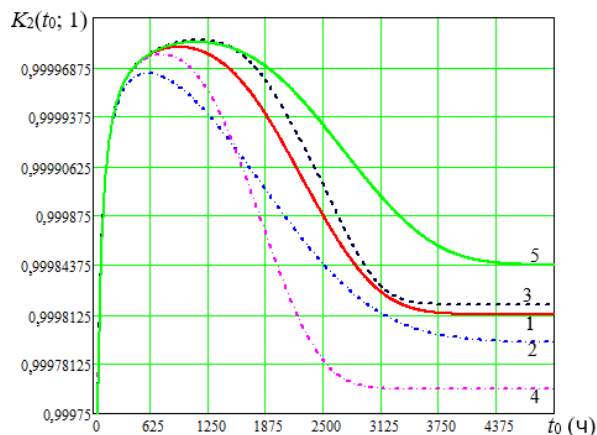


Рис. 6. Анализ чувствительности коэффициента готовности к изменению параметров распределения Вейбулла (η и β)

4. Упрощенная модель. Использование предложенной полумарковской модели для оптимизации в реальном времени может быть затруднено из-за её высокой вычислительной сложности. В связи с этим представлена упрощенная математическая модель процесса обслуживания ПО в компьютерной системе с контейнерной виртуализацией. Переход к упрощенной форме обоснован тем, что процессы восстановления системного ПО и контейнеров описываются экспоненциальными законами распределения, при которых вероятность завершения восстановления быстро стремится к единице.

Поскольку в реальных сценариях оптимальный интервал проактивного перезапуска контейнера t_0 значительно превышает время восстановления, вероятность возникновения вторичного отказа непосредственно в моменты перезапуска или миграции становится пренебрежимо малой. Это допущение позволяет существенно снизить вычислительную сложность модели, сохраняя при этом её высокую точность и физическую адекватность.

В рамках упрощенной модели рассматривается случай полной реализации стратегии проактивного восстановления ($g = 1$). В этом случае матрица вероятностей переходов полумарковского процесса $M_3(t)$ принимает следующий вид:

$$M_3(t) = \begin{pmatrix} 0 & R_1(t)R_2(t) & 0 & F_1(t) & R_1(t)F_2(t) \\ 0 & 0 & R_1(t) & 1-R_1(t) & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & R_1(t) & 1-R_1(t) & 0 \end{pmatrix}.$$

На основе матрицы $M_3(t)$ путем решения соответствующей системы линейных алгебраических уравнений выведены аналитические выражения для компонентов вектора вероятностей состояний $v(t)$ в явном виде.

$$v_0(t) = v_2(t) = \frac{1}{3 + R_1(t) - R_1^2(t)}; v_1(t) = \frac{R_1(t)R_2(t)}{3 + R_1(t) - R_1^2(t)};$$

$$v_3(t) = \frac{1 - R_1^2(t)}{3 + R_1(t) - R_1^2(t)}; v_4(t) = \frac{R_1(t) - R_1(t)R_2(t)}{3 + R_1(t) - R_1^2(t)}.$$

Это позволяет выполнять быстрые расчеты показателей надежности без необходимости проведения итерационных вычислений. Расчет коэффициента готовности произведен по формуле (1). Для подтверждения допустимости предложенных упрощений проведена серия сравнительных расчетов в программной среде MathCad. Основной целью верификации являлось определение относительной погрешности коэффициента готовности, возникающей при использовании упрощенной модели по сравнению со строгой полумарковской моделью. Результат сравнительного анализа представлен на рисунке 7.

Проведенный сравнительный анализ показал, что результаты, полученные с помощью упрощенной модели, имеют пренебрежимо малую погрешность по сравнению со строгой полумарковской моделью. Упрощенная модель подтверждает гипотезу о том, что при оптимальных интервалах проактивного перезапуска контейнера t_0 , значительно превышающих время восстановления, экспоненциальный характер процессов позволяет пренебречь

вероятностью возникновения отказов в фазах перезапуска и миграции. Таким образом, разработанная упрощенная модель эффективна для решения задач оперативной оптимизации стратегий проактивного восстановления. Она обеспечивает необходимый баланс между математической строгостью и вычислительной эффективностью. Полученные аналитические выражения позволяют интегрировать данную модель в системы автоматизированного управления облачными ресурсами для динамической настройки параметров обслуживания в зависимости от текущей интенсивности старения программного обеспечения.

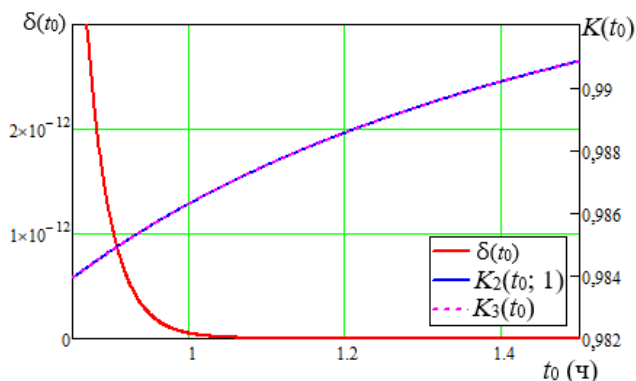


Рис. 7. Относительная погрешность коэффициента готовности $\delta(t_0)$ упрощенной модели

5. Моделирование надежности компьютерной системы с n контейнерами. На данном этапе исследования осуществляется логическое расширение разработанной ранее математической модели на случай сложной кластерной инфраструктуры. Рассматриваемая система состоит из общего системного ПО и n изолированных функциональных контейнеров. Будем считать, что для сохранения работоспособности всей системы в целом достаточно функционирования хотя бы одного из n контейнеров. Исходя из этого, вероятность работоспособного состояния кластерной системы рассчитывается как произведение надежности общего ПО на вероятность того, что хотя бы один из n контейнеров находится в рабочем состоянии:

$$L(t, n) = R_1(t) \left[1 - \prod_{i=1}^n (1 - R_{c,i}(t)) \right],$$

где $R_1(t)$ – функция вероятности безотказной работы общего ПО, а $R_{c,i}(t)$ – функция вероятности безотказной работы отдельных контейнеров. Введение множителя $R_1(t)$ в данную формулу является критически важным, так как отказ хост-платформы является точкой единого отказа, приводящей к неработоспособности всего кластера независимо от количества резервных контейнеров.

Для упрощения математической формализации и последующего анализа кластерной архитектуры в данной работе принимается допущение, что все n контейнеров являются идентичными и характеризуются одинаковой функцией надежности $R_{c,i}(t) = R_2(t)$ для всех $i = 1, \dots, n$. Такое допущение обосновано практикой развертывания современных облачных микросервисов, где контейнеры создаются на основе единого эталонного образа и функционируют в однородной программной среде. В этом случае вероятность работоспособного состояния контейнеров определяется через биномиальное распределение, и общая формула вероятности функционирования системы принимает более компактный вид:

$$L(t, n) = R_1(t) \left[1 - (1 - R_2(t))^n \right].$$

Использование идентичных функций $R_2(t)$ позволяет сфокусировать исследование на изучении влияния масштабируемости (числа развернутых контейнеров n) и стратегий проактивного восстановления на итоговый коэффициент готовности, исключив влияние стохастической неоднородности отдельных узлов. Это обеспечивает высокую прозрачность моделирования и позволяет получить аналитические зависимости для определения оптимальных параметров технического обслуживания всей кластерной группы как единого функционального объекта. Зависимость вероятности работоспособного состояния кластерной системы $L(t, n)$ от наработки при различном числе развернутых контейнеров приведена на рисунке 8.

Рост числа контейнеров n в кластере приводит к существенному увеличению вероятности работоспособного состояния системы, так как для возникновения полного отказа требуется одновременный отказ всех параллельно соединенных контейнеров. Возведение вероятности отказа отдельного контейнера $(1 - R_2(t))$ в степень n приводит к быстрому стремлению этого значения к нулю, тем самым

максимизируя итоговый показатель $L(t, n)$. Это смещает область высокой готовности вправо по оси времени, обеспечивая системе дополнительный запас надежности даже при высокой интенсивности программного старения отдельных компонентов.

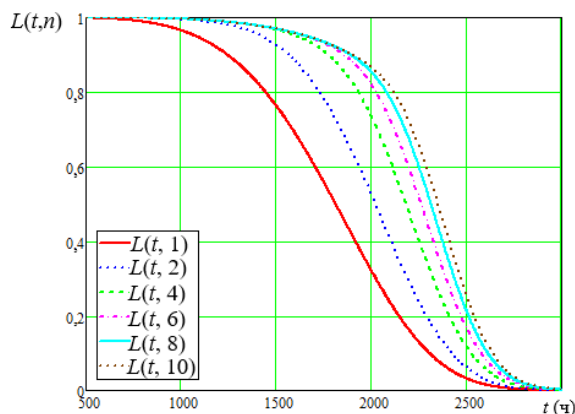


Рис. 8. Зависимость вероятности работоспособного состояния кластерной системы от наработки при различном числе развернутых контейнеров

На основе упрощенной модели построена диаграмма состояний и переходов кластерной системы, представленная на рисунке 9. Приняты следующие обозначения состояний системы: S_0 – работоспособность системы; S_1 – проактивный перезапуск и миграция контейнеров; S_2 – отказ системы.

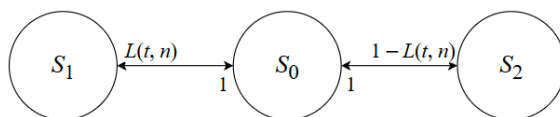


Рис. 9. Диаграмма состояний и переходов упрощенной полумарковской модели системы

Основой для расчета всех показателей служит матрица вероятностей переходов $M_4(t, n)$, которая описывает логику взаимодействия между работоспособным состоянием, состоянием проактивного восстановления и состоянием полного отказа. Матрица переходов имеет вид:

$$M_4(t, n) = \begin{pmatrix} 0 & L(t, n) & 1 - L(t, n) \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

Важно подчеркнуть, что процесс программного старения рассматривается как непрерывная деградация ресурсов, происходящая непосредственно в состоянии работоспособности S_0 . Переход системы в состояние проактивного восстановления S_1 осуществляется исключительно в детерминированный момент времени $t = t_0$. Таким образом, состояние S_1 интерпретируется не как процесс старения, а как активная управляющая процедура, направленная на устранение последствий деградации. Вероятность $L(t_0, n)$ в данном контексте определяет долю успешно доживших до момента проактивного восстановления. Такое разделение позволяет четко разграничить естественный износ системы в S_0 и искусственно инициируемый цикл её омоложения в S_1 , что исключает неопределенность при расчете оптимального интервала проактивного перезапуска контейнеров.

На основе данной матрицы вычисление компонентов вектора $v(t, n)$ проводится аналогично описанному выше подходу. В результате получены следующие выражения для компонентов вектора $v(t, n)$:

$$v_0(t, n) = \frac{1}{2}; v_1(t, n) = \frac{L(t, n)}{2}; v_2(t, n) = \frac{1 - L(t, n)}{2}.$$

Для кластерной системы, включающей общее ПО и n изолированных контейнеров, коэффициент готовности определяется следующим выражением:

$$K_4(t_0, n) = \frac{\int_0^{t_0} R_1(t) \left[1 - (1 - R_2(t))^n \right] dt}{\int_0^{t_0} R_1(t) \left[1 - (1 - R_2(t))^n \right] dt + v_1(t_0, n)(t_1 + nt_2) + v_2(t_0, n)(t_3 + nt_2)},$$

где t_0 – интервал проактивного перезапуска контейнеров, t_1 – время перезагрузки контейнеров, t_2 – время миграции контейнеров, t_3 – время восстановления общего ПО.

В связи с этим возникает задача определения оптимального числа контейнеров и интервала проактивного перезапуска контейнеров t_0 для достижения максимального коэффициента готовности системы. Для решения данной задачи на листинге 1 представлена программа, написанная в среде системы компьютерной математики Mathcad.

```

Max(n, m){
    t0 := 1;
    n0 := 1;
    Amax := K4(t0, n0);
    for(x ∈ 1,2,...,m){
        for(y ∈ 1,2,...,n){
            if(G2(x,y) > Amax){
                Amax := K4(x, y);
                t0 := x;
                n0 := y;
            }
        }
    }
    Return (t0, n0, Amax);
}

```

Листинг 1. Оптимизация коэффициента готовности системы по параметрам n и t_0

Реализация численного алгоритма в среде Mathcad позволила получить аналитическое решение задачи оптимизации и визуализировать поведение целевой функции. На рисунке 10 представлена трехмерная поверхность зависимости коэффициента готовности кластерной системы от варьируемых параметров: интервала проактивного перезапуска контейнеров и количества контейнеров.

Визуальный анализ данной поверхности позволяет наглядно определить область глобального максимума и оценить чувствительность системы к изменениям каждого из параметров. Характер представленной зависимости подтверждает существование единственной точки оптимума, обеспечивающей наилучший баланс между отказоустойчивостью инфраструктуры и минимизацией временных затрат на техническое обслуживание. В результате проведенных расчетов установлено, что оптимальное количество контейнеров составляет $n = 4$, а оптимальный интервал $t_0 = 1659$ ч.

Увеличение числа контейнеров в кластере способствует повышению общей отказоустойчивости системы. Однако чрезмерное увеличение числа контейнеров имеет и негативную сторону: возрастает время задержки при выполнении процедур миграции

и перезапуска, а также увеличиваются накладные расходы на управление инфраструктурой. Найденный в ходе исследования экстремум функции демонстрирует баланс между этими двумя факторами. Результаты позволяют определить оптимальное количество контейнеров и наилучший момент времени для проведения профилактики. Таким образом, разработанная модель, учитывающая как старение прикладных контейнеров, так и надежность базовой ОС, представляет собой комплексный инструмент для проектирования высоконадежных облачных сред. Полученные результаты создают теоретическую и практическую базу для построения отказоустойчивых контейнеризированных систем, обеспечивая их стабильное функционирование и эффективное управление ресурсами в условиях отказов ПО. В качестве дальнейшего развития предложенных моделей предполагается модернизация для распределенных кластерных систем при перераспределении нагрузки и миграции виртуальных контейнеров через сеть [50, 51]. Такая модернизация должна учитывать реконфигурацию и накопление отказов узлов многопутевой сети [50, 51], в том числе при организации граничных, туманных и облачных вычислений [13, 14]. Также предполагается развитие работ в части постановки и решения задачи векторной оптимизации [52, 53] с учетом надежности, производительности, стоимости реализации, эксплуатационных расходов и энергопотребления.

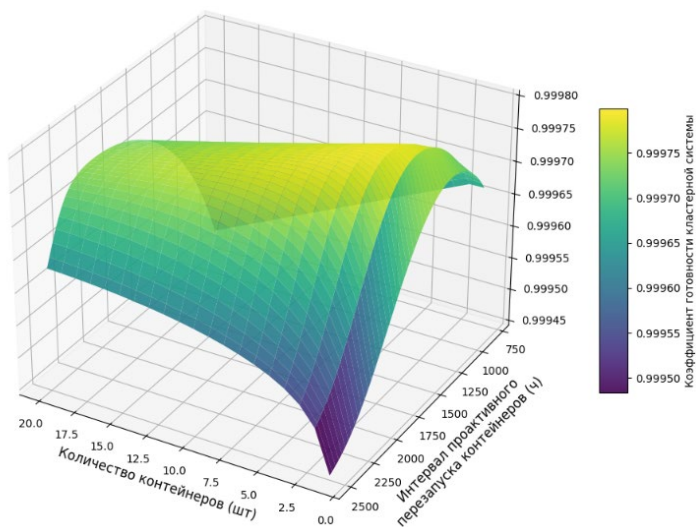


Рис. 10. Зависимость коэффициента готовности кластера от параметров n и t_0

6. Заключение. На основе полумарковских процессов разработана, верифицирована и оптимизирована комплексная математическая модель надежности программного обеспечения в компьютерной системе с контейнерной виртуализацией, позволяющая с высокой точностью прогнозировать периоды деградации программных ресурсов и обосновать необходимость внедрения стратегий проактивного восстановления.

Показано, что стратегии проактивного восстановления являются эффективным инструментом борьбы со старением ПО. Реализация стратегии проактивного восстановления в диапазоне от 1200 до 2400 часов обеспечивает существенный прирост коэффициента готовности (рост на 0,16%) и достигает максимума при полной вероятности проактивного восстановления ($g = 1$).

Разработана упрощенная полумарковская модель, характеризующаяся пренебрежимо малой погрешностью при снижении вычислительной сложности, что позволяет рассматривать её как эффективный инструмент для оперативных инженерных расчетов показателей готовности в реальных эксплуатационных условиях.

Представлена расширенная модель кластера, учитывающая надежность общего системного ПО и ПО, поддерживающего функционирование контейнеров.

Показано существование экстремума функции коэффициента готовности в зависимости от числа развернутых в системе контейнеров и периодичности их проактивного перезапуска. В результате проведенных расчетов установлено, что оптимальное количество контейнеров составляет $n = 4$, а оптимальный интервал $t_0 = 1659$ ч.

В отличие от модели Джелинского-Моранды [34], предложенный подход описывает возрастающую интенсивность отказов через распределение Вейбулла, что адекватно отражает накопление ресурсных ошибок в долгоживущих контейнерах. По сравнению с монолитными моделями Мусы-Шумана [36] и Миллса [38], модель явно разделяет деградацию общего ПО и контейнеров, выявляя отказ общего ПО как доминирующий фактор потери готовности. В отличие от подхода [48], рассматривающего совместный анализ производительности и готовности в рамках обобщённых марковских моделей без учёта специфики программного старения, предложенная модель явно включает механизм деградации через распределение Вейбулла и аналитически решает задачу совместной оптимизации числа контейнеров n и интервала проактивного перезапуска t_0 . По сравнению с работой [47], являющейся методологическим прототипом применения аппарата SMP к оценке надёжности ПО, предложенная модель заменяет

экспоненциальное распределение времени до отказа (постоянная интенсивность) на распределение Вейбулла ($\beta > 1$), разделяет иерархические уровни общего ПО хоста и изолированных контейнеров. При этом дополняет реактивное восстановление механизмом проактивного омоложения с управляемым параметром g , а также решает задачу совместной оптимизации конфигурационных параметров кластера – тогда как работа [47] ориентирована на монолитное ПО с реактивным восстановлением и статичной архитектурой.

Направления дальнейших исследований включают расширение модели для учета неоднородности нагрузки на различные узлы кластера, а также разработку программных агентов для мониторинга ресурсов в реальном времени с целью динамической корректировки интервалов проактивного перезапуска контейнера t_0 . Предложенные решения могут быть внедрены в системы управления облачными дата-центрами для повышения их отказоустойчивости и автоматизации процессов технического обслуживания.

Литература

1. Dong H., Zhang C., Li G., Zhang H. Cloud-Native Databases: A Survey // IEEE Transactions on Knowledge and Data Engineering. vol. 36. no. 12. pp. 7772–7791. 2024. DOI: 10.1109/TKDE.2024.3397508.
2. Goyal P., Deora S.S. Reliability of Trust Management Systems in Cloud Computing // Indian Journal of Cryptography and Network Security. 2022. vol. 2. no. 1. pp. 1–5. DOI: 10.54105/ijcns.C1417.051322.
3. Assiri B., Sheneamer A. Fault tolerance in distributed systems using deep learning approaches // PLOS One. 2025. vol. 20. no. 1. DOI: 10.1371/journal.pone.0310657.
4. Херсонский Н.С., Большедворская Л.Г. Анализ развития статистических методов оценки надежности как свойства качества объектов и продукции // Crede Experto: транспорт, общество, образование, язык. 2024. №1. С. 6–23. DOI: 10.51955/2312-1327_2024_1_6.
5. Gurjanov A.V., Korobeynikov A.G., Zharinov I.O., Zharinov O.O. Edge, fog and cloud computing in the cyber-physical systems networks // Proceedings of the III International Workshop on Modeling, Information Processing and Computing. 2021. pp. 103–108. DOI: 10.47813/dnit-mip3/2021-2899-103-108
6. Kushchazli A., Safargalieva A., Kochetkova I., Gorshenin A. Queuing Model with Customer Class Movement across Server Groups for Analyzing Virtual Machine Migration in Cloud Computing // Mathematics. 2024. vol. 12. no. 3. 468 p. DOI: 10.3390/math12030468.
7. Damarapati A. Containers vs. Virtual Machines: Understanding the Shift to Kubernetes // World Journal of Advanced Engineering Technology and Sciences. 2025. vol. 15. no. 1. pp. 852–861. DOI: 10.30574/wjaets.2025.15.1.0305.
8. Kozhimbayev Z., Sinnott R.O. A performance comparison of container-based technologies for the Cloud // Future Generation Computer Systems. 2017. vol 68. pp. 175–182. DOI: 10.1016/j.future.2016.08.025.
9. Mirna A., Nadja K., Aris L. Utilization prediction-based VM consolidation approach // Journal of Parallel and Distributed Computing. 2022. vol. 170. pp. 24–38. DOI: 10.1016/j.jpdc.2022.08.001.

10. Половко А.М., Гуров С.В. Основы теории надежности // Санкт-Петербург: БХВ-Петербург, 2008. 702 с.
11. Tatamikova T.M., Arkhiptsev E.M. Designing Fault-tolerant Systems with Micro-service Architecture // Proceedings of the 27-th International Conference on Soft Computing and Measurements (SCM). 2024. pp. 348–351. DOI: 10.1109/SCM62608.2024.10554143.
12. Koren I., Krishna S.M. Fault-Tolerant Systems // Boston: Morgan Kaufmann. 2007. 400 p.
13. Yermakov S.G., Khalil M.M., Khomonenko A.D., Bukharova K.A. Evaluating the efficiency of fog computing on the Internet of Things using a Non-Markov model // T-Comm. 2022. vol. 16. no. 12. pp. 46–52.
14. Хомоненко А.Д., Кириенко А.Б., Пугачев С.В. Оценивание оперативности туманных вычислений в геоинформационных системах // Инженерный вестник Дона. 2024. №6. С. 364–382.
15. Хомоненко А.Д., Благовещенская Е.А., Проурзин О.В., Андрук А.А. Прогноз надежности кластерной вычислительной системы с помощью полумарковской модели альтернирующих процессов и мониторинга // Наукоемкие технологии в космических исследованиях Земли. 2018. Т. 10. №4. С. 72–82. DOI: 10.24411/2409-5419-2018-10099.
16. Богатырев В.А., Богатырев С.В., Богатырев А.В. Оценка готовности компьютерной системы к своевременному обслуживанию запросов при его совмещении с информационным восстановлением памяти после отказов // Научно-технический вестник информационных технологий, механики и оптики. 2023. Т. 23. №3. С. 608–617. DOI: 10.17586/2226-1494-2023-23-3-608-617.
17. Bogatyrev V.A., Bogatyrev S.V., Bogatyrev A.V. Recovery of Real-Time Clusters with the Division of Computing Resources into the Execution of Functional Queries and the Restoration of Data Generated Since the Last Backup // Lecture Notes in Computer Science. Springer. 2024. vol. 14123. pp. 236–250. DOI: 10.1007/978-3-031-50482-2_19.
18. Bogatyrev V.A., Vinokurova M.S. Control and Safety of Operation of Duplicated Computer Systems // Communications in Computer and Information Science. Springer. 2017. vol. 700. pp. 331–342. DOI: 10.1007/978-3-319-66836-9_28.
19. Алексанков С.М. Модели динамической миграции с итеративным подходом и сетевой миграции виртуальных машин // Научно-технический вестник информационных технологий, механики и оптики. 2015. Т. 15. №6. С. 1098–1104. DOI: 10.17586/2226-1494-2015-15-6-1098-1104.
20. Bogatyrev V.A., Derkach A.N. Evaluation of a Cyber-Physical Computing System with Migration of Virtual Machines During Continuous Computing // Computers. 2020. vol. 9. no 2. pp. 42. DOI: 10.3390/computers9020042.
21. Богатырев В.А., Фунг В.К. Оценка надежности восстанавливаемого кластера контейнерной виртуализацией // Научно-технический вестник информационных технологий, механики и оптики. 2025. Т. 25. №5. С. 988–995. DOI: 10.17586/2226-1494-2025-25-5-988-995.
22. Богатырев В.А., Богатырев С.В., Богатырев А.В. Граничная оценка надежности кластерных систем на основе декомпозиции марковской модели при ограниченном восстановлении узлов с накоплением отказов // Научно-технический вестник информационных технологий, механики и оптики. 2025. Т. 25. №3. С. 574–583. DOI: 10.17586/2226-1494-2025-25-3-574-583.
23. Rodriguez M.A., Buuya R. Container-based cluster orchestration systems: A taxonomy and future directions // Software: Practice and Experience. 2019. vol. 49. pp. 698–719. DOI: 10.1002/spe.2660.
24. Application Container Market Size And Forecast // Verified Market Research. URL: www.verifiedmarketresearch.com/product/application-container-market. (дата обращения: 16.06.2026).

25. Application Container Market (2025-2023) // Grand View Research. URL: www.grandviewresearch.com/industry-analysis/application-container-market. (дата обращения: 16.06.2026).
26. Marella V. Comparative Analysis of Container Orchestration Platforms: Kubernetes vs. Docker Swarm // *International Journal of Scientific Research in Science and Technology*. 2024. vol. 11. no. 5. pp. 526–543. DOI: 10.32628/IJSRST24105254.
27. Grottko M., Trivedi K.S. Fighting bugs: remove, retry, replicate, and rejuvenate // *Computer*. 2007. vol. 40. no. 2. pp. 107–109. DOI: 10.1109/MC.2007.55.
28. Cotroneo D., Natella R., Pietrantuono R., Russo S. Software Aging Analysis of the Linux Operating System // *Proceedings of the IEEE 21st International Symposium on Software Reliability Engineering (ISSRE)*. 2010. pp. 71–80. DOI: 10.1109/ISSRE.2010.24.
29. Nascimento M.G., Moura R.J., Machida F., Andrade E. Comparison of Machine Learning Algorithms for Detecting Software Aging in SQL Server // *Proceedings of the 13th Latin-American Symposium on Dependable and Secure Computing*. 2024. pp. 159–167. DOI: 10.1145/3697090.3699798.
30. Oliveira F., Araujo J., Matos R., Maciel P. Software Aging in Container-Based Virtualization: An Experimental Analysis // *Proceedings of the 16th Iberian Conference on Information Systems and Technologies (CISTI)*. 2021. pp. 1–7. DOI: 10.23919/CISTI52073.2021.9476625.
31. Nguyen T.A., Min D., Choi E., Tran T.D. Reliability and Availability Evaluation for Cloud Data Center Networks Using Hierarchical Models // *IEEE Access*. 2019. vol. 7. pp. 9273–9313. DOI: 10.1109/ACCESS.2019.2891282.
32. Shruthi P., Nagaraj G.C. Software aging prediction – a new approach // *International Journal of Electrical and Computer Engineering*. 2023. vol. 13. no. 2. pp. 1773–1781. DOI: 10.11591/ijece.v13i2.pp1773-1781.
33. Pietrantuono R., Russo S. A survey on software aging and rejuvenation in the cloud // *Software Quality Journal*. 2020. vol. 28. pp. 7–38. DOI: 10.1007/s11219-019-09448-3.
34. Turk L.I.A., Alsolami E.G. Jelinski-Moranda Software Reliability Growth Model: A Brief Literature and Modification // *International Journal of Software Engineering & Applications (IJSEA)*. 2016. vol. 7. no. 2. pp. 33–44. DOI: 10.5121/ijsea.2016.7204.
35. Li N., Tan Y., Wang X., Li B., Luo J. Dynamic Resource Allocation for Containerized Applications in Edge Computing // *Lecture Notes in Electrical Engineering*. Springer. 2023. vol. 1015. pp. 121–130. DOI: 10.1007/978-981-99-0416-7_12.
36. Musa J.D., Iannino A., Okumoto K. Software Reliability: Measurement, Prediction, Application // New York: McGraw-Hill. 1987. 621 p.
37. Dohi T., Goseva-Popstojanova K., Trivedi K.S. Statistical non-parametric algorithms to estimate the optimal software rejuvenation schedule // *Proceedings of the IEEE Pacific Rim International Symposium on Dependable Computing*. 2000. pp. 77–84. DOI: 10.1109/PRDC.2000.897287.
38. Machida F., Kim D.S., Trivedi K.S. Modeling and analysis of software rejuvenation in a server virtualized system with live VM migration // *Performance Evaluation*. 2013. vol. 70. no. 3. pp. 212–230. DOI: 10.1016/j.peva.2012.09.003.
39. Zhang Y., Zheng Z., Lyu M.R. BFTCloud: A Byzantine Fault Tolerance Framework for Voluntary-Resource Cloud Computing // *Proceedings of the IEEE 4th International Conference on Cloud Computing*. 2011. pp. 444–451. DOI: 10.1109/CLOUD.2011.16.
40. Manker H., Hemant K.S., Iti S. Reliability analysis of a system using semi Markov process // *International Journal of Statistics and Applied Mathematics*. 2024. vol. 9. no. 4. pp. 145–151. DOI: 10.22271/math.2024.v9.i4b.1785.

41. Martyna J. Semi-Markov Model of Processing Requests Reliability and Availability in Mobile Cloud Computing Systems // *Advances in Intelligent Systems and Computing*. Springer. 2021. vol. 1389. pp. 264–272. DOI: 10.1007/978-3-030-76773-0_26.
42. Howard R.A. *Dynamic Probabilistic Systems: Semi-Markov and Decision Processes* // New York: Wiley. 1971. 564 p.
43. Machida F., Miyoshi N. Analysis of an optimal stopping problem for software rejuvenation in a deteriorating job processing system // *Reliability Engineering & System Safety*. 2007. vol. 168. pp. 128–135. DOI: 10.1016/j.res.2017.05.019.
44. Weibull W. A Statistical Distribution Function of Wide Applicability // *Journal of Applied Mechanics*. 1951. vol. 18. no. 3. pp. 293–297. DOI: 10.1115/1.4010337.
45. Rinne H. *The Weibull Distribution: A Handbook* // New York: CRC Press. 2008. 808 p. DOI: 10.1201/9781420087444.
46. Murthy D.N.P., Xie M., Jiang R. *Weibull Models* // New York: John Wiley & Sons. 2003. 416 p. DOI: 10.1002/047147326X.
47. Новиков И.С. Методы расчёта количественных показателей надёжности сложных программных комплексов на стадии проектирования и разработки // *Труды СПИРАН*. 2008. №6. С. 86–111. DOI: 10.15622/sp.6.8.
48. Okamura H., Dohi T. Faster Maximum Likelihood Estimation Algorithms for Markovian Arrival Processes // *Proceedings of the IEEE International Conference on Quantitative Evaluation of Systems*. 2009. pp. 73–82. DOI: 10.1109/QEST.2009.28.
49. Trivedi K., Andrade E., Machida F. Combining Performance and Availability Analysis in Practice // *Advances in Computers*. 2012. vol. 84. pp. 1–38. DOI: 10.1016/B978-0-12-396525-7.00001-0.
50. Bogatyrev V.A., Bogatyrev S.V., Bogatyrev A.V. Control of Multipath Transmissions in the Nodes of Switching Segments of Reserved Paths // *Proceedings of the International Conference on Information, Control, and Communication Technologies (ICCT)*. 2022. pp. 1–5. DOI: 10.1109/ICCT56057.2022.9976839.
51. Bogatyrev V.A., Bogatyrev A.V., Bogatyrev S.V. Multipath Transmission of Heterogeneous Traffic in Acceptable Delays with Packet Replication and Destruction of Expired Replicas in the Nodes that Make Up the Path // *Communications in Computer and Information Science*. Springer. 2023. vol. 1748. pp. 104–121. DOI: 10.1007/978-3-031-30648-8_9.
52. Уткин Л.В., Мулоха В.А., Лукашин А.А., Константинов А.В. Принятие решений в условиях неопределённости // *Санкт-Петербург: Политех-Пресс*, 2025. 49 с.
53. Уткин Л.В. Анализ риска и принятие решений при неполной информации // *Санкт-Петербург: Наука*, 2007. 403 с.

До Мань Киём — аспирант, факультет программной инженерии и компьютерной техники, Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО» (ИТМО). Область научных интересов: теория надёжности, теория массового обслуживания, отказоустойчивости и эффективности компьютерных систем и сетей. Число научных публикаций — 2. domanhkiemnd@gmail.com; Кронверкский пр., 49А, 197101, Санкт-Петербург, Россия; р.т.: +7(911)199-8063.

Богатырев Владимир Анатольевич — д-р техн. наук, профессор, факультет программной инженерии и компьютерной техники, Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО» (ИТМО). Область научных интересов: теория и методы обеспечения надёжности, отказоустойчивости и эффективности компьютерных систем и сетей. Число научных публикаций — 497. vladimir.bogatyrev@gmail.com; Кронверкский пр., 49А, 197101, Санкт-Петербург, Россия; р.т.: +7(812)232-5278.

M.K. DO, V. BOGATYREV
**SEMI-MARKOV RELIABILITY MODEL OF A COMPUTER
SYSTEM WITH CONTAINER VIRTUALIZATION**

Do M.K., Bogatyrev V. Semi-Markov Reliability Model of a Computer System with Container Virtualization.

Abstract. In modern cloud infrastructures and microservice architectures, ensuring high reliability, fault tolerance, and continuity of application operation is a critically important task. The primary factor reducing the efficiency of such systems is the phenomenon of software aging – the gradual degradation of system resources (memory leaks, data fragmentation) during prolonged operation, leading to a nonlinear increase in failure intensity. The complexity of reliability prediction under such conditions is обусловлена by the hierarchical dependency between the host's shared software and isolated containers, as well as the dynamic nature of recovery processes. This paper addresses the pressing problem of ensuring software reliability in a computer system with container virtualization subject to the phenomenon of software aging. The scientific novelty lies in the development of a comprehensive mathematical model based on semi-Markov processes, which accounts for the hierarchical structure of the system (host shared software and isolated containers) and enables the optimization of proactive recovery strategies. The Weibull distribution is used to describe the aging process, reflecting an increasing failure rate. A semi-Markov model of states and transitions is proposed, incorporating mechanisms of proactive container restart and migration. To reduce computational complexity, a simplified model is developed based on the assumption of negligibly low probability of secondary failures during short recovery intervals. Calculations of steady-state probabilities and availability are implemented in the Mathcad environment. The conducted numerical experiments showed that implementing a proactive recovery strategy within the interval range of 1200-2400 hours substantially improves the system availability. It is established that the error of the simplified model is negligible compared to the rigorous model. The proposed model allows determining the maximum availability value depending on the number of deployed containers and proactive container restart intervals. A comparison of the developed approach with traditional reactive recovery models confirms the advantage of proactive methods in minimizing downtime caused by error accumulation (memory leaks, resource fragmentation). The proposed model can be applied in the design of highly reliable cloud infrastructures and DevOps pipelines.

Keywords: reliability, availability factor, software, container virtualization, semi-Markov model, proactive recovery, container restart.

References

1. Dong H., Zhang C., Li G., Zhang H. Cloud-Native Databases: A Survey. IEEE Transactions on Knowledge and Data Engineering. vol. 36. no. 12. pp. 7772–7791. 2024. DOI: 10.1109/TKDE.2024.3397508.
2. Goyal P., Deora S.S. Reliability of Trust Management Systems in Cloud Computing. Indian Journal of Cryptography and Network Security. 2022. vol. 2. no. 1. pp. 1–5. DOI: 10.54105/ijcns.C1417.051322.
3. Assiri B., Sheneamer A. Fault tolerance in distributed systems using deep learning approaches. PLOS One. 2025. vol. 20. no. 1. DOI: 10.1371/journal.pone.0310657.
4. Khersonsky N.S., Bolshedvorskaya L.G. [Analysis of the development of statistical methods for assessing reliability as a quality property of objects and products]. Crede Experto: transport, obshchestvo, obrazovanie, yazyk – Crede Experto: transport, society,

- education, language. 2024. no. 1. pp. 6–23. DOI: 10.51955/2312-1327_2024_1_6. (In Russ.).
5. Gurjanov A.V., Korobeynikov A.G., Zharinov I.O., Zharinov O.O. Edge, fog and cloud computing in the cyber-physical systems networks. Proceedings of the III International Workshop on Modeling, Information Processing and Computing. 2021. pp. 103–108. DOI: 10.47813/dnit-mip3/2021-2899-103-108
6. Kushchazli A., Safargalieva A., Kochetkova I., Gorshenin A. Queuing Model with Customer Class Movement across Server Groups for Analyzing Virtual Machine Migration in Cloud Computing. Mathematics. 2024. vol. 12. no. 3. 468 p. DOI: 10.3390/math12030468.
7. Damarapati A. Containers vs. Virtual Machines: Understanding the Shift to Kubernetes. World Journal of Advanced Engineering Technology and Sciences. 2025. vol. 15. no. 1. pp. 852–861. DOI: 10.30574/wjaets.2025.15.1.0305.
8. Kozhirkbayev Z., Sinnott R.O. A performance comparison of container-based technologies for the Cloud. Future Generation Computer Systems. 2017. vol 68. pp. 175–182. DOI: 10.1016/j.future.2016.08.025.
9. Mirna A., Nadja K., Aris L. Utilization prediction-based VM consolidation approach. Journal of Parallel and Distributed Computing. 2022. vol. 170. pp. 24–38. DOI: 10.1016/j.jpdc.2022.08.001.
10. Polovko A.M., Gurov S.V. Osnovy' teorii nadezhnosti [Fundamentals of reliability theory]. Sankt-Peterburg: BXV-Peterburg. 2008. 702 p. (In Russ.).
11. Tatarnikova T.M., Arkhipov E.M. Designing Fault-tolerant Systems with Micro-service Architecture. Proceedings of the 27-th International Conference on Soft Computing and Measurements (SCM). 2024. pp. 348–351. DOI: 10.1109/SCM62608.2024.10554143.
12. Koren I., Krishna C.M. Fault-Tolerant Systems. Boston: Morgan Kaufmann. 2007. 400 p.
13. Yermakov S.G., Khalil M.M., Khomonenko A.D., Bukharova K.A. Evaluating the efficiency of fog computing on the Internet of Things using a Non-Markov model. T-Comm. 2022. vol. 16. no. 12. pp. 46–52.
14. Khomonenko A.D., Kiriynko A.B., Pugachev S.V. [Evaluation of the efficiency of fog computing in geoinformation systems]. Inzhenerny'j vestnik Dona – Engineering Bulletin of the Don. 2024. no. 6. pp. 364–382. (In Russ.).
15. Khomonenko A.D., Blagoveshchenskaya E.A., Prouzin O.V., Andruk A.A. [Forecasting the reliability of a cluster computing system using a semi-Markov model of alternative processes and monitoring]. Naukoemkie tekhnologii v kosmicheskix issledovaniyax Zemli – H&ES Research. 2018. vol. 10. no. 4. pp. 72–82. DOI: 10.24411/2409-5419-2018-10099. (In Russ.).
16. Bogatyrev V.A., Bogatyrev S.V., Bogatyrev A.V. [Assessment of the computer system's readiness for timely query service when combined with information recovery of memory after failures]. Nauchno-tekhnicheskij vestnik informacionny'x tekhnologij, mexaniki i optiki – Scientific and Technical Journal of Information Technologies, Mechanics and Optics. 2023. vol. 23. no. 3. pp. 608–617. DOI: 10.17586/2226-1494-2023-23-3-608-617. (In Russ.).
17. Bogatyrev V.A., Bogatyrev S.V., Bogatyrev A.V. Recovery of Real-Time Clusters with the Division of Computing Resources into the Execution of Functional Queries and the Restoration of Data Generated Since the Last Backup. Lecture Notes in Computer Science. Springer. 2024. vol. 14123. pp. 236–250. DOI: 10.1007/978-3-031-50482-2_19
18. Bogatyrev V.A., Vinokurova M.S. Control and Safety of Operation of Duplicated Computer Systems. Communications in Computer and Information Science. Springer. 2017. vol. 700. pp. 331–342. DOI: 10.1007/978-3-319-66836-9_28.
19. Aleksankov S.M. [Models of dynamic migration with an iterative approach and network migration of virtual machines]. Nauchno-tekhnicheskij vestnik informacionny'x tekhnologij, mexaniki i optiki – Scientific and Technical Journal

- of Information Technologies, Mechanics and Optics. 2015. vol. 15. no. 6. pp. 1098–1104. DOI: 10.17586/2226-1494-2015-15-6-1098-1104. (In Russ.).
20. Bogatyrev V.A., Derkach A.N. Evaluation of a Cyber-Physical Computing System with Migration of Virtual Machines During Continuous Computing. *Computers*. 2020. vol. 9. no 2. pp. 42. DOI: 10.3390/computers9020042.
21. Bogatyrev V.A., Fung V.K. [Assessment of the reliability of the cluster being restored by container virtualization]. *Nauchno-texnicheskij vestnik informacionny'x texnologij, mexaniki i optiki* – Scientific and Technical Journal of Information Technologies, Mechanics and Optics. 2025. vol. 25. no. 5. pp. 988–995. DOI: 10.17586/2226-1494-2025-25-5-988-995. (In Russ.).
22. Bogatyrev V.A., Bogatyrev S.V., Bogatyrev A.V. [Boundary estimation of cluster system reliability based on Markov model decomposition with limited node recovery with accumulated failures]. *Nauchno-texnicheskij vestnik informacionny'x texnologij, mexaniki i optiki* – Scientific and Technical Journal of Information Technologies, Mechanics and Optics. 2025. vol. 25. no. 3. pp. 574–583. DOI: 10.17586/2226-1494-2025-25-3-574-583. (In Russ.).
23. Rodriguez M.A., Buyya R. Container-based cluster orchestration systems: A taxonomy and future directions. *Software: Practice and Experience*. 2019. vol. 49. pp. 698–719. DOI: 10.1002/spe.2660.
24. Application Container Market Size And Forecast. Verified Market Research. Available at: www.verifiedmarketresearch.com/product/application-container-market. (accessed 16.06.2026). (In Russ.).
25. Application Container Market (2025-2023). Grand View Research. Available at: www.grandviewresearch.com/industry-analysis/application-container-market. (accessed 16.06.2026). (In Russ.).
26. Marella V. Comparative Analysis of Container Orchestration Platforms: Kubernetes vs. Docker Swarm. *International Journal of Scientific Research in Science and Technology*. 2024. vol. 11. no. 5. pp. 526–543. DOI: 10.32628/IJSRST24105254.
27. Grotke M., Trivedi K.S. Fighting bugs: remove, retry, replicate, and rejuvenate. *Computer*. 2007. vol. 40. no. 2. pp. 107–109. DOI: 10.1109/MC.2007.55.
28. Cotroneo D., Natella R., Pietrantuono R., Russo S. Software Aging Analysis of the Linux Operating System. *Proceedings of the IEEE 21st International Symposium on Software Reliability Engineering (ISSRE)*. 2010. pp. 71–80. DOI: 10.1109/ISSRE.2010.24.
29. Nascimento M.G., Moura R.J., Machida F., Andrade E. Comparison of Machine Learning Algorithms for Detecting Software Aging in SQL Server. *Proceedings of the 13th Latin-American Symposium on Dependable and Secure Computing*. 2024. pp. 159–167. DOI: 10.1145/3697090.3699798.
30. Oliveira F., Araujo J., Matos R., Maciel P. Software Aging in Container-Based Virtualization: An Experimental Analysis. *Proceedings of the 16th Iberian Conference on Information Systems and Technologies (CISTI)*. 2021. pp 1–7. DOI: 10.23919/CISTI52073.2021.9476625.
31. Nguyen T.A., Min D., Choi E., Tran T.D. Reliability and Availability Evaluation for Cloud Data Center Networks Using Hierarchical Models. *IEEE Access*. 2019. vol. 7. pp. 9273–9313. DOI: 10.1109/ACCESS.2019.2891282.
32. Shruthi P., Nagaraj G.C. Software aging prediction – a new approach. *International Journal of Electrical and Computer Engineering*. 2023. vol. 13. no 2. pp. 1773–1781. DOI: 10.11591/ijece.v13i2.pp1773-1781.
33. Pietrantuono R., Russo S. A survey on software aging and rejuvenation in the cloud. *Software Quality Journal*. 2020. vol 28. pp. 7–38. DOI: 10.1007/s11219-019-09448-3.

34. Turk L.I.A., Alsolami E.G. Jelinski-Moranda Software Reliability Growth Model: A Brief Literature and Modification. *International Journal of Software Engineering & Applications (IJSEA)*. 2016. vol. 7. no. 2. pp. 33–44. DOI: 10.5121/ijsea.2016.7204.
35. Li N., Tan Y., Wang X., Li B., Luo J. *Dynamic Resource Allocation for Containerized Applications in Edge Computing. Lecture Notes in Electrical Engineering*. Springer. 2023. vol. 1015. pp. 121–130. DOI: 10.1007/978-981-99-0416-7_12.
36. Musa J.D., Iannino A., Okumoto K. *Software Reliability: Measurement, Prediction, Application*. New York: McGraw-Hill. 1987. 621 p.
37. Dohi T., Goseva-Popstojanova K., Trivedi K.S. Statistical non-parametric algorithms to estimate the optimal software rejuvenation schedule. *Proceedings of the IEEE Pacific Rim International Symposium on Dependable Computing*. 2000. pp. 77–84. DOI: 10.1109/PRDC.2000.897287.
38. Machida F., Kim D.S., Trivedi K.S. Modeling and analysis of software rejuvenation in a server virtualized system with live VM migration. *Performance Evaluation*. 2013. vol. 70. no. 3. pp. 212–230. DOI: 10.1016/j.peva.2012.09.003.
39. Zhang Y., Zheng Z., Lyu M.R. BFTCloud: A Byzantine Fault Tolerance Framework for Voluntary-Resource Cloud Computing. *Proceedings of the IEEE 4th International Conference on Cloud Computing*. 2011. pp. 444–451. DOI: 10.1109/CLOUD.2011.16.
40. Manker H., Hemant K.S., Iti S. Reliability analysis of a system using semi Markov process. *International Journal of Statistics and Applied Mathematics*. 2024. vol. 9. no. 4. pp. 145–151. DOI: 10.22271/math.2024.v9.i4b.1785.
41. Martyna J. Semi-Markov Model of Processing Requests Reliability and Availability in Mobile Cloud Computing Systems. *Advances in Intelligent Systems and Computing*. Springer. 2021. vol. 1389. pp. 264–272. DOI: 10.1007/978-3-030-76773-0_26.
42. Howard R.A. *Dynamic Probabilistic Systems: Semi-Markov and Decision Processes*. New York: Wiley. 1971. 564 p.
43. Machida F., Miyoshi N. Analysis of an optimal stopping problem for software rejuvenation in a deteriorating job processing system. *Reliability Engineering & System Safety*. 2007. vol. 168. pp. 128–135. DOI: 10.1016/j.res.2017.05.019.
44. Weibull W. A Statistical Distribution Function of Wide Applicability. *Journal of Applied Mechanics*. 1951. vol. 18. no. 3. pp. 293–297. DOI: 10.1115/1.4010337.
45. Rinne H. *The Weibull Distribution: A Handbook*. New York: CRC Press. 2008. 808 p. DOI: 10.1201/9781420087444.
46. Murthy D.N.P., Xie M., Jiang R. *Weibull Models*. New York: John Wiley & Sons. 2003. 416 p. DOI: 10.1002/047147326X.
47. Novikov I.C. [Methods of calculation of quantity indicators of reliability of difficult program complexes on a design stage and development]. *Trudy SPIRAN – SPIIRAS Proceedings*. 2008. no. 6. pp. 86–111. DOI: 10.15622/sp.6.8. (In Russ.).
48. Okamura H., Dohi T. Faster Maximum Likelihood Estimation Algorithms for Markovian Arrival Processes. *Proceedings of the IEEE International Conference on Quantitative Evaluation of Systems*. 2009. pp. 73–82. DOI: 10.1109/QEST.2009.28.
49. Trivedi K., Andrade E., Machida F. Combining Performance and Availability Analysis in Practice. *Advances in Computers*. 2012. vol. 84. pp. 1–38. DOI: 10.1016/B978-0-12-396525-7.00001-0.
50. Bogatyrev V.A., Bogatyrev S.V., Bogatyrev A.V. Control of Multipath Transmissions in the Nodes of Switching Segments of Reserved Paths. *Proceedings of the International Conference on Information, Control, and Communication Technologies (ICCT)*. 2022. pp. 1–5. DOI: 10.1109/ICCT56057.2022.9976839.
51. Bogatyrev V.A., Bogatyrev A.V., Bogatyrev S.V. Multipath Transmission of Heterogeneous Traffic in Acceptable Delays with Packet Replication and Destruction of Expired Replicas in the Nodes that Make Up the Path.

- Communications in Computer and Information Science. Springer. 2023. vol. 1748. pp. 104–121. DOI: 0.1007/978-3-031-30648-8_9.
52. Utkin L.V., Mulukha V.A., Lukashin A.A., Konstantinov A.V. Prinyatie reshenij v usloviyax neopredelennosti [Decision-making in conditions of uncertainty]. Sankt-Peterburg: Politekh-Press. 2025. 49 p. (In Russ.).
 53. Utkin L.V. Analiz riska i prinyatie reshenij pri nepolnoj informacii [Risk analysis and decision-making with incomplete information]. Sankt-Peterburg: Nauka. 2007. 403 p. (In Russ.).

Do Manh Kiem — Graduate Student, Department of Software Engineering and Computer Science, ITMO University. Research interests: reliability theory, queueing theory, fault tolerance, and the efficiency of computer systems and networks. The number of publications — 2. domanhkiemnd@gmail.com; 49A, Kronverkskiy Ave., 197101, St. Petersburg, Russia; office phone: +7(911)199-8063.

Bogatyrev Vladimir — Ph.D., Dr.Sci., Professor, Department of Software Engineering and Computer Science, ITMO University. Research interests: theory and methods for ensuring the reliability, fault tolerance, and efficiency of computer systems and networks. The number of publications — 497. vladimir.bogatyrev@gmail.com; 49A, Kronverkskiy Ave., 197101, St. Petersburg, Russia; office phone: +7(812)232-5278.