УДК 004.9:681.3
doi: 10.21685/2072-3059-2025-3-3

Формализация и реализация логико-вероятностных и логико-алгебраических операционных моделей функциональной архитектуры кластерных вычислительных систем

Г. В. Петушков[1], А. С. Сигов[2]

[1,2]МИРЭА – Российский технологический университет, Москва, Россия

[1]petushkov@mirea.ru, [2]sigov@mirea.ru

Аннотация. *Актуальность и цели*. На основе проведенного исследования разработок в области функциональной архитектуры кластерных вычислительных систем показана актуальность выполнения работ в данном направлении, что обусловлено увеличением спроса на доступные высокопроизводительные вычисления, растущим внедрением методов искусственного интеллекта и машинного обучения. Цель – разработка и экспериментальная апробация на уровне имитационных моделей методики и технологии реализации функциональной архитектуры вычислительных систем кластерного типа на основе высокоскоростных коммутаторов. *Материалы и методы.* Использован логико-вероятностный подход к созданию моделей определяемой приложениями и программным обеспечением промежуточного уровня *middleware* функциональной архитектуры кластерных вычислительных систем, позволяющий ускорить создание имитационных моделей для ряда важных режимов использования кластера и осуществить переход к логико-алгебраическим формализованным спецификациям на программные приложения. *Результаты.* Построены имитационные логико-вероятностные и логико-алгебраические модели для ряда важных вариантов использования вычислительного кластера, проведены необходимые статистические эксперименты с данными моделями, давшие обоснования к реализациям соответствующему моделям программному обеспечению промежуточного уровня *middleware*. *Выводы.* Предложена методология разработки функциональной архитектуры вычислительной системы кластерного типа, определяемой спецификациями в форме логико-вероятностных и родственных им логико-алгебраических моделей, что может ускорить подготовку кластера к эксплуатации в организации.

Ключевые слова: вычислительная система кластерного типа, приложение промежуточного уровня, функциональная архитектура, логико-вероятностная модель, логико-алгебраическая модель, режимы обработки запросов, результаты моделирования

Для цитирования: Петушков Г. В., Сигов А. С. Формализация и реализация логико-вероятностных и логико-алгебраических операционных моделей функциональной архитектуры кластерных вычислительных систем // Известия высших учебных заведений. Поволжский регион. Технические науки. 2025. № 3. С. 26–62. doi: 10.21685/2072-3059-2025-3-3

Formalization and implementation of logical-probabilistic and logical-algebraic operational models of the functional architecture of cluster computing systems

G.V. Petushkov[1], A.S. Sigov[2]

[1,2]MIREA – Russian Technological University, Moscow, Russia

[1]petushkov@mirea.ru, [2]sigov@mirea.ru

Abstract. *Background.* Based on the conducted study of developments in the field of functional architecture of cluster computing systems, the relevance of work in this direction is shown, which is due to the increasing demand for affordable high-performance computing, the growing implementation of artificial intelligence and machine learning methods. The purpose of the work is to develop and experimentally test at the level of simulation models a methodology and technology for implementing the functional architecture of cluster-type computing systems based on high-speed switches. *Materials and methods.* A logical-probabilistic approach is used to create models of the functional architecture of cluster computing systems determined by applications and middleware, which allows accelerating the creation of simulation models for a number of important modes of cluster use and making the transition to logical-algebraic formalized specifications for software applications. *Results.* Simulation logical-probabilistic and logical-algebraic models for a number of important variants of using a computing cluster were constructed, the necessary statistical experiments with these models were carried out, which provided justification for implementing the corresponding models of the middleware software. *Conclusions.* A methodology for developing the functional architecture of a cluster-type computing system, defined by specifications in the form of logical-probabilistic and related logical-algebraic models, is proposed, which can accelerate the preparation of the cluster for operation in an organization.

Keywords: cluster-type computing system, middleware-level application, functional architecture, logical-probabilistic model, logical-algebraic model, query processing modes, modeling results

For citation: Petushkov G.V., Sigov A.S. Formalization and implementation of logical-probabilistic and logical-algebraic operational models of the functional architecture of cluster computing systems. *Izvestiya vysshikh uchebnykh zavedeniy. Povolzhskiy region. Tekhnicheskie nauki = University proceedings. Volga region. Engineering sciences.* 2025;(3):26–62. (In Russ.). doi: 10.21685/2072-3059-2025-3-3

Введение

Кластерные вычислительные системы, как правило, отличаются архитектурной простотой, обусловленной регулярностью и однородностью, разделением аппаратуры, связью между узлами в ограниченной окрестности, топологией, не зависящей от размерности кластера, т.е. горизонтальной масштабируемостью. Реже для вычислительных кластеров характерна вертикальная или иерархическая масштабируемость. Тем не менее применение топологии, не зависящей от размерности, не препятствует созданию других виртуальных топологий, необходимых по условиям решения задач, при допустимых ограничениях производительности вследствие дополнительных виртуальных межузловых связей [1–3].

Обычно вычислительный кластер определяется как разновидность параллельной или распределенной системы, которая состоит из нескольких связанных между собой компьютеров, используемых как единый, унифицированный компьютерный ресурс [4–7].

Кластеры, предназначенные для высокопроизводительных вычислений, так называемые HPC-кластеры (HPC – High Performance Computing), позволяют выполнять большое количество однотипных вычислительных операций за короткое время. При этом вычислительный процесс распараллеливается не только между ядрами одного вычислительного узла, но и по всему набору компьютеров, которые связаны высокоскоростными каналами передачи данных. Современный уровень развития технологий позволяет организовывать

кластеры производительностью в десятки, сотни и тысячи терафлопс [8–11]. Кластерную HPC-архитектуру имеют, например, российские суперкомпьютеры «Червоненкис», «Галушкин» и «Ляпунов», используемые в дата-центрах Яндекса [12].

В решении проблемы обеспечения высокой производительности мультипроцессорных вычислительных систем большое значение имеет завершение отечественного проекта создания коммуникационной сети «Ангара», а также других коммуникационных структур, разработанных в России [13].

Известен целый ряд подобных проектов, перечисленных на сайте *parallel.ru* – сайте лаборатории параллельных информационных технологий Научно-исследовательского вычислительного центра МГУ, на котором собрана информация о параллельных вычислениях, технологиях параллельного программирования, высокопроизводительных компьютерах, метакомпьютинге, суперкомпьютерных центрах и др.

Технологии построения систем распределенных вычислений, или *grid*-систем, также можно условно отнести к кластерным технологиям. Отличие заключается в меньшей доступности вычислительных *grid*-узлов для контроля, что отрицательно сказывается на их надежности.

Вычислительным системам кластерного типа «родственны» однородные и распределенные вычислительные системы, разработанные в разное время в Академгородке Сибирского отделения Академии наук РАН. В трудах ученых исследована организация параллельных вычислений в многомодульных системах, где в качестве модулей используются элементарные машины, обладающие возможностями хранения, переработки и транспортировки данных [14, 15]. Разработана первая в мире программно-реконфигурируемая ВС «Минск-222», созданы мини-ВС МИНИМАКС и СУММА, семейство микро-ВС МИКРОС, МИКРОС-2 и МИКРОС-Т [16–18].

Отличием современных вычислительных систем кластерного типа на основе локальных сетей с коммутаторами 2-го (канального) и 3-го (межсетевого) уровней от разработанных ранее однородных вычислительных систем (ОВС) является бо́льшая гибкость архитектуры, обеспечивающая лучшие возможности реконфигурации на уровне связей между модулями, повышенная отказоустойчивость.

Многие фирмы дали оценку достоинств и преимуществ кластерной архитектуры: возможна равномерная или определяемая правилами балансировка запросов между узлами кластера; высокая отказоустойчивость за счет использования нескольких узлов и резервных каналов; масштабируемость – простое увеличение производительности кластера за счет увеличения количества узлов; возможность обслуживания и замены узлов кластера без остановки работы приложения [10, 11].

Для многих кластерных вычислительных систем в силу особенностей их построения на основе коммутаторов второго и третьего уровней характерна работа в пиринговом режиме, когда вычислительный узел может работать и в режиме клиента, и в режиме сервера, другими словами, – может работать как на прием, так и на передачу. В подобных системах узлы кластера могут обмениваться ресурсами напрямую, без посредника-сервера, либо каждый из узлов может выполнять функции сервера или управлять работой кластера. Возможен также такой пиринговый режим работы кластера, при котором

функции центрального сервера сведены лишь к контролю за потоками информации и временному размещению данных [19–22].

К числу впервые реализованных вычислительных кластеров принадлежит, например, «Беовульф» (NASA, США) – компьютерный кластер, состоящий из идентичных компьютеров потребительского класса, объединенных в небольшую локальную сеть с установленными библиотеками и программами, которые позволяют совместно использовать ресурсы [23]. В результате получается вычислительный кластер для параллельных вычислений из недорогих персональных компьютеров (ПК). Обычно используются библиотеки для параллельной обработки: интерфейс передачи сообщений (MPI) и параллельная виртуальная машина (PVM) [24].

Кластерная архитектуру одной из первых также получила вычислительная система Blue Gene/L (фирма IBM) с производительностью до 1 петафлопс; это масштабируемый суперкомпьютер, который может состоять из более чем 65536 вычислительных процессоров [25].

Примером крупного кластера, используемого в астрофизических расчетах, является, например, высокопроизводительный компьютерный кластер DEGIMA, используемый в Центре передовых вычислений Университета Нагасаки (Япония) [26]. Система состоит из 144-узлового кластера ПК, соединенных через интерфейс InfiniBand. Каждый узел состоит из процессора Intel Core i7 920 с тактовой частотой 2,66 ГГц, двух видеокарт GeForce GTX295, 12 Гб памяти DDR3-1333 и Mellanox MHES14-XTC SDR InfiniBand-адаптера на материнской плате MSI X58 pro-E. Каждая видеокарта оснащена двумя графическими процессорами GT200. В целом система состоит из 144 центральных процессоров и 576 графических процессоров [26].

Технологии Infiniband и RoCE используются для высокоскоростного обмена данными в вычислительных кластерах [27, 28]. Специализированная сетевая архитектура Infiniband может достигать пропускной способности до 800 Гбит/с.

Архитектура современных кластеров строится на основе архитектурных решений в области межузловых коммуникационных сетей, позволяющих создавать сетевые инфраструктуры – как коммерчески доступные, так и высшего диапазона производительности типа Infiniband (Mellanox, США), OmniPass (Intel, США), Ангара (РФ), 6D-тор (Fujitsu, Япония), Sugon (Китай), линейки коммутаторов CRAY высшего диапазона производительности (США) и др. [29–32].

Технология создания кластеров на базе высокоскоростных коммутаторов доступна многим компаниям в разных странах; в настоящее время более 50 высокопроизводительных кластеров на коммутаторах входят в верхнюю часть списка Top 500 [33]. Многие системы распределенных вычислений могут быть отнесены к кластерам. Отличительной особенностью таких кластеров, являющихся частями больших вычислительных систем и сетей, является относительно невысокая доступность узлов-компьютеров, которые в силу разных причин подключаются и отключаются в процессе работы. В этом случае переменность конфигурации компенсируется подключением дополнительных узлов.

Естественная распределенность развертывания узлов кластера часто может быть связана с его реализацией на базе локальной или глобальной сети

с предметной ориентацией на распределенные предприятия, филиалы, офисы, производственные участки и др. В первом случае кластер может быть развернут в среде, где каждый узел должен находиться рядом с управляемым объектом или с другим источником информации. Во втором случае кластер может выполнять функции так называемого метакомпьютера, узлы которого распределены в глобальной вычислительной сети, например в Интернете. К крупномасштабным интернет-кластерам может быть отнесена *grid*-платформа BOINC (Berkeley Open Infrastructure for Network Computing) [34], предназначенная для «волонтерских» научных вычислений; распределенная сеть BOINC может содержать до миллиона и более активных пользовательских компьютеров, на базе которых организуется коллективный вычислительный кластер.

В последнее время большое внимание уделяется так называемым системам высокой готовности (англ. High Availability Systems), от которых требуется минимизация времени простоя. При этом обычно выделяют четыре уровня готовности [35]:

Уровень готовности, %	Макс. время простоя	Тип системы
99,0	3,5 дня в год	Обычная (Conventional)
99,9	8,5 ч в год	Высокая надежность (High Availability)
99,99	1 ч в год	Отказоустойчивая (Fault Resilient)
99,999	5 мин в год	Безотказная (Fault Tolerant)

Соблюдение этих уровней готовности особенно требуется от корпоративных кластерных систем. При подготовке вычислительного кластера к использованию на предприятии или учреждении возникает проблемная ситуация выбора приложений, для которых использование кластера будет приемлемым или наиболее эффективным по критерию производительности. Поэтому решение подобной проблемы является актуальной научной задачей.

1. Логико-вероятностные и логико-алгебраические модели функциональной архитектуры кластерных вычислительных систем

1.1. Особенности организации кластерных вычислительных систем

При разработке моделей и реальных приложений использованы логико-вероятностный и логико-алгебраический подходы. Оба подхода основаны на элементах исчисления высказываний, исчисления предикатов первого порядка, алгебре алгоритмов, на методах дискретно-событийного моделирования и создания сетевых приложений.

Первый подход предназначен для создания имитационных статистических моделей функционирования кластерных и других вычислительных систем. При создании статистических моделей используются программные генераторы псевдослучайных чисел с различными законами распределения, а также программы для сбора статистических данных о времени задержек, о загрузке компьютеров.

Второй подход основан на создании предварительных спецификаций прототипного программного обеспечения для реальных приложений. В то же время ряд особенностей функционирования реального программного обеспечения в моделях не учтен, поэтому при программировании моделей следует учитывать возможность появления дополнительных задержек и передач управляющей информации.

В настоящей работе рассматриваются вопросы использования логико-вероятностного и логико-алгебраического подходов при разработке моделей и реальных приложений для вычислительных систем кластерного типа и других вычислительных систем. Логико-вероятностный подход соответствует методике моделирования, получившей название «событийно-управляемое моделирование» (EDM – от англ. *Event-Driven Modeling*). Логико-алгебраический подход тоже используется для построения имитационных моделей, но в первую очередь он предназначен для создания формализованных спецификаций компьютерных программ и в данной работе по аналогии получает название «управляемое моделями программирование» (MDP – *Model-Driven Programming*).

Для лучшего запоминания этих задач также используются метафоры, но, естественно, имеются в виду задачи обработки данных и макроанализа производительности преимущественно кластерных вычислительных систем. Приведены примеры формализации модели «Круглый стол» о случайных парных взаимодействиях процессов в вычислительном кластере, модели «Резидент – агенты», посвященной формализации взаимодействия процесса подготовки заданий для кластера и последовательной загрузки агентов-исполнителей. Задачи «Собрание 1» и «Собрание 2» похожи на предыдущую, но в них задана параллельная и распределенная обработка информации. Анализ характеристик производительности моделируемых систем доведен до численных результатов и работающих приложений в кластере, построенном на основе коммутаторов второго и третьего уровней.

На рис. 1 представлена распространенная структура вычислительной системы, включающей в свой состав вычислительный кластер, условно названный «M1-M16». Узлы этого кластера взаимодействуют между собой через коммутатор K3 по протоколу канального уровня L2. Узлы кластера адресуются по физическим MAC-адресам. Коммутатор K2 используется кластером как резерв, а также для связи с серверами S1-S4. Этот коммутатор может использовать как IP-адреса, так и аппаратные MAC-адреса, брать на себя функцию маршрутизации пакетов. Коммутатор K1 представляет собой коммутатор второго уровня L2+, использующий дополнительные функции, позволяющие работать с IP-адресами [36, 37].

Как было сказано ранее, на основе коммутаторов локальных сетей различных уровней строятся вычислительные кластеры различной производительности – от высокопроизводительных кластеров HPC до массовых недорогих кластеров, иногда называемых кластерами-лоукостерами, т.е. кластерами низкой стоимости. Простота реализации обусловливает широкую распространенность кластеров вычислительных машин. Функциональная архитектура сетевых кластеров обычно строится на основе программного обеспечения различного уровня. Операционные системы узлов кластера не зависят друг от друга. Организация работы кластера начинается обычно с промежу-

точного уровня (уровня *middleware*) и уровня приложений. Работу важных приложений следует рассмотреть подробнее.

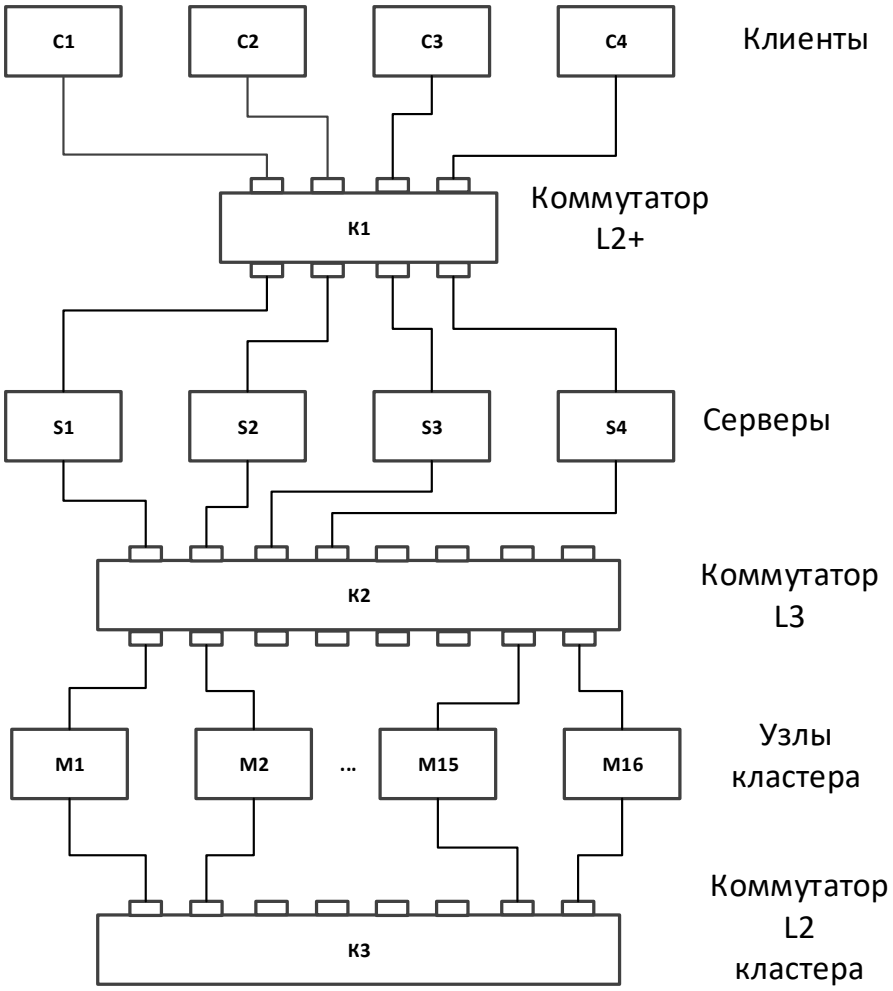


Рис. 1. Сетевая инфраструктура вычислительного кластера «M1-M16»

1.2. Выбор методологического подхода к макро- и микроанализу функционирования вычислительных систем кластерного типа

Имитационная, или поведенческая, модель вычислительного кластера должна быть основана на выборе концептуальной схемы. Концептуальная схема должна базироваться на определенном методологическом подходе. В рамках этого подхода описываются функциональные взаимосвязи системы, т.е. ее функциональная архитектура. Общеизвестно, что методологический подход, выбранный исследователем, позволяет ему четко сформулировать описание системы [38]. Методологический подход в настоящей работе основан на выборе формальных моделей: логических исчислений, таких как исчисление высказываний и предикатов; исполнимых, или операционных моделей, таких как сети Петри и конечные автоматы; модели параллельных, конкурирующих и распределенных вычислений, которые относятся также к тео-

ретическому и прикладному программированию [39–41]. К последнему относятся технологии автоматного программирования [42] и программирования на основе логико-алгебраических спецификаций [43–45].

В дальнейшем речь пойдет о *макро-* и *микроанализе* функционирования вычислительных систем кластерного типа, поэтому проследим возникновение этих терминов на примерах некоторых общенаучных и технических дисциплин. Средства вычислительной техники используются практически во всех сферах деятельности современного человека: привычными стали термины Интернет, интернет вещей, большие данные, искусственный интеллект. Информатизация общества непрерывно растет, что в свою очередь требует постоянного развития компьютерной техники. Практически все отрасли науки и техники стали смежными по отношению к информатике и вычислительной технике.

Например, в экономике *микроанализ* исследует мотивы и принципы принятия субъектами решений, их взаимодействие [46]. *Макроанализ* позволяет исследовать экономику как систему связей между агрегатами – совокупностями экономических единиц, рассматриваемых как одна обобщенная единица [47]. Технико-экономический анализ серверов как вычислительных модулей вычислительных систем класса WSC (Warehouse Scale Computer – компьютер хранилища данных) на указанных выше уровнях выполнен в нашей работе [48].

В известной монографии [49] (Э. В. Евреинова и Ю. Г. Косарева) на *макроструктурном* уровне для универсальных вычислительных систем (УВС) различают элементарные машины, коммутаторы и каналы связи. На *микроструктурном* уровне рассматриваются структуры подсистем УВС, состоящие из микроэлементов, под которыми понимаются рассматриваемые как неделимое целое элементы, из которых построены все макроэлементы.

В монографии Б. Байцера «Микроанализ производительности вычислительных систем» [50] определено, что «при *макроанализе* системные характеристики выводятся на основании поведения всей системы… при выполнении *микроанализа* исследования начинаются на самом нижнем уровне».

1.3. Построение логико-вероятностной модели функционирования вычислительного кластера, реализующего равнодоступные парные взаимодействия (модель «Круглый стол»)

Рассматривается задача макроанализа производительности вычислительного кластера в следующей содержательной постановке. В кластере реализуется вычислительный процесс, сопровождаемый межузловым обменом запросами и затребованными данными. Как узлы, отправляющие запросы (узлы-источники), так и узлы, принимающие запросы (узлы-приемники), выбираются в соответствии с заданным распределением вероятностей. Каждый запрос предваряется выполнением некоторой подготовительной программы на узле-отправителе. Ответ на запрос сопровождается выполнением некоторой программы на узле-приемнике запроса. Естественным ограничением является тот факт, что номера узла-источника и узла-приемника не могут совпадать. Запросы, подаваемые с одного и того же узла, могут повторяться, что может привести к задержкам при передаче и подготовке запроса. К задержкам также приводят обращения некоторых узлов-источников к одним и тем

же узлам-приемникам. Таким образом, в вычислительной системе кластерного типа будут взаимодействовать конкурирующие процессы, и образование очередей ожидания не нарушит работу системы.

Рассматриваемый режим работы кластера является важной разновидностью диалоговых взаимодействий в компьютерных сетях. На содержательном уровне постановки задачи можно привести метафорический аналог задачи о круглом столе. Согласно задаче «Круглый стол» в беседе на какую-то научную тему участвуют N ученых. Сразу после оглашения председателем вопроса, подлежащего обсуждению, каждый участник выбирает одного собеседника, готовит вопросы к нему и затем вступает в беседу. В беседах могут участвовать только независимые пары ученых. Председатель переходит к оглашению следующего запроса только тогда, когда все беседы завершены (число «бесед» заранее ограничено). Результатом решения задачи должно быть экспериментально определенное распределение вероятностей (гистограмма) продолжительностей бесед, включающих также и время подготовки (обдумывания) предстоящей и состоявшейся впоследствии беседы.

Формализация решения задачи состоит в следующем построении логико-вероятностной модели функционирования вычислительного кластера, реализующего равнодоступные парные взаимодействия. Предлагается провести формализацию в два этапа. На первом этапе строится логико-вероятностная имитационная поведенческая модель, в которой понятие исполнимости в реальной системе ограничивается некоторыми модулями, работа которых задается распределением вероятностей некоторого интервала времени функционирования. На втором этапе формальное описание модели работа модулей уточняется, что позволяет называть модель логико-алгебраической, пригодной для последующей реализации на ее основе программного приложения, работающего в реальном вычислительном кластере.

Моделирование сложных крупномасштабных систем, к которым относится и моделирование кластерных вычислительных систем, представляет сложную задачу. Ее особенностью является большое число состояний, что затрудняет использование таких формализмов, как конечные автоматы и сети Петри. Поэтому для решения этой задачи выбрано построение моделей, основанных на исчислении предикатов первого порядка. Отличием от других логических моделей, использующих исчисление предикатов, является придание модели динамичности за счет использования операций модификации предикатов, определенных на множествах предметной области.

Рассмотрим полный орграф $G = (V, U)$, где V – множество вершин, $U = V^2$ – полное множество дуг, $V^2 = V \times V$. Введем S_x: $V_x \to \{\textbf{true}, \textbf{false}\}$ – унарный предикат, определенный на подмножестве вершин-источников запросов в кластерной вычислительной системе, S_y: $V_y \to \{\textbf{true}, \textbf{false}\}$ – унарный предикат, определенный на подмножестве вершин-приемников запросов в кластере; подмножества $V_x \subset V$ и $V_y \subset V$ здесь имеют переменный состав и формируются в процессе моделирования. Истинные значения предикатов S_x и S_y, играющих соответственно роли характеристических функций для подмножеств V_x и V_y, будут соответствовать назначению соответствующих вершин из множества V на участие в передаче данных диалогов от источников к приемникам. Очевидно, что V_x и V_y (согласно постановке задачи) непересекающиеся подмножества ($V_x \cap V_y = \varnothing$) множества V. Предикаты S_x, S_y и мно-

жества V_x, V_y изменяются (эволюционируют) в процессе моделирования, причем на любом этапе моделирования множества V_x и V_y равномощны. Моделирование диалоговой системы «Круглый стол» состоит в формировании последовательности орграфов паросочетаний H_0, H_1, H_2, …, H_m, имитирующих подключение и отключение участников диалога в кластерной вычислительной системе. Одновременно с этим производится сбор статистических данных, характеризующих загрузку и производительность системы. В теории графов паросочетание, или независимое множество дуг в орграфе, – это набор попарно несмежных дуг. Таким образом, орграфы паросочетаний будут содержать множества некоторых упорядоченных пар (x, y), где $x \in V_x$, $y \in V_y$, $x \neq y$. Задача моделирования упрощается тем, что x и y $(x \neq y)$ выбираются из одного и того же множества V, т.е. из множества участников диалога в кластерной вычислительной системе.

Орграф паросочетаний не может иметь петель, так как полагается, что у участника круглого стола нет необходимости в передаче сообщений самому себе. Если паросочетание орграфа покрывает все его вершины, то это соответствует связи максимального числа пар. Очевидно, что при четном числе вершин n число таких пар максимально и равно $n/2$. Число пар меньше данной величины в случаях, когда наблюдается конфликт запросов при обращении на передачу к одним и тем же приемникам.

Эволюцию орграфов паросочетаний при моделировании предлагается организовать при помощи следующего логико-вероятностного выражения, сочетающего в себе как декларативные, так и процедурные знания о предметной области:

$$L_0 = [(\exists_{\boldsymbol{Any}}\, x \in V) \neg S_x(x)]([(\exists_{\boldsymbol{One}}\, (x, y) \in V^2) \neg S_y(y)\ \&\ (x \neq y)]$$
$$(\{S_x(x) \leftarrow \textbf{true}, S_y(y) \leftarrow \textbf{true}, R(x, y) \leftarrow \textbf{true}, \mathit{Transfer}(x, y) \leftarrow \textbf{true},$$
$$\mathit{Delay}(x, y) \leftarrow \textbf{true}, S_x(x) \leftarrow \textbf{false}, S_y(y) \leftarrow \textbf{false}, R(x, y) \leftarrow \textbf{false},$$
$$\mathit{Delay}(x, y) \leftarrow \textbf{false}, \mathit{Transfer}(x, y) \leftarrow \textbf{false}\} \vee \mathit{Ret}) \vee \mathit{Ret}), \qquad (1)$$

где в квадратные скобки заключены условия успешного выполнения заданных операторов. В фигурные скобки заключены операции модификации предикатов. Квадратные и фигурные скобки являются элементами синтаксиса выражения (1) и доопределяют его операционную семантику. Оператор $\exists_{\boldsymbol{Any}}\, x \in V$ – это оператор псевдослучайного выбора будущей вершины x графа паросочетаний H при условии, что ранее эта вершина в граф не входила, т.е. выбор реализуется успешно, если истинно высказывание $\neg S_x(x)$ при конкретном вхождении x. При выполнении данной операции используется генератор целочисленных псевдослучайных величин с заданным законом распределения; вид выбранного распределения вероятностей может задавать приоритетный выбор запросов от участников «Круглого стола». При равномерном распределении моделируется равновероятный выбор запросов. Далее оператор $\exists_{\boldsymbol{One}}\, (x, y) \in V^2$ осуществляет выбор кортежа (x, y) из декартова произведения $V^2 = V \times V$ с учетом того, что первый элемент кортежа x уже известен в результате предыдущего выбора и истинно высказывание $\neg S_y(y)$ & ($x \neq y$). Отметим, что операторы $\exists_{\boldsymbol{Any}}$ и $\exists_{\boldsymbol{One}}$ не удаляют элементы или кортежи из множеств, а лишь предварительно отмечают их в структурах – списках. Включе-

ние или удаление этих элементов осуществляется путем модификации соответствующих предикатов – характеристических функций множеств. В фигурные скобки заключены операции модификации унарных $S_x(x) \leftarrow$ **true**, $S_y(y) \leftarrow$ **true** и бинарного $R(x, y) \leftarrow$ **true** предикатов, где значения предметных переменных x, y известны. В результате подобных модификаций добавляются две вершины и инцидентная им дуга в очередной конфигурации графа H паросочетаний. Это построение является ядром моделирующей программы и в дальнейшем используется при реализации программного обеспечения кластерной системы.

Орграф паросочетаний имеет переменную (эволюционирующую) структуру и его текущее состояние формально может быть определено следующим образом:

$$H = (V_x, V_y, S_x, S_y, R). \quad (2)$$

На рис. 2 представлена возможная реализация орграфа паросочетаний с шестью дугами, которые включаются в орграф при моделировании. Шести дугам соответствуют шесть кортежей (x_1, x_{13}), (x_4, x_7), (x_6, x_{14}), (x_9, x_{16}), (x_{15}, x_3) и (x_{11}, x_5), входящих в бинарное отношение R. Это отношение является областью истинности одноименного бинарного предиката. Указанным парам в свою очередь соответствуют истинные высказывания $R(x_1, x_{13})$, $R(x_4, x_7)$, $R(x_6, x_{14})$, $R(x_9, x_{16})$, $R(x_{15}, x_3)$ и $R(x_{11}, x_5)$, где R – бинарный предикатный символ.

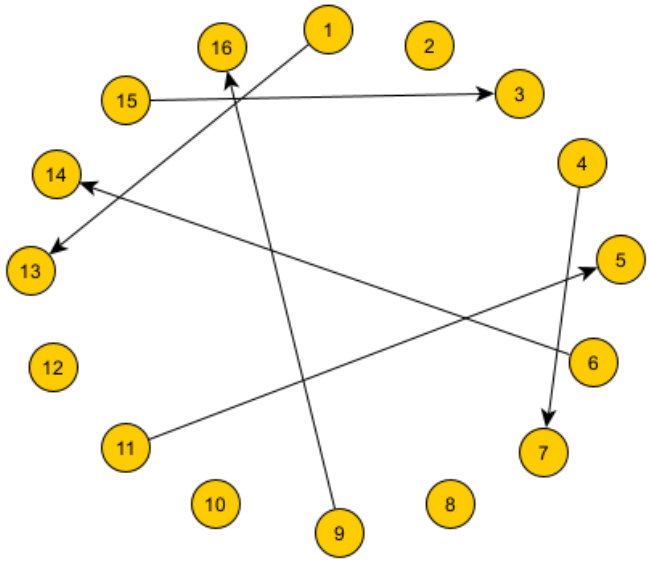


Рис. 2. Возможный вид орграфа паросочетаний с шестью дугами

Остальные дуги в орграфе отсутствуют, поскольку, например, в кластерной системе и в модели произошел случай, когда для свободных узлов-источников сообщений не нашлось адресуемых свободных узлов-приемников. Поэтому реализация орграфа паросочетаний, определенного выражением (2), может иметь изолированные вершины. Орграф паросочетаний на

рис. 3 не имеет изолированных вершин и имеет максимально возможное число дуг – 8.

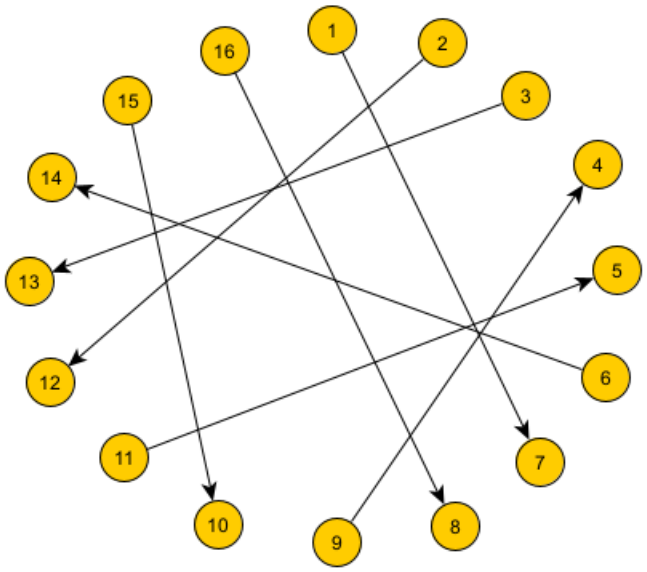


Рис. 3. Орграф паросочетаний с максимальным числом дуг

Продолжаем описание модели, описанной логико-вероятностным выражением (1). Все компоненты кортежа для графа H для этого примера были определены ранее. После добавления очередной дуги в данный граф путем модификаций предикатов $S_x(x) \leftarrow$ **true**, $S_y(y) \leftarrow$ **true**, $R(x, y) \leftarrow$ **true**, $Transfer(x, y) \leftarrow$ **true** выполняется модификация предиката $Delay(x, y) \leftarrow$ **true** – задержки на передачу информации и имитации проведения диалога между источником x и приемником y. В выражении (1) при $Transfer(x, y) \leftarrow$ **true** имитируется передача информации от узла-источника к узлу-приемнику, а при $Transfer(x, y) \leftarrow$ **false** передача заканчивается. По истечении времени задержки связь между источником x и приемником y разрывается, и соответствующая модификация графа паросочетаний выполняется путем удаления дуги (x, y) , где значения предметных переменных x, y известны:

$$S_x(x) \leftarrow \textbf{false}, S_y(y) \leftarrow \textbf{false}, R(x, y) \leftarrow \textbf{false}, Delay(x, y) \leftarrow \textbf{false},$$

$$Transfer(x, y) \leftarrow \textbf{false},$$

выполнение очередного этапа моделирования заканчивается. В случае, когда одновременный выбор вершин x и y ввиду их занятости невозможен, одна или обе условные части в квадратных скобках выражения (1) не могут быть выполнены успешно, выполняется один или оба оператора *Ret*, имитирующие отработку тайм-аута и передающие управление на повторные итерации выражения (1). В программной реализации происходит переход в состояние ожидания.

Выражение (1) позволяет строить на основе логико-вероятностного подхода имитационные модели большой размерности, в отличие от исполнимых автоматных моделей или моделей на базе сетей Петри. Например, на ос-

нове данного подхода проведено моделирование кластера «Круглый стол» с 8, 16, 24, 32, 64, 128, 256, 512 и 1024 пользовательскими компьютерами, и при выборках до 10000 прогонов модели время ожидания не превышало 10 с.

Примечание 1. Нотация α-дизъюнкции первоначально была предложена в системах алгоритмических алгебр для формульной записи алгоритмов [44, 51]: $[\alpha](a \vee b)$ – тернарная операция, зависящая от условия α и операторов a и b, используется в настоящей работе для всюду определенных условий. При α = **true** выполняется оператор a, а при α = **false** – оператор b. Возможны различные суперпозиции α-дизъюнкций, получаемые подстановкой новых операций α-дизъюнкции на место операторов a и b: при $a = [\alpha_2](c \vee d)$ и $b = [\alpha_3](e \vee f)$ результирующее выражение получит следующий вид:

$$[\alpha]([\alpha_2](c \vee d) \vee [\alpha_3](e \vee f)).$$

В настоящей работе используется модификация следующей суперпозиции операций α-дизъюнкций:

$$[\alpha_1]([\alpha_2](\{a_1, a_2, \ldots, a_k\} \vee Ret) \vee Ret)).$$

Модификация заключается в том, что вместо пустого оператора E использован оператор возврата с ожиданием *Ret* (от англ. *return*) к проверке с ожиданием истинности соответствующего ему α-условия в случае его первоначальной ложности. В фигурные скобки заключен блок совместимых операций $\{a_1, a_2, \ldots, a_k\}$, в данной работе это элементы своего рода семафорной техники – модификации предикатов, отмечающие происходящие события в моделируемой системе. Символом запятой в блоке обозначена операция конкатенации, или последовательной композиции, операторов, взятая из алгебры операторов системы алгоритмических алгебр [43, 51]. Квадратные, круглые и фигурные скобки являются элементами синтаксиса формул и их нельзя заменять или переносить произвольно. Данная нотация была ранее введена в работах [44, 45, 52].

Отметим, что использование нотации систем алгоритмических алгебр в принципе не является обязательным – возможно использование и других нотаций, позволяющих записывать алгоритмы моделирования в виде формул.

Примечание 2. В формульной записи моделей использованы операторы $\exists_{Any}$ и $\exists_{One}$ выбора кортежей из отношений, нотация и семантика для которых были использованы в работах [44, 45, 52]. Основная форма записи первого из этих операторов имеет следующий вид:

$$\exists_{Any}\langle x_1, x_2, \ldots, x_k\rangle R_1,$$

где $x_1, x_2, \ldots, x_k$ – элементы кортежа – предметные переменные, получающие значения выбранных предметных констант, возможно, входящих в какой-либо кортеж отношения R_1 – области истинности одноименного k-арного предиката. В выражении (1) и других выражениях для формульной записи логико-вероятностных и логико-алгебраических моделей этот оператор заключен в квадратные скобки $[\exists_{Any}\langle x_1, x_2, \ldots, x_k\rangle R_1]$, которые превращают данный оператор в условие, истинное при успешном завершении операции выбора произвольного кортежа и ложное в противном случае. При построе-

нии логико-вероятностных моделей такой оператор используется часто. Второй оператор вида $\exists_{One}\langle x_1, x_2, \ldots, x_k\rangle R_2$ отличается от первого тем, что в отношении R_2 выбранный кортеж должен был быть единственным.

1.4. Переход от имитационной логико-вероятностной модели к логико-алгебраической спецификации функциональной архитектуры вычислительного кластера

Программа имитационного моделирования, созданная на основании логико-вероятностного выражения (1), предназначена для выполнения на одном компьютере в многопоточном режиме. Однако на основе этого выражения путем несложного изменения можно получить выражение, пригодное для спецификации программного обеспечения реальной кластерной вычислительной системы. При этом нужно учесть, что программа должна выполняться в каждом компьютере, подключенном к коммутатору вычислительного кластера. Первый шаг трансформации заключается в том, что выражений, подобных выражению (1), должно быть m – по одному выражению для будущих программ на каждый компьютер кластера. Следующее выражение используется в качестве прототипа спецификации сетевого приложения для i-го компьютера кластера:

$$M_i = [Start(x_i)](\{Start(x_i) \leftarrow \textbf{false}, Master(x_i) \leftarrow \textbf{true}, Work^*(x_i) \leftarrow \textbf{true}\} \vee Ret),$$

$$([(\exists_{One}(x_i, y_j) \in V^2) \neg S_y(y_j) \;\&\; (x_i \neq y_j)]$$

$$(\{S_x(x_i) \leftarrow \textbf{true}, S_y(y_j) \leftarrow \textbf{true}, R(x_i, y_j) \leftarrow \textbf{true}, Transfer(x_i, y_j) \leftarrow \textbf{true},$$

$$Proc^*(x_i, y_j) \leftarrow \textbf{true}, S_x(x_i) \leftarrow \textbf{false}, S_y(y_j) \leftarrow \textbf{false}, R(x_i, y_j) \leftarrow \textbf{false},$$

$$Transfer(x_i, y_j) \leftarrow \textbf{true}, Proc^*(x_i, y_j) \leftarrow \textbf{false},$$

$$Master(x_i) \leftarrow \textbf{false}, Work^*(x_i) \leftarrow \textbf{false}\} \vee Ret), i, j = 1, 2, \ldots, m; i \neq j. \qquad (3)$$

От выражения (1) выражение (3) отличается введением индексов i и j, которым в реальном сетевом приложении соответствуют адреса компьютеров, подключенных к коммутатору. Задано новое имя выражения M_i (первая буква от имени *Master*, это означает, что передающий узел кластера получает статус мастера). Введены новые унарные предикаты *Start*, *Master*, *Work**, поновому интерпретируется бинарный предикат *Proc**. Звездочки при предикатных именах *Work** и *Proc** означают, что при реализации прототипного программного обеспечения кластера этим предикатам соответствуют вычислительные процедуры.

На основании логико-алгебраического выражения M_i строится прототипное программное обеспечение, которое устанавливается на каждый узел вычислительного кластера. При дальнейшем описании там, где это не вызывает противоречий, для упрощения формулировок будут использованы термины как от модели, так и от действующего кластера.

При $Start(x_i) = \textbf{true}$ начинается работа приложения-мастера на узле x_i кластера, которая отмечается модификациями предикатов $Master(x_i) \leftarrow \textbf{true}$ и $Work^*(x_i) \leftarrow \textbf{true}$. С этих событий начинается работа узла-передатчика, получившего статус мастера, т.е. инициатора последующих действий.

Выражение

$$[Start(x_i)](\{Master(x_i) \leftarrow \textbf{true}, Work^*(x_i) \leftarrow \textbf{true}\} \vee Ret) \quad (4)$$

имеет такой смысл: оператор *Ret* возвращает управление проверке условия $[Start(x_i)]$ при ложности высказывания $Start(x_i)$. От оставшейся части выражения (3) выражение (4) отделено символом запятой, означающим в данном контексте последовательное выполнение последующих действий. Следующая часть выражения начинается с оператора $(\exists_{One}(x_i, y_j) \in V^2)$ выбора единственного кортежа (x_i, y_j), где x_i соответствует узлу-передатчику, а y_j – узлу-приемнику кластера. Для реализации операции выбора необходимо, чтобы было истинно составное высказывание $\neg S_y(y_j)$ & $(x_i \neq y_j)$ – узел-приемник y_j кластера не занят, а x_i и y_j – разные узлы. Оператор $\exists_{One}(x_i, y_j)$ путем обращения к коммутатору при известном адресе порта компьютера-источника x_i определяет адрес порта компьютера-приемника y_j. Далее выполняются следующие модификации предикатов $S_x(x_i) \leftarrow \textbf{true}$, $S_y(y_j) \leftarrow \textbf{true}$, которым в реальной кластерной системе соответствуют события занятия узлов x_i и y_j. Модификации предиката $R(x_i, y_j) \leftarrow \textbf{true}$ соответствует событие связывания узлов x_i и y_j кластера при подготовке с совместным действиям при реализации диалога и обработке данных; модификации предиката $Proc^*(x_i, y_j) \leftarrow \textbf{true}$ соответствует выполнение операции обработки запроса. Оставшаяся часть выражения (3) описывает события, связанные завершением сеанса связи и с освобождением пары узлов кластера x_i и y_j.

В каждом i-м элементе кластера локально формируется реализация графа паросочетаний H_i, который включает не более одной дуги (x_i, y_j):

$$H_i = (V_{xi}, V_{yj}, S_{xi}, S_{yj}, R_i); i, j = 1, 2, \ldots, m; i \neq j. \quad (5)$$

Завершая описания событий в вычислительном кластере, отметим, что информация, указанная в выражении (3) и затрагивающая узел-приемник y_j, передается в данный узел от узла-мастера x_i при помощи управляющих сообщений. Детали формирования и передачи подобных сообщений в сетевом приложении для кластера просты и не затронуты в формализации.

2. Результаты макроанализа производительности вычислительного кластера на основе логико-вероятностной имитационной модели, задающей равнодоступные парные взаимодействия (модель «Круглый стол»)

2.1. Результаты макроанализа имитационной модели «Круглый стол» в основном режиме

Для режима «Круглый стол» достаточно сложно получить оценки вероятностных характеристик для распределений вероятностей времени подготовки запросов и времени обработки в реальной системе, поскольку нагрузка зависит от множества решаемых задач, времени суток и др. Экспоненциальное распределение предполагает значительную вариабельность переменной и по сравнению с многими распространенными в теории массового обслуживания систем распределений имеет большую дисперсию [38]. Для сравнительного макроанализа кластерных вычислительных систем такие предпосылки

приемлемы, так как они обычно позволяют имитировать работу системы в более жестких условиях. Следует сказать, что при статистическом моделировании можно сменять распределения вероятностей – от произвольных распределений, задаваемых при помощи кусочно-линейных аппроксимаций, до многих других, применяемых при моделировании сложных систем.

Выражения (1), (3), (4) и (5) определяют основную концептуальную схему для разработки имитационной поведенческой модели функционирования вычислительного кластера, реализующего равнодоступные парные взаимодействия (модель «Круглый стол»).

При моделировании далее используются псевдослучайные величины, заданные смещенным экспоненциальным распределением, для которого функция плотности вероятности $f(t)$ имеет следующий вид [38, 53]:

$$f(t) = \lambda \exp(-\lambda (t - t_0) \text{ при } t \geq t_0 \text{ и } 0 \text{ при } t < t_0,$$

где t_0 – параметр смещения; в программе такие псевдослучайные величины генерируются при помощи функции ($expon(i, t_0, t_{ср})$), где среднее время $t_{ср}$ между событиями в потоке определяется равенством $t_{ср} = 1/\lambda$. Такая функция широко известна в теории вероятностей и массового обслуживания и нередко используется при статистическом моделировании систем, когда приходится оперировать со псевдослучайными величинами, минимальное значение которых ограничено некоторой постоянной величиной t_0 [54]. Функция распределения такой величины имеет следующий вид:

$$F(t) = 1 - \exp(-\lambda (t - t_0)).$$

При проведении экспериментов с имитационной моделью число пар $N \times N$ изменялось от 2 × 2 до 1024 × 1024. Запись вида $N \times N$ означает, что каждым из N узлов кластера организуются случайные парные обмены («беседы») с другими узлами из этого же множества узлов с учетом перечисленных ранее ограничений, связанными к конфликтами и образованием очередей. Результаты сведены в табл. 1.

Таблица 1

Результаты экспериментальных исследований имитационной модели кластера, работающего в режиме «Круглый стол»

$N_{п/п}$	Число пар $N \times N$	T_{Pair}, мс	T_1, мс	K_{Slow}
1	2×2	411	220	1,87
2	4×4	614	220	2,8
3	8×8	803	220	3,65
4	16×16	980	220	4,45
5	24×24	1077	220	4,9
6	32×32	1150	220	5,22
7	48×48	1240	220	5,63
8	64×64	1300	220	5,9
9	128×128	1449	220	6,59
10	256×256	1595	220	7,25
11	512×512	1732	220	7,87
12	1024×1024	1870	220	8,5

Предполагалось, что время подготовки запроса («обдумывания темы беседы») распределено в соответствии со смещенным экспоненциальным распределением с параметрами $t_{ср}$ = 100 мс, t_0 = 10 мс. Время обработки запроса определено теми же параметрами. В результате моделирования определялось значение главного параметра – времени T_{Pair} обслуживания запроса, включающего время его подготовки. Значения этого времени, оцененные по результатам статистического эксперимента, указаны в 3-й колонке табл. 1. Интерес также представляло соотношение между реальным временем T_{Pair} и «идеальным» T_1, которое было бы равно $2(t_{ср} + t_0)$ = 220 мс в случаях, когда граф паросочетаний имел бы N дуг при бесконфликтной ситуации. Коэффициент замедления обслуживания из-за взаимного влияния запросов равен $K_{Slow} = T_{Pair} / T_1$. При учете этого коэффициента возможно оценить работу кластера при наличии случайных парных взаимодействий. Рисунки 4 и 5 наглядно иллюстрируют влияние зависимости запросов друг от друга при наличии конфликтов и очередей. Моделирование производилось при изменении N = 2, 4, 8, 16, 24, 32, 48, 64, 128, 256, 512, 1024; число портов N_P коммутатора равно N, объем выборки составлял 10 000. Подтверждено, что благодаря организации очередей в коммутаторе кластерная система в режиме «Круглый стол» может работать без тупиков.

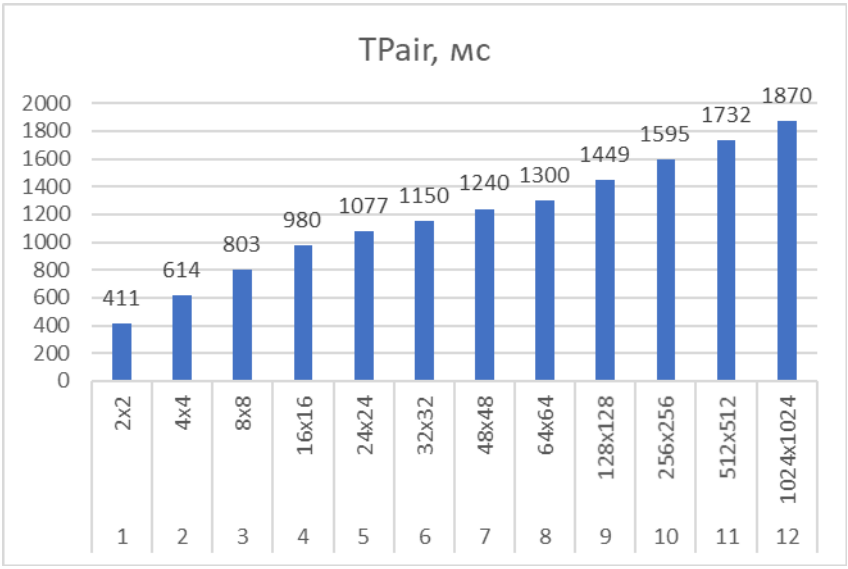


Рис. 4. Зависимость среднего времени обработки запроса T_{Pair} от числа узлов N в вычислительном кластере

На рис. 6 приведены гистограммы распределения времени обработки запросов T_{Pair}. Слева от каждой гистограммы указано число узлов. Гистограммы, как графические представления данных, позволяют наглядно визуализировать распределение частот величины времени обработки запроса. Из рисунков видно, что гистограммы имеют длинные «хвосты», это свидетельствует о наличии выбросов или экстремальных значений времени обработки запроса. Однако эти отклонения достаточно редки и не оказывают значительного влияния на работу вычислительного кластера в режиме «Круглый стол».

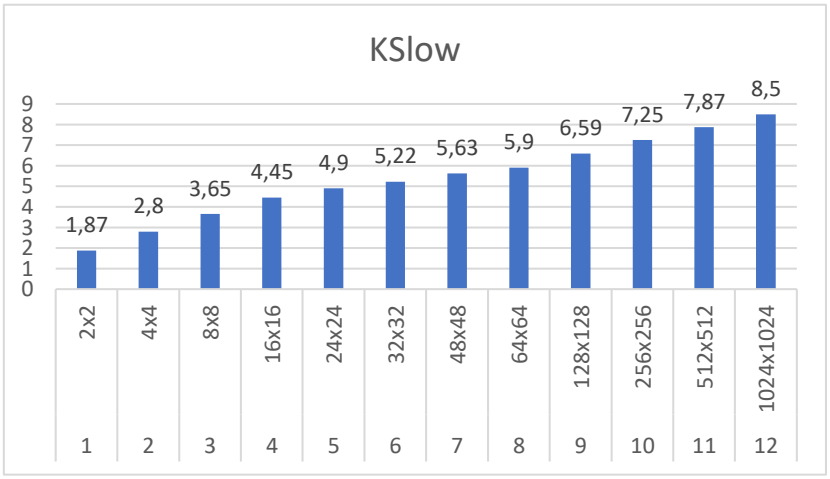
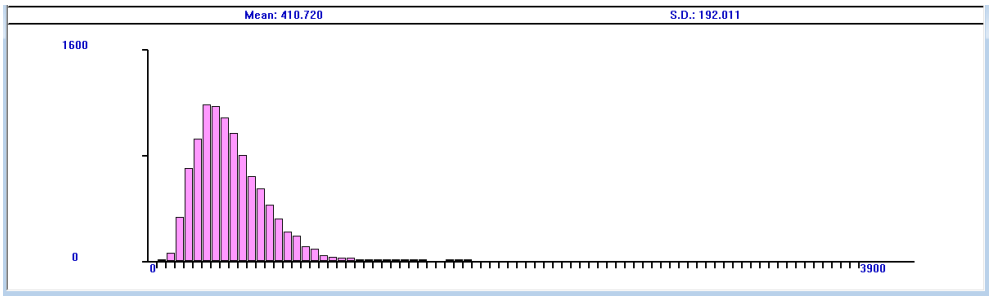
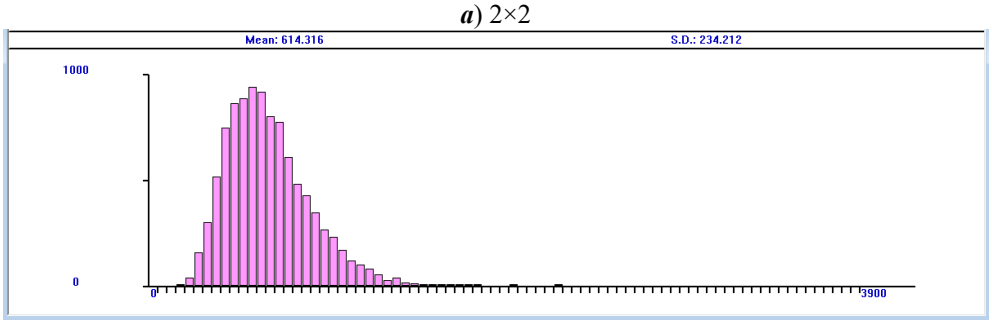


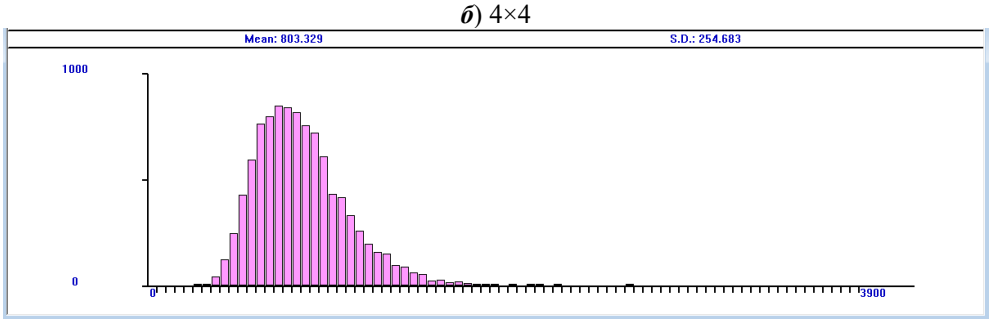
Рис. 5. Зависимость коэффициента замедления обработки запроса K_{Slow} от числа узлов N в вычислительном кластере



а) 2×2

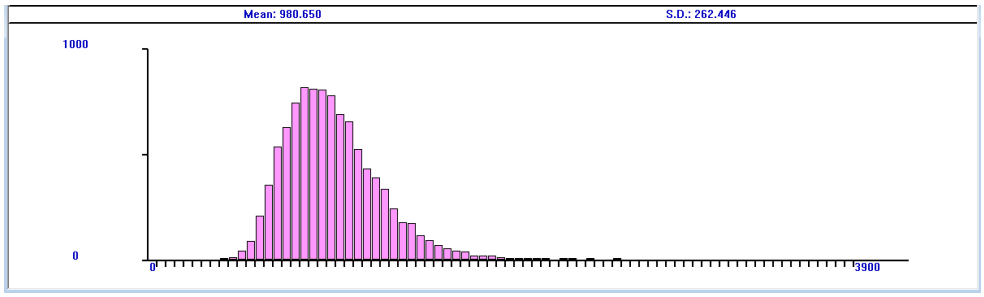


б) 4×4

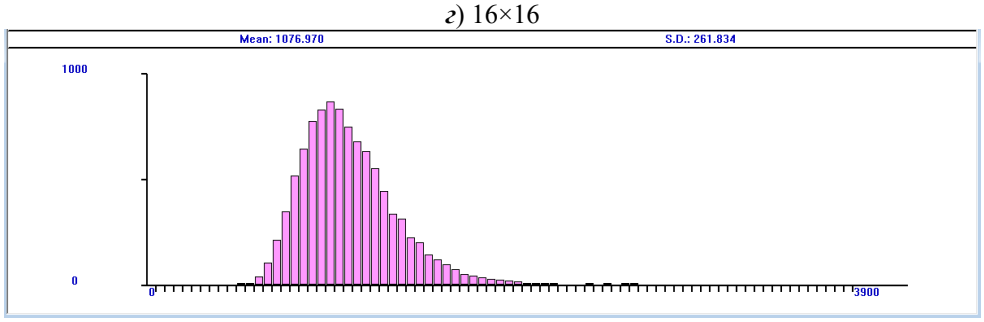


в) 8×8

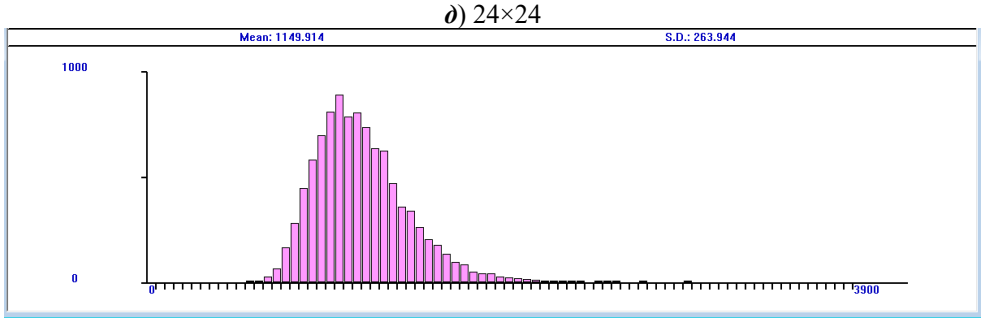
Рис. 6. Гистограммы распределения времени обработки запросов T_{Pair}



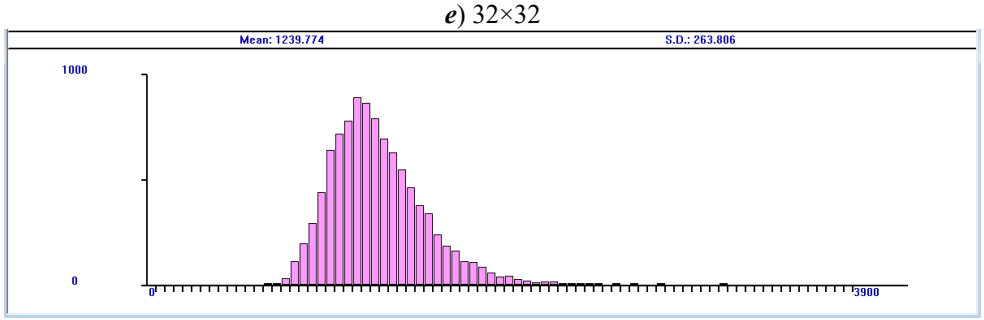
г) 16×16



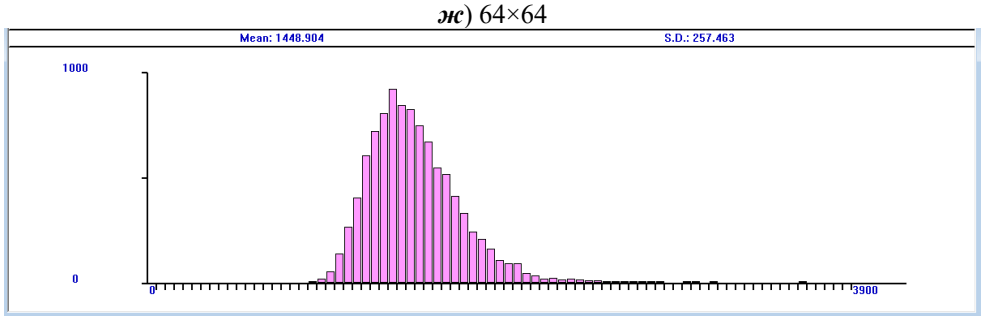
д) 24×24



е) 32×32

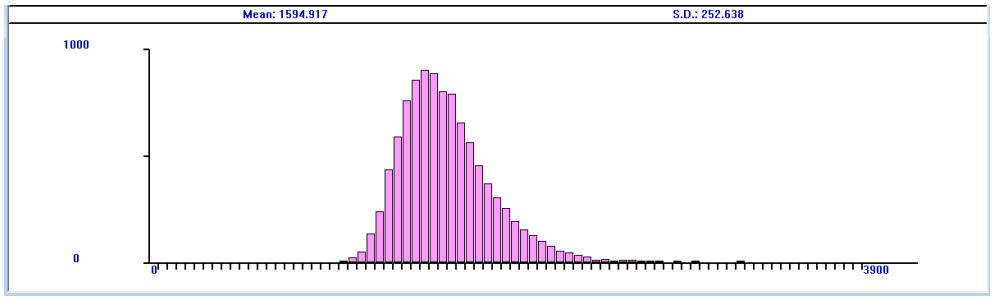


ж) 64×64

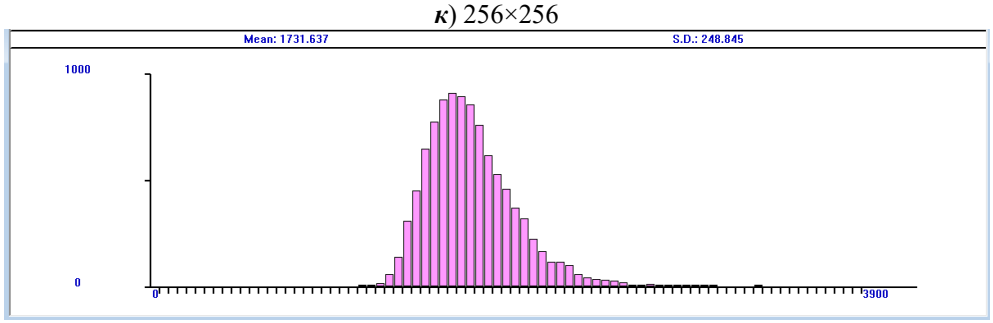


и) 128×128

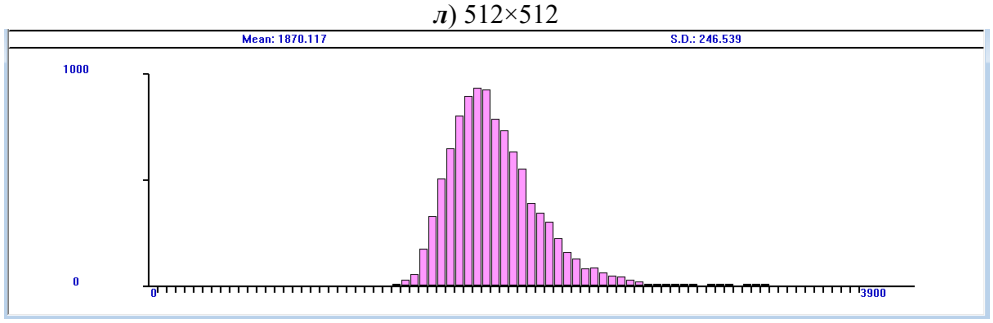
Рис. 6. Продолжение



к) 256×256



л) 512×512



м) 1024×1024

Рис. 6. Окончание

2.2. Результаты макроанализа имитационной модели «Круглый стол» при ограничении числа портов коммутатора локальной сети

Следующий этап макроанализа функционирования вычислительного кластера по построенной имитационной логико-вероятностной модели будет связан с выбором числа портов коммутатора. Для этого выберем $N = 1024$ и проведем эксперименты с моделью при изменяющемся числе портов N_P. Результаты испытаний сведены в табл. 2. Как и ранее, запись вида $N \times N$ означает, что каждым из N узлов кластера организуются случайные парные обмены с другими узлами из этого же множества.

Как и следовало ожидать, с убыванием числа портов коммутатора кластера среднее время T_{Pair} подготовки и обработки запросов увеличивается. При числе портов 256, 512, 1024 значения T_{Pair} сохраняются почти равными, при этом максимальное число занятых портов приблизительно равно максимальному числу «беседующих» пар (≈200). При дальнейшем уменьшении числа портов N_P время T_{Pair} увеличивается. Таким образом, пользователь кластера на основании таблиц 1 и 2 может оценить время T_{Pair} подготовки и обработки запросов и выбрать необходимое число портов коммутатора класте-

ра. Рисунки 7 и 8 наглядно иллюстрируют влияние числа портов N_P на важнейшую характеристику производительности T_{Pair}.

Таблица 2

Результаты экспериментальных исследований
имитационной модели кластера, работающего в режиме
«Круглый стол», при изменении числа портов коммутатора

$N_{п/п}$	Число пар $N \times N$	Кол-во портов N_P	T_{Pair}, мс	K_{Lim}
1	1024×1024	1024	1870	1
2	1024×1024	512	1873	1
3	1024×1024	256	1873	1
4	1024×1024	128	1905	1,02
5	1024×1024	64	2440	1,21
6	1024×1024	32	4017	2,15
7	1024×1024	24	5130	2,74
8	1024×1024	16	7404	3,96
9	1024×1024	8	14339	7,67

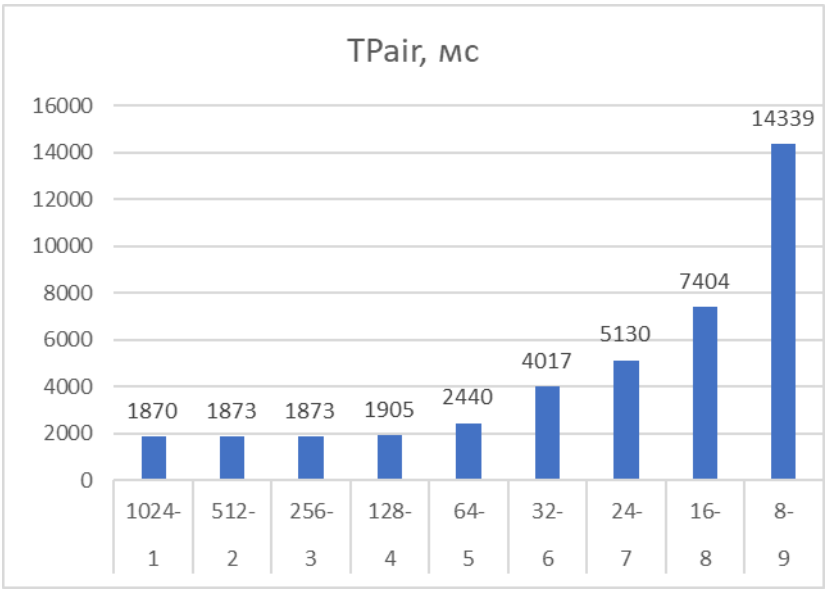


Рис. 7. Зависимость среднего времени обработки запроса T_{Pair}
от числа N_P портов коммутатора при числе пар $N \times N = 1024 \times 1024$

Значения коэффициента K_{Lim} вычисляются как отношения времени обработки T_{Pair} при $N_p < 1024$ к наименьшему значению этого же параметра при $N_p = 1024$. Например, $K_{Lim,8} = T_{Pair,8}/T_{Pair,1024} = 14339/1870 = 7,67$.

На рис. 9 представлены гистограммы распределения времени T_{Pair} при $N = 1024$ и различных значениях числа портов N_p коммутатора COM. Из рисунков, как и в предыдущем случае, видно, что гистограммы имеют длинные «хвосты», что свидетельствует о наличии выбросов или экстремальных значений времени обработки запроса. Однако эти отклонения достаточно редки и не оказывают значительного влияния на работу вычислительного кластера в режиме «Круглый стол».

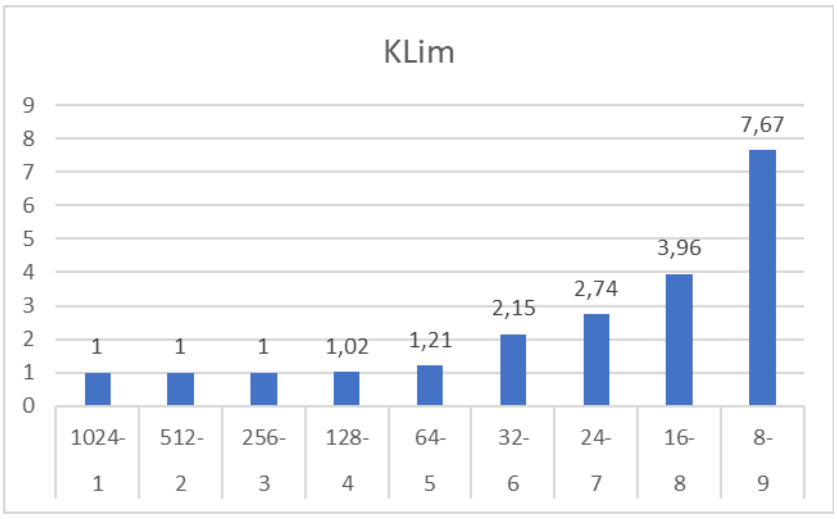
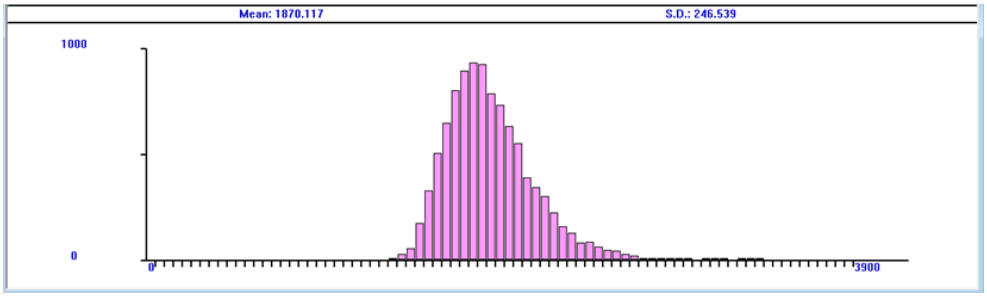
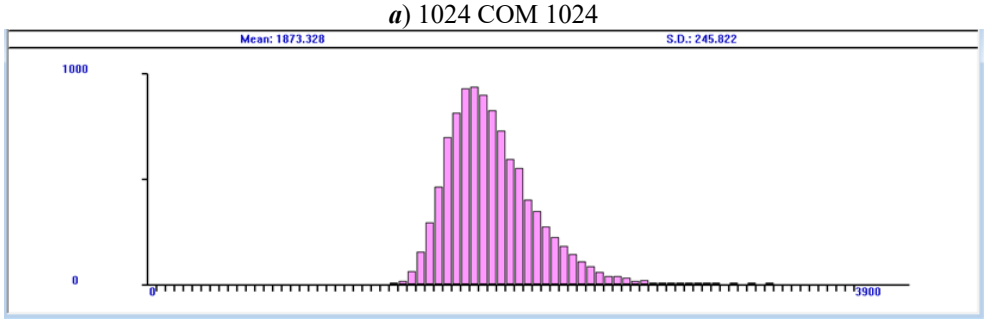


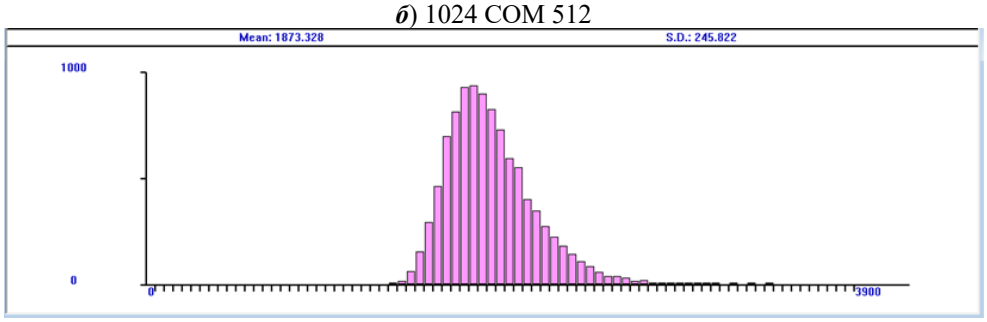
Рис. 8. Зависимость коэффициента замедления обработки запроса K_{Lim} от числа N_P портов коммутатора при числе пар $N \times N = 1024 \times 1024$



а) 1024 COM 1024

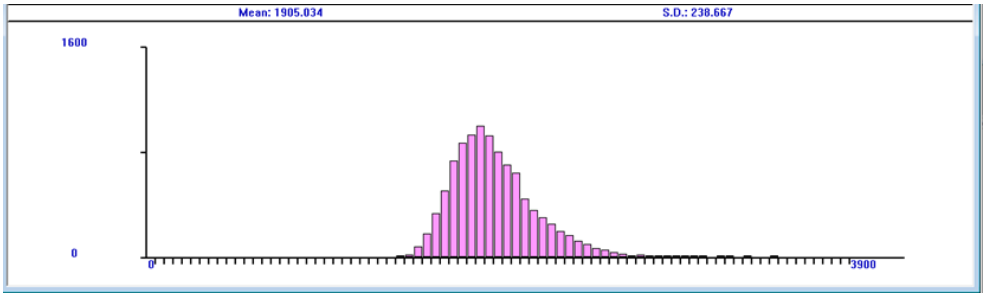


б) 1024 COM 512

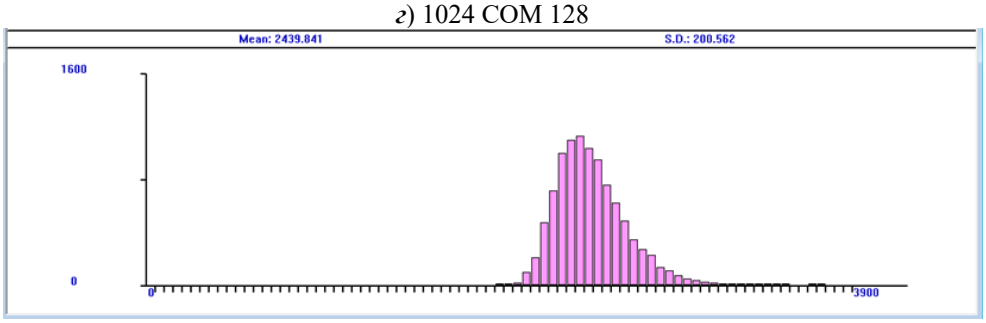


в) 1024 COM 256

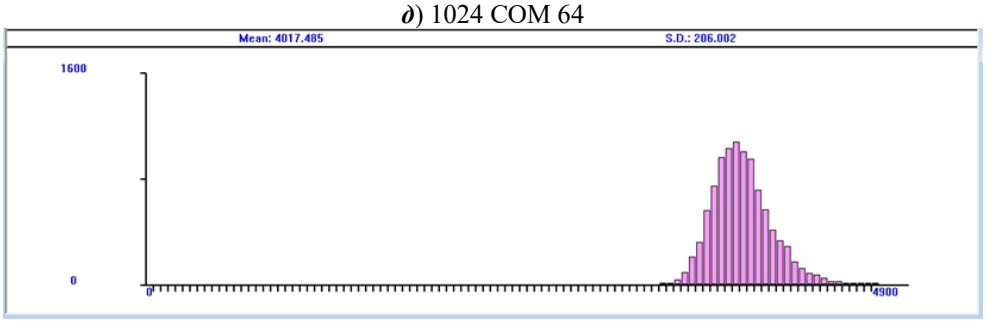
Рис. 9. Гистограммы распределения времени T_{Pair}



г) 1024 COM 128



д) 1024 COM 64



е) 1024 COM 32

Рис. 9. Окончание

2.3. Логико-вероятностная модель «Резидент – агенты»: последовательная раздача заданий – последовательное выполнение

Пусть теперь предметная константа x_0 представляет в модели ведущий компьютер кластера, условно названный «резидентом», а множество $V = \{x_1, x_2, \ldots, x_m\}$ представляет агенты – «рядовые» машины кластера. Обозначим через $Q = \{x_0\} \times V$ множество всех пар вида (x_0, x_i), $i = 1, 2, \ldots, m$, тогда модификации бинарного предиката $R(x_0, x_i) \leftarrow \mathbf{true}$ в модели соответствует включение дуги (x_0, x_i) в граф C устанавливаемых межузловых связей в кластере, а модификации $R(x_0, x_i) \leftarrow \mathbf{false}$ соответствует удаление дуги (x_0, x_i). Выборку кортежа (x_0, x_i) из отношения Q осуществляет оператор $\exists_{One}(x_0, x_i) \in Q$. Остальные предикаты в выражении (6) имеют такой же смысл, как и в определениях моделей (1) и (3). Модификации унарного предиката *Resident* соответствует инициирование и завершение работы ведущего компьютера кластера:

$$M_0 = [Start(x_0)](\{Resident(x_0) \leftarrow \mathbf{true}, Work^*(x_0) \leftarrow \mathbf{true}\} \vee Ret),$$

$$SeqReplicate(i = 1..m)\text{: } ([(\exists_{One}(x_0, x_i) \in Q) \neg S_x(x_i)]$$
$$(\{S_x(x_i) \leftarrow \textbf{true}, R(x_0, x_i) \leftarrow \textbf{true}, Proc^*(x_0, x_i) \leftarrow \textbf{true},$$
$$S_x(x_i) \leftarrow \textbf{false}, R(x_0, x_i) \leftarrow \textbf{false}, Proc^*(x_0, x_i) \leftarrow \textbf{false},$$
$$Work^*(x_i) \leftarrow \textbf{false}\} \vee Ret)), Resident(x_0) \leftarrow \textbf{false}). \quad (6)$$

Отличительной особенностью формулы (6) является использование оператора репликации $SeqReplicate(i = 1..m)$, которая в формуле означает последовательную репликацию (повторение) записи формулы, заключенной в скобки слева, с последовательной подстановкой индексированных переменных:

$$([(\exists_{One}(x_0, x_i) \in Q) \neg S_x(x_i)]$$
$$(\{S_x(x_i) \leftarrow \textbf{true}, R(x_0, x_i) \leftarrow \textbf{true}, Proc^*(x_0, x_i) \leftarrow \textbf{true},$$
$$S_x(x_i) \leftarrow \textbf{false}, R(x_0, x_i) \leftarrow \textbf{false}, Proc^*(x_0, x_i) \leftarrow \textbf{false},$$
$$Work^*(x_i) \leftarrow \textbf{false}\} \vee Ret)). \quad (7)$$

В прототипной программе, управляющей согласно формуле (6) работой всего кластера, реплицируемой части данной формулы соответствует исполняемая циклически часть программы.

В выражении (6) считалось, что рабочие программы элементов кластера уже загружены. Однако во многих случаях требуется выполнение первоначальной загрузки, проводимой последовательно для всех элементов или параллельно, одновременно для всех элементов кластера.

При последовательной загрузке элементов кластера и логико-алгебраическом и последовательном выполнении рабочих программ выражение M_{SeqSeq} будет иметь следующий вид:

$$M_{SeqSeq} = [Start(x_0)](\{ Start(x_0) \leftarrow \textbf{false}, Resident(x_0) \leftarrow \textbf{true},$$
$$Work^*(x_0) \leftarrow \textbf{true}\} \vee Ret),$$
$$SeqReplicate(i = 1..m)\text{: } ([(\exists_{One}(x_0, x_i) \in Q) \neg S_x(x_i)]$$
$$(\{S_x(x_i) \leftarrow \textbf{true}, R(x_0, x_i) \leftarrow \textbf{true}, Load^*(x_0, x_i) \leftarrow \textbf{true},$$
$$S_x(x_i) \leftarrow \textbf{false}, R(x_0, x_i) \leftarrow \textbf{false}, Load^*(x_0, x_i) \leftarrow \textbf{false},$$
$$Work^*(x_i) \leftarrow \textbf{false}\} \vee Ret)), Resident(x_0) \leftarrow \textbf{false}),$$
$$[Start(x_0)](\{Start(x_0) \leftarrow \textbf{false}, Resident(x_0) \leftarrow \textbf{true},$$
$$Work^*(x_0) \leftarrow \textbf{true}, Start(x_0) \leftarrow \textbf{true}\} \vee Ret),$$
$$SeqReplicate(i = 1..m)\text{: } ([(\exists_{One}(x_0, x_i) \in Q) \neg S_x(x_i)]$$
$$(\{Start(x_0) \leftarrow \textbf{false}, S_x(x_i) \leftarrow \textbf{true}, R(x_0, x_i) \leftarrow \textbf{true},$$
$$Proc^*(x_0, x_i) \leftarrow \textbf{true},$$
$$S_x(x_i) \leftarrow \textbf{false}, R(x_0, x_i) \leftarrow \textbf{false}, Proc^*(x_0, x_i) \leftarrow \textbf{false},$$
$$Work^*(x_i) \leftarrow \textbf{false}\} \vee Ret)), Resident(x_0) \leftarrow \textbf{false}). \quad (8)$$

Здесь бинарный предикат *Load* моделирует процедуру начальной загрузки каждого элемента кластера.

2.4. Построение логико-вероятностных моделей функционирования вычислительного кластера, реализующего параллельный доступ «Один ко многим» (модели «Собрание 1» и «Собрание 2»)

Следующий важный режим связан с параллельной обработкой информации в кластерной вычислительной системе. **В режиме «Собрание 1»** ведущий компьютер кластера подготавливает данные, а затем передает их для последующей параллельной обработки в ведомых компьютерах. Метафорическое название новой модели – «Собрание 1», где «председатель» (*Chairman*) подготавливает и раздает последовательно задания участникам (режим репликации – *SeqReplicate*($i = 1..m$)). Новая логико-алгебраическая модель $C_{Seq,Par}$ (9) построена путем переинтерпретации предыдущей модели (8) и изменения режима выполнения загруженных программ кластером с последовательного на параллельный (но при последовательной загрузке программ и данных на каждый элемент кластера):

$$
\begin{gathered}
C_{Seq,Par} = [Start(x_0)](\{\, Start(x_0) \leftarrow \textbf{false}, Chairman(x_0) \leftarrow \textbf{true}, \\
Work^*(x_0) \leftarrow \textbf{true}\} \vee Ret), \\
SeqReplicate(i = 1..m): ([(\exists_{One}(x_0, x_i) \in Q) \neg S_x(x_i)] \\
(\{S_x(x_i) \leftarrow \textbf{true}, R(x_0, x_i) \leftarrow \textbf{true}, Load^*(x_0, x_i) \leftarrow \textbf{true}, \\
S_x(x_i) \leftarrow \textbf{false}, R(x_0, x_i) \leftarrow \textbf{false}, Load^*(x_0, x_i) \leftarrow \textbf{false}, \\
Work^*(x_i) \leftarrow \textbf{false}\} \vee Ret)), Chairman(x_0) \leftarrow \textbf{false}), \\
[Start(x_0)](\{Start(x_0) \leftarrow \textbf{false}, Chairman(x_0) \leftarrow \textbf{true}, \\
Work^*(x_0) \leftarrow \textbf{true}, Start(x_0) \leftarrow \textbf{true}\} \vee Ret), \\
ParReplicate(i = 1..m): ([(\exists_{One}(x_0, x_i) \in Q) \neg S_x(x_i)] \\
(\{Start(x_0) \leftarrow \textbf{false}, S_x(x_i) \leftarrow \textbf{true}, R(x_0, x_i) \leftarrow \textbf{true}, \\
Proc^*(x_0, x_i) \leftarrow \textbf{true}, \\
S_x(x_i) \leftarrow \textbf{false}, R(x_0, x_i) \leftarrow \textbf{false}, Proc^*(x_0, x_i) \leftarrow \textbf{false}, \\
Work^*(x_i) \leftarrow \textbf{false}\} \vee Ret)), Chairman(x_0) \leftarrow \textbf{false}).
\end{gathered} \tag{9}
$$

Здесь *ParReplicate*($i = 1..m$) – оператор параллельной репликации выражения, записанного справа от него в круглых скобках, что представлено в рабочей программе репликацией частей программы, соответствующей каждому выражению. Таких частей будет m, по одной на каждый ведомый компьютер. Поскольку полагается, что все реплики для выражения (9) независимы друг от друга, то все отношения, используемые в этом выражении, сегментируются, и каждый сегмент входит в реплику, т.е. затем в реализованное программно выражение. Например, при $i = 2$ в программу для 2-го компьютера кластера входят кортежи $< x_2 >$ и $< x_0, x_2 >$. Передача и сбор результатов вычислений от каждого из m ведомых компьютеров сопровождается событи-

ем модификации предиката $Chairman(x_0) \leftarrow \mathbf{false}$ в ведущем компьютере кластера.

Режим работы «Собрание 2». При возможной реализации параллельной первоначальной загрузки рабочих программ в элементы кластера в параллельном широковещательном режиме логико-алгебраическая модель $C_{Seq,Par}$ (9) преобразуется к следующему виду:

$$C_{Par,Par} = [Start(x_0)](\{ Start(x_0) \leftarrow \mathbf{false}, Chairman(x_0) \leftarrow \mathbf{true},$$
$$Work^*(x_0) \leftarrow \mathbf{true}\} \vee Ret),$$
$$ParReplicate(i = 1..m): ([(\exists_{One}(x_0, x_i) \in Q) \neg S_x(x_i)]$$
$$(\{S_x(x_i) \leftarrow \mathbf{true}, R(x_0, x_i) \leftarrow \mathbf{true}, Load^*(x_0, x_i) \leftarrow \mathbf{true},$$
$$S_x(x_i) \leftarrow \mathbf{false}, R(x_0, x_i) \leftarrow \mathbf{false}, Load^*(x_0, x_i) \leftarrow \mathbf{false},$$
$$Work^*(x_i) \leftarrow \mathbf{false}\} \vee Ret)), Chairman(x_0) \leftarrow \mathbf{false}),$$
$$[Start(x_0)](\{Start(x_0) \leftarrow \mathbf{false}, Chairman(x_0) \leftarrow \mathbf{true},$$
$$Work^*(x_0) \leftarrow \mathbf{true}, Start(x_0) \leftarrow \mathbf{true}\} \vee Ret),$$
$$ParReplicate(i = 1..m): ([(\exists_{One}(x_0, x_i) \in Q) \neg S_x(x_i)]$$
$$(\{Start(x_0) \leftarrow \mathbf{false}, S_x(x_i) \leftarrow \mathbf{true}, R(x_0, x_i) \leftarrow \mathbf{true},$$
$$Proc^*(x_0, x_i) \leftarrow \mathbf{true},$$
$$S_x(x_i) \leftarrow \mathbf{false}, R(x_0, x_i) \leftarrow \mathbf{false}, Proc^*(x_0, x_i) \leftarrow \mathbf{false},$$
$$Work^*(x_i) \leftarrow \mathbf{false}\} \vee Ret)), Chairman(x_0) \leftarrow \mathbf{false}). \quad (10)$$

Выражения (6)–(10) возможно отнести к классу логико-алгебраических спецификаций к программному обеспечению уровня *middleware* кластерной системы. Эту же роль, но на начальных этапах работ по организации функционирования кластера, выполняют логико-вероятностные модели. По завершении этапа формализации моделей вычислительных кластеров необходимо перейти собственно к этапу макроанализа их производительности на основе проведения статистических экспериментов.

2.5. Результаты статистического моделирования к задачам организации функциональной архитектуры вычислительного кластера, работающего в режимах «Резидент – агенты» и двух вариантах режима «Собрание»

На основе логико-вероятностных и логико-алгебраических моделей построены имитационные поведенческие модели, позволяющие получить главные характеристики производительности – оценки среднего времени выполнения запроса, гистограммы распределения времени выполнения запроса. Определены следующие оценки математического ожидания времени выполнения запросов в вычислительном кластере:

T_{seq} – оценка математического ожидания времени выполнения запроса в режиме последовательной загрузки всех узлов кластера и дальнейшей последовательной же обработки запросов (режим «Резидент-агенты»);

T_{sp} – оценка математического ожидания времени выполнения запросов в режиме последовательной загрузки всех узлов кластера и дальнейшей параллельной обработки запросов всеми узлами кластера (режим «Собрание 1»);

T_{pp} – оценка математического ожидания времени выполнения запроса в режиме параллельной загрузки всех узлов кластера и дальнейшей параллельной обработки запросов всеми узлами кластера (режим «Собрание 2»).

Время загрузки программ в узлы кластера выбрано равным 10 мс; время обработки запроса задается смещенным экспоненциальным распределением [38, 54]:

$$f(t) = \lambda\, exp(-\lambda\,(t - t_0) \text{ при } t \geq t_0 \text{ и } 0 \text{ при } t < t_0,$$

с параметрами $t_{\text{ср}} = 500$ мс, $t_0 = 100$ мс.

В табл. 3 представлены результаты статистических экспериментов с имитационными моделями вычислительного кластера, работающего в режимах «Резидент – агенты», «Собрание 1» и «Собрание 2». Кроме указанных выше оценок математических ожиданий времен обработки запросов (при учете времени загрузки обрабатывающих программ и данных) T_{seq}, T_{sp} и T_{pp}, приведены значения коэффициентов относительного выигрыша от организации параллельных режимов загрузки и обработки заданий в кластере:

$$K_{seq/sp} = T_{seq} / T_{sp},$$

$$K_{seq/pp} = T_{seq} / T_{pp},$$

$$K_{sp/pp} = T_{sp} / T_{pp}.$$

Таблица 3

Результаты экспериментальных исследований имитационной модели кластера при работе в режимах «Резидент – агенты», «Собрание 1» и «Собрание 2»

Число узлов	Оценка среднего времени выполнения задания в кластере из N узлов			Относительный выигрыш по времени выполнения задания		
	T_{seq}	T_{sp}	T_{pp}	$K_{seq/sp}$	$K_{seq/pp}$	$K_{sp/pp}$
1	613	615	617	1,0	1,0	1,0
2	1227	881	871	1,39	1,41	1,01
4	2439	1186	1156	2,06	2,12	1,02
8	4873	1543	1473	3,16	3,31	1,05
16	9768	1950	1796	5,0	5,50	1,09
32	19516	2451	2141	7,96	9,11	1,14
64	39021	3114	2484	12,5	15,7	1,25
128	78056	4104	2835	19,0	27,5	1,45
256	156059	5722	3172	27,3	49,2	1,82
512	312174	8632	3522	36,2	88,6	2,45
1024	624333	14090	3860	44,3	161,7	3,65

Проведенный макроанализ производительности показывает, что при реализации указанных режимов настоятельно необходимо организовать параллельную работу как при загрузке программ и данных на узлы кластера, так и, особенно, при обработке запросов. Диаграммы, показывающие зависимость коэффициентов относительного выигрыша при переходе к параллельным режимам работы, приведены на рис. 10–12. Необходимость перехода

к параллельным режимам работы узлов кластера особенно важна при большом числе его узлов.

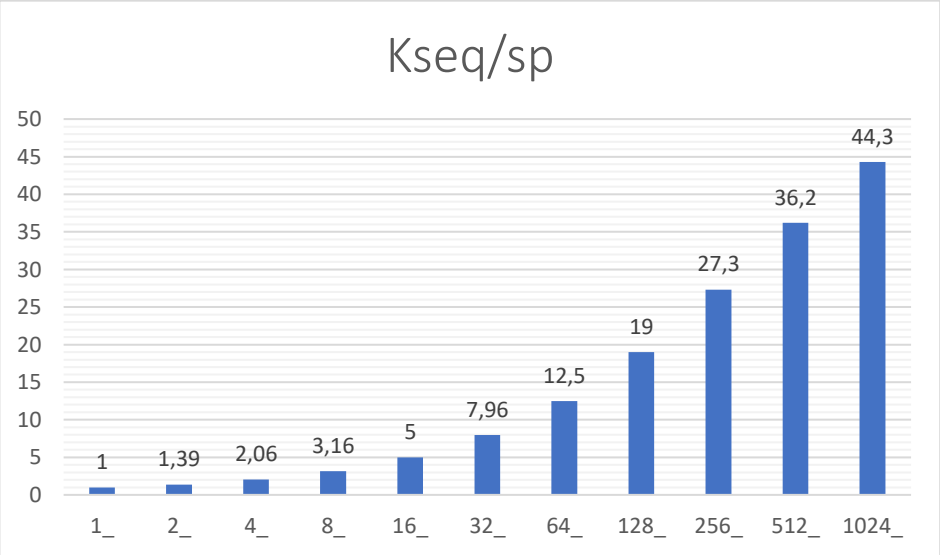


Рис. 10. Зависимость относительного выигрыша $K_{seq/sp}$ по времени подготовки и обработки запросов при переходе от последовательной загрузки и последовательной обработки запросов к последовательной загрузке и параллельной обработке запросов

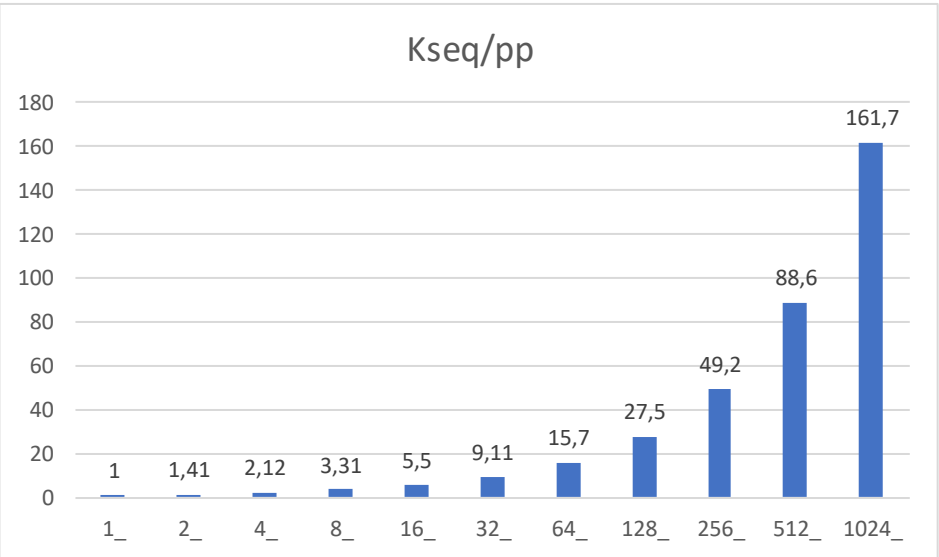


Рис. 11. Зависимость относительного выигрыша $K_{seq/pp}$ по времени подготовки и обработки запросов при переходе от последовательной загрузки и последовательной обработки запросов к параллельной загрузке и параллельной обработке запросов

На рис. 13 представлены гистограммы для времени обработки заданий в предметно-ориентированной кластерной вычислительной системе при различных режимах использования.

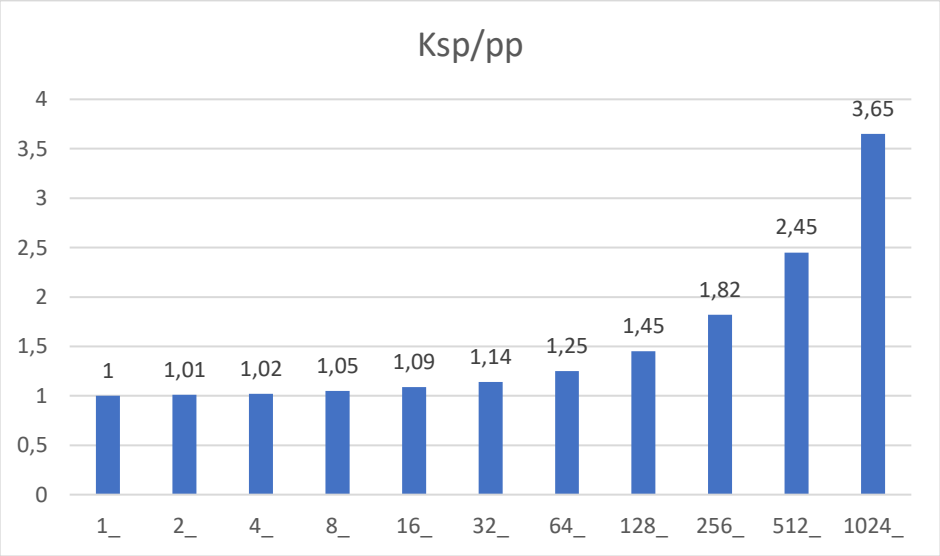
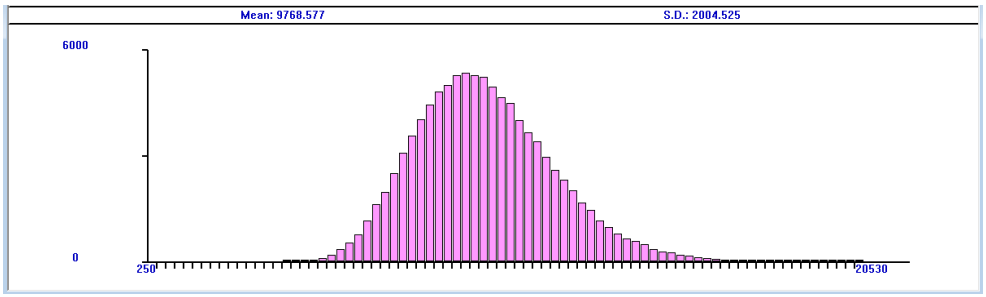
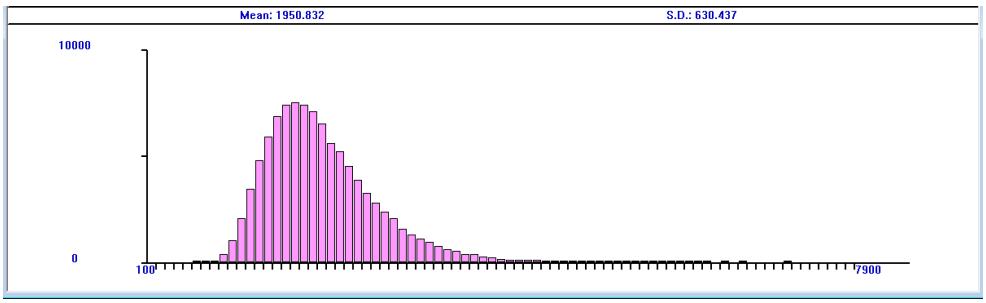


Рис. 12. Зависимость относительного выигрыша $K_{sp/pp}$
по времени подготовки и обработки запросов при переходе
от последовательной загрузки и параллельной обработки запросов
к параллельной загрузке и параллельной обработке запросов

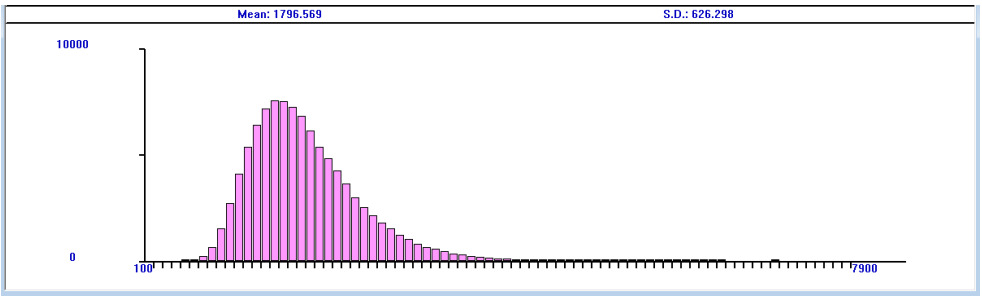


а) кластер из 16 машин; режим работы «seq» последовательной
загрузки и последовательного выполнения заданий;
цена деления – 250 мс; объем выборки – 100 000



б) кластер из 16 машин; режим работы «sp» последовательной
загрузки и параллельного выполнения заданий;
цена деления – 100 мс; объем выборки – 100 000

Рис. 11. Гистограммы для времени обработки заданий
в предметно-ориентированной кластерной вычислительной системе
при различных режимах использования



в) кластер из 16 машин; режим работы «pp» параллельной загрузки и параллельного выполнения заданий; цена деления – 100 мс; объем выборки – 100 000

Рис. 11. Окончание

Обращают на себя внимание длинные «хвосты» распределений времени обслуживания запросов, что, как и в предыдущих случаях, свидетельствует о наличии выбросов или экстремальных значений времени обработки запроса. Однако эти отклонения достаточно редки и не оказывают значительного влияния на работу вычислительного кластера в режимах «Резидент – агенты», «Собрание 1» и «Собрание 2».

2.6. Рекомендации по организации работы кластера при загрузке и выполнении рабочих программ и данных

При работе кластера в режиме «Круглый стол» реализуется диалоговое взаимодействие пар узлов, необходимость в широковещании отсутствует, поэтому нет необходимости в дополнительной информации к описанию. Работа вычислительного кластера в режимах «Резидент – агенты», «Собрание 1» и «Собрание 2» предполагает использование, в свою очередь, режимов запуска SPMD и MPMD параллельных программ [1, 18, 24]. При работе в режиме SPMD (*Single Program Multiple Data* – одна программа, множество данных) создается единая для всех узлов кластера (или для всех потоков при многопоточных узлах) программа [1, 18]. Копии этой программы загружаются параллельно с управляющего узла или с сервера (см. рис. 1) на каждый рабочий узел кластера. У каждого узла (или потока) имеется собственный набор данных.

При работе в режиме MPMD (*Multiple Program Multiple Data* – множество программ, множество данных) для каждого узла (или потока) создается своя собственная программа. При параллельном выполнении этих программ каждая из них использует собственный набор данных [20, 22].

Режим работы «Резидент – агенты» кластера возможно использовать в случае, когда на управляющем узле кластера («Резиденте») сформированы индивидуальные для каждого рабочего узла программы и данные, которые загружаются при отсутствии в кластере широковещательного режима. Такое явление может наблюдаться в TCP/IP сети, протоколы которой не допускают широковещание на сетевом уровне. Ситуация улучшается, когда после загрузки программ и данных сразу начинается обработка на рабочем узле кластера, тогда в распределенном кластере устанавливается режим, занимающий промежуточное положение между режимом «Резидент – агенты» и «Собрание 1».

В режиме работы «Собрание 1» кластера индивидуальные данные загружаются последовательно с управляющего узла на рабочие узлы кластера по IP-адресам (которые имелись к моменту начала загрузки у пользователя и были заданы в программе клиента или управляющего узла). Сразу после завершения всех последовательных загрузок управляющий узел кластера, используя протокол ARP (*Address Resolution Protocol*) [36], обеспечивает сопоставление IP- и MAC-адресов для последующей корректной широковещательной передачи данных в кластере, т.е. при наличии MAC-адресов появляется возможность широковещательной параллельной загрузки копий одной и той же рабочей программы, которые (копии) должны затем параллельно обработать данные, загруженные ранее последовательно на каждый узел кластера.

В режиме «Собрание 2» предполагается, что MAC-адреса уже находятся в предварительно созданной временной таблице (кэш-памяти) соответствий IP- и MAC-адресов для ускорения процесса передачи данных [36]. Поэтому для широковещательной загрузки программ и данных в режимах SPMD и «Собрание 2» проблем не возникает.

Для справки: MAC-адрес (от англ. *Media Access Control* – управление доступом к среде передачи) – уникальный идентификатор, присваиваемый каждой единице сетевого оборудования или некоторым их интерфейсам в компьютерных сетях Ethernet и в некоторых других сетях стандартов IEEE 802 [37].

Заключение

На основе проведенного исследования разработок в области функциональной архитектуры кластерных вычислительных систем показана актуальность выполнения работ в данном направлении, что обусловлено увеличением спроса на доступные высокопроизводительные вычисления, растущим внедрением методов искусственного интеллекта и машинного обучения.

Кластерные системы обладают рядом преимуществ, облегчающих их внедрение в научные исследования и промышленные разработки: кластерные вычисления помогают организациям повысить производительность своих приложений за счет рационального использования ресурсов компьютеров, входящих в кластер.

Внедрение кластеров пониженной стоимости, так называемых кластеров-лоукостеров, позволяет организациям сократить расходы на IT-инфраструктуру. Масштабируемость, гибкость модульных аппаратных и программных решений для кластерных вычислений позволяет организациям уменьшать или увеличивать размер кластера в зависимости от изменяющихся потребностей.

Предложен логико-вероятностный подход к созданию моделей определяемой приложениями и программным обеспечением промежуточного уровня *middleware* функциональной архитектуры кластерных вычислительных систем, позволяющий ускорить создание имитационных моделей для ряда важных режимов использования кластера и осуществить переход к логико-алгебраическим формализованным спецификациям на программные приложения.

Построены имитационные логико-вероятностные и логико-алгебраические модели для ряда важных вариантов использования вычислительного кластера, проведены необходимые статистические эксперименты с данными моделями, давшие обоснования к реализациям соответствующему моделям программному обеспечению промежуточного уровня *middleware*.

Список литературы

1. Воеводин В. В., Воеводин Вл. В. Параллельные вычисления. СПб. : БХВ-Петербург, 2002. 608 с.
2. Архитектурные аспекты параллелизма. URL: http://www.intuit.ru/studies/courses/4447/983/lecture/14921?Page=3 (дата обращения: 02.03.2025).
3. Хенесси Д. Л., Паттерсон Д. А. Компьютерная архитектура. Количественный подход. 5-е изд. М. : Техносфера, 2016. 936 с.
4. Архитектуры вычислительных систем: кластерные системы. URL: https://uzor.belturs.ru/arkhitektury-vychislitel-nykh-sistem-klasternyye-sistemy (дата обращения: 02.03.2025).
5. Вычислительный кластер: его компоненты, виды и примеры реализации. URL: https://itelon.ru/blog/vychislitelnyy-klaster/?ysclid=maavbfouji850319464 (дата обращения: 02.03.2025).
6. Эффективные кластерные решения. URL: https://www.ixbt.com/cpu/clustering.shtml?ysclid=maavg9yxkq334655198 (дата обращения: 02.03.2025).
7. Чем распределенная ВС отличается от кластерной? URL: https://ru.stackoverflow.com/ (дата обращения: 02.03.2025).
8. Петушков Г. В., Сигов А. С. Анализ и выбор структуры многопроцессорной вычислительной системы по критерию быстродействия // Russian Technological Journal. 2024. Т. 12, № 6. С. 20−25. doi: 10.32362/2500-316X-2024-12-6-20-25
9. Многопроцессорные и многомашинные вычислительные системы. URL: https://intuit.ru/studies/educational_groups/960/courses/460/lecture/10345?page=2&ysclid=maawtf2w8o719310938 (дата обращения: 02.03.2025).
10. Виды кластерных систем. URL: https://vuzlit.com/963152/vidy_klasternyh_sistem (дата обращения: 02.03.2025).
11. Computer cluster. URL: https://en.wikipedia.org/wiki/Computer_cluster (дата обращения: 02.03.2025).
12. Суперкомпьютеры «Яндекса». URL: https://ru.wikipedia.org/wiki/Суперкомпьютеры_«Яндекса» (дата обращения: 02.03.2025).
13. Симонов А. С., Жабин И. А., Куштанов Е. Р., Макагон Д. В., Семенов А. С., Щербак А. Н. Высокоскоростная сеть Ангара: архитектура и результаты применения // Вопросы кибербезопасности. 2019. № 4 (32). С. 46–53.
14. Евреинов Э. В., Хорошевский В. Г. Однородные вычислительные системы. Новосибирск : Наука, Сибирское отделение, 1978. 319 с.
15. Димитриев Ю. К., Хорошевский В. Г. Вычислительные системы из мини-ЭВМ / под ред. Э. В. Евреинова. М. : Радио и связь, 1982. 304 с.
16. Хорошевский В. Г. Инженерный анализ функционирования вычислительных машин и систем. М. : Радио и связь, 1987. 254 с.
17. Миренков М. М. Параллельное программирование для многомодульных вычислительных систем. М. : Радио и связь, 1989. 320 с.
18. Хорошевский В. Г. Архитектура вычислительных систем. М. : Изд-во МГТУ им. Н. Э. Баумана, 2005. 510 с.
19. Feng Liu, Haitao Wu, Xiaochun Lu, Xiyang Liu. Parallel Distributed Acceleration Based on MPI and OpenMP Technology // International Journal of Grid Distribution Computing. 2015. Vol. 8, № 6. P. 171–184.
20. Mittal G., Kesswani N, Goswami K. A Survey of Current Trends in Distributed, Grid and Cloud Computing // International Journal of Advanced Studies in Computer Science and Engineering (IJASCSE). 2013. Vol. 2, iss. 3. P. 1–6.

21. Kahanwal B., Singh T. P. The Distributed Computing Paradigms: P2P, Grid, Cluster, Cloud, and Jungle // International Journal of Latest Research in Science and Technology. 2012. Vol. 1, iss. 2. P. 183–187.
22. Seinstra F. J., Maassen J., van Nieuwpoort R. V., Drost N. [et al.]. Jungle Computing: Distributed Supercomputing beyond Clusters, Grids, and Clouds / Department of Computer Science, Vrije Universiteit, De Boelelaan 1081A, 1081 HV Amsterdam, The Netherlands, 2010. P. 1–31.
23. Kumar R. Comparison between Cloud Computing, Grid Computing, Cluster Computing and Virtualization // International Journal of Modern Computer Science and Applications (IJMCSA). 2015. Vol. 3, № 1. P. 42–47.
24. Sterling T., Cwik T., Becker D., Salmon J., Warren M., Nitzberg B. An Assessment of Beowulf-class Computing for NASA Requirements: Initial Findings from the First NASA Workshop on Beowulf-class Clustered Computing // IEEE Aerospace Conference Proceedings. 1998. P. 1–16. doi: 10.1109/AERO.1998.682207
25. MPI: A Message-Passing Interface Standard. Version 3.1 // Message Passing Interface Forum. 2015. 836 p. URL: https://www.mpi-forum.org/docs/mpi-3.1/mpi31-report.pdf (дата обращения: 02.03.2025).
26. Almasi G. S., Bhanot G. V. [et al.]. Early Experience with Scientific Applications on the Blue Gene/L Supercomputer // Lecture Notes in Computer Science. 2005. Vol. 3648. P. 560–570. doi: 10.1007/11549468_63
27. Hamada T., Nitadori K. 190 TFlops astrophysical N-body simulation on a cluster of GPUs // In Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC '10). Washington, DC, 2010. P. 1–9. doi: 10.1109/SC.2010.1
28. Enabling the Modern Data Center – RDMA for the Enterprise. URL: https://www.infinibandta.org/wp-ontent/uploads/2019/05/IBTA_WhitePaper_May-20-2019.pdf (дата обращения: 02.03.2025).
29. RDMA_over_Converged_Ethernet. URL: https://en.wikipedia.org/wiki/RDMA_over_Converged_Ethernet (дата обращения: 02.03.2025).
30. Deploying HPC Cluster with Mellanox InfiniBand Interconnect Solutions / Reference Design. Rev 1.3. 2017. 40 p.
31. На пути к созданию отечественного суперкомпьютера субэкзафлопсной производительности. URL: https://www.ospcon.ru/files/media/Simonov.pdf (дата обращения: 02.03.2025).
32. Sugon. URL: https://en.wikipedia.org/wiki/Sugon (дата обращения: 02.03.2025).
33. InfiniBand Clustering. Delivering Better Price/Performance than Ethernet. URL: https://network.nvidia.com/pdf/whitepapers/IB_vs_Ethernet_Clustering_WP_100.pdf (дата обращения: 02.03.2025).
34. Список 500 самых мощных компьютеров мира. URL: https://parallel.ru/computers/top500.list61.html (дата обращения: 02.03.2025).
35. Anderson D. P. BOINC: A System for Public-Resource Computing and Storage. URL: https://boinc.berkeley.edu/grid_paper_04.pdf (дата обращения: 02.03.2025).
36. Зелов С. Кластерные технологии. URL: https://compress.ru/article.aspx?id=9958&ysclid=maawhrd9vh678328188 (дата обращения: 02.03.2025).
37. Forouzan B. A. TCP/IP Protocol Suite. McGraw-Hill, 2009. 1024 p.
38. Standard Group MAC Addresses : a Tutorial Guide. URL: http://standards.ieee.org/regauth/groupmac/tutorial.html (дата обращения: 02.03.2025).
39. Прицкер А. Введение в имитационное моделирование и язык СЛАМ II. М. : Мир, 1987. 646 с.
40. Ломакина Л. С., Базин А. С., Вигура А. Н., Киселев А. В. Теория и практика структурного тестирования программных систем. Воронеж : Научная книга, 2013. 220 с.
41. Кулагин В. П. Формирование информационных ресурсов на основе параллельных вычислений // Перспективы науки и образования. 2013. № 6. С. 26–31.

42. Котов В. Е. Сети Петри. М. : Наука, 1984. 160 с.
43. Поликарпова Н. И., Шалыто А. А. Автоматное программирование. СПб. : Изд-во Санкт-Петербургского государственного университета информационных технологий, механики и оптики, 2008. 167 с.
44. Ющенко Е. Л., Цейтлин Г. Е., Грицай В. П., Терзян Т. К. Многоуровневое структурное проектирование программ: Теоретические основы, инструментарий. М. : Финансы и статистика, 1989. 208 с.
45. Волчихин В. И., Зинкин С. А. Логико-алгебраические модели и методы в проектировании функциональной архитектуры распределенных систем хранения и обработки данных // Известия высших учебных заведений. Поволжский регион. Технические науки. 2012. № 2. С. 3–16.
46. Зинкин С. А. Элементы новой объектно-ориентированной технологии для моделирования и реализации систем и сетей хранения и обработки данных // Информационные технологии. 2008. № 10. С. 20–27.
47. Микроэкономика. URL: https://ru.wikipedia.org/wiki/Микроэкономика (дата обращения: 02.03.2025).
48. Макроэкономика. URL: https://ru.wikipedia.org/wiki/Макроэкономика (дата обращения: 02.03.2025).
49. Петушков Г. В., Сигов А. С. Технико-экономический анализ серверов как вычислительных модулей вычислительных систем класса WSC // Russian Technological Journal. 2025. Vol. 13 (1). P. 49–58. doi: 10.32362/2500-316X-2025-13-1-49-58 EDN: JQICRJ
50. Евреинов Э. В., Косарев Ю. Г. Однородные универсальные системы высокой производительности. Новосибирск : Наука, Сибирское отделение. 1966. 308 с.
51. Байцер Б. Микроанализ производительности вычислительных систем. М. : Радио и связь, 1983. 360 с.
52. Капитонова Ю. В., Кривой С. Л., Летичевский А. А., Луцкий Г. М. Лекции по дискретной математике. СПб. : БХВ-Петербург, 2004. 624 с.
53. Зинкин С. А. Интеллектуализация и интеграция систем и сетей хранения и обработки данных. Пенза : Изд-во ПГУ, 2023. 416 с.
54. Кудрявцев Е. М. GPSS World. Основы имитационного моделирования различных систем. М. : ДМК Пресс, 2004. 320 с.

References

1. Voevodin V.V., Voevodin Vl.V. *Parallel'nye vychisleniya = Parallel computing*. Saint Petersburg: BKhV-Peterburg, 2002:608. (In Russ.)
2. *Arkhitekturnye aspekty parallelizma = Architectural aspects of parallelism*. (In Russ.). Available at: http://www.intuit.ru/studies/courses/4447/983/lecture/14921?Page=3 (accessed 02.03.2025).
3. Khenessi D.L., Patterson D.A. *Komp'yuternaya arkhitektura. Kolichestvennyy podkhod. 5-e izd. = Computer architecture: a quantitative approach: the 5th edition*. Moscow: Tekhnosfera, 2016:936. (In Russ.)
4. *Arkhitektury vychislitel'nykh sistem: klasternye sistemy = Computer system architectures: cluster systems.* (In Russ.). Available at: https://uzor.belturs.ru/arkhitektury-vychislitel-nykh-sistem-klasternyye-sistemy (accessed 02.03.2025).
5. *Vychislitel'nyy klaster: ego komponenty, vidy i primery realizatsii = Computing cluster: its components, types and implementation examples.* (In Russ.). Available at: https://itelon.ru/blog/vychislitelnyy-klaster/?ysclid=maavbfouji850319464 (accessed 02.03.2025).
6. *Effektivnye klasternye resheniya = Effective cluster solutions*. (In Russ.). Available at: https://www.ixbt.com/cpu/clustering.shtml?ysclid=maavg9yxkq334655198 (accessed 02.03.2025).

7. *Chem raspredelennaya VS otlichaetsya ot klasternoy? = How does a distributed computing system differ from a clustered one?* (In Russ.). Availabble at: https://ru.stackoverflow.com/ (accessed: 02.03.2025).
8. Petushkov G.V., Sigov A.S. Analysis and selection of the structure of a multiprocessor computing system based on the performance criterion. *Russian Technological Journal*. 2024;12(6):20−25. (In Russ.). doi: 10.32362/2500-316X-2024-12-6-20-25
9. *Mnogoprotsessornye i mnogomashinnye vychislitel'nye sistemy = Multiprocessor and multimachine computing systems.* (In Russ.). Available at: https://intuit.ru/studies/educational_groups/960/courses/460/lecture/10345?page=2&ysclid=maawtf2w8o719310938 (accessed 02.03.2025).
10. *Vidy klasternykh system = Types of cluster systems*. (In Russ.). Available at: https://vuzlit.com/963152/vidy_klasternyh_sistem (accessed 02.03.2025).
11. *Computer cluster*. Available at: https://en.wikipedia.org/wiki/Computer_cluster (accessed 02.03.2025).
12. *Superkomp'yutery «Yandeksa» = Yandex supercomputers*. (In Russ.). Available at: https://ru.wikipedia.org/wiki/Superkomp'yutery_«Yandeksa» (accessed 02.03.2025).
13. Simonov A.S., Zhabin I.A., Kushtanov E.R., Makagon D.V., Semenov A.S., Shcherbak A.N. High-speed Angara network: architecture and application results. *Voprosy kiberbezopasnosti = Cybersecurity issues*. 2019;(4):46–53. (In Russ.)
14. Evreinov E.V., Khoroshevskiy V.G. *Odnorodnye vychislitel'nye sistemy = Homogeneous computing systems.* Novo-sibirsk: Nauka, Sibirskoe otdelenie. 1978:319. (In Russ.)
15. Dimitriev Yu.K., Khoroshevskiy V.G. *Vychislitel'nye sistemy iz mini-EVM = Computing systems from minicomputers*. Moscow: Radio i svyaz', 1982:304. (In Russ.)
16. Khoroshevskiy V.G. *Inzhenernyy analiz funktsionirovaniya vychislitel'nykh mashin i system = Engineering analysis of the functioning of computers and systems*. Moscow: Radio i svyaz', 1987:254. (In Russ.)
17. Mirenkov M.M. *Parallel'noe programmirovanie dlya mnogomodul'nykh vychislitel'nykh system = Parallel programming for multi-module computing systems*. Moscow: Radio i svyaz', 1989:320. (In Russ.)
18. Khoroshevskiy V.G. *Arkhitektura vychislitel'nykh system = Architecture of computing systems.* Moscow: Izd-vo MGTU im. N.E. Baumana, 2005:510. (In Russ.)
19. Feng Liu, Haitao Wu, Xiaochun Lu, Xiyang Liu. Parallel Distributed Acceleration Based on MPI and OpenMP Technology. *International Journal of Grid Distribution Computing*. 2015;8(6):171–184.
20. Mittal G., Kesswani N, Goswami K. A Survey of Current Trends in Distributed, Grid and Cloud Computing. *International Journal of Advanced Studies in Computer Science and Engineering (IJASCSE)*. 2013;2(3):1–6.
21. Kahanwal B., Singh T.P. The Distributed Computing Paradigms: P2P, Grid, Cluster, Cloud, and Jungle. *International Journal of Latest Research in Science and Technology*. 2012;1(2):183–187.
22. Seinstra F.J., Maassen J., van Nieuwpoort R.V., Drost N. et al. *Jungle Computing: Distributed Supercomputing beyond Clusters, Grids, and Clouds*. Department of Computer Science, Vrije Universiteit, De Boelelaan 1081A, 1081 HV Amsterdam, The Netherlands, 2010:1–31.
23. Kumar R. Comparison between Cloud Computing, Grid Computing, Cluster Computing and Virtualization. *International Journal of Modern Computer Science and Applications (IJMCSA)*. 2015;3(1):42–47.
24. Sterling T., Cwik T., Becker D., Salmon J., Warren M., Nitzberg B. An Assessment of Beowulf-class Computing for NASA Requirements: Initial Findings from the First NASA Workshop on Beowulf-class Clustered Computing. *IEEE Aerospace Conference Proceedings*. 1998:1–16. doi: 10.1109/AERO.1998.682207
25. MPI: A Message-Passing Interface Standard. Version 3.1. *Message Passing Interface Forum*. 2015:836. Available at: https://www.mpi-forum.org/docs/mpi-3.1/mpi31-report.pdf (accessed 02.03.2025).

26. Almasi G.S., Bhanot G.V. et al. Early Experience with Scientific Applications on the Blue Gene/L Supercomputer. *Lecture Notes in Computer Science*. 2005;3648:560–570. doi: 10.1007/11549468_63
27. Hamada T., Nitadori K. 190 TFlops astrophysical N-body simulation on a cluster of GPUs. *In Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC '10)*. Washington, DC, 2010:1–9. doi: 10.1109/SC.2010.1
28. *Enabling the Modern Data Center – RDMA for the Enterprise*. Available at: https://www.infinibandta.org/wp-ontent/uploads/2019/05/IBTA_WhitePaper_May-20-2019.pdf (accessed 02.03.2025).
29. *RDMA_over_Converged_Ethernet*. Available at: https://en.wikipedia.org/wiki/RDMA_over_Converged_Ethernet (accessed 02.03.2025).
30. *Deploying HPC Cluster with Mellanox InfiniBand Interconnect Solutions* / Reference Design. Rev 1.3. 2017:40.
31. *Na puti k sozdaniyu otechestvennogo superkomp'yutera subekzaflopsnoy proizvoditel'nosti = Towards the creation of a domestic supercomputer with sub-exascale performance.* (In Russ.). Available at: https://www.ospcon.ru/files/media/Simonov.pdf (accessed 02.03.2025).
32. *Sugon*. Available at: https://en.wikipedia.org/wiki/Sugon (accessed 02.03.2025).
33. *InfiniBand Clustering. Delivering Better Price/Performance than Ethernet*. Available at: https://network.nvidia.com/pdf/whitepapers/IB_vs_Ethernet_Clustering_WP_100.pdf (accessed 02.03.2025).
34. *Spisok 500 samykh moshchnykh komp'yuterov mira = List of the 500 most powerful computers in the world.* (In Russ.). Available at: https://parallel.ru/computers/top500.list61.html (accessed 02.03.2025).
35. Anderson D.P. *BOINC: A System for Public-Resource Computing and Storage*. Available at: https://boinc.berkeley.edu/grid_paper_04.pdf (accessed 02.03.2025).
36. Zelov S. *Klasternye tekhnologii = Cluster technologies*. (In Russ.). Available at: https://compress.ru/article.aspx?id=9958&ysclid=maawhrd9vh678328188 (accessed 02.03.2025).
37. Forouzan B.A. *TCP/IP Protocol Suite*. McGraw-Hill, 2009:1024.
38. *Standard Group MAC Addresses: a Tutorial Guide*. Available at: http://standards.ieee.org/regauth/groupmac/tutorial.html (accessed 02.03.2025).
39. Pritsker A. *Vvedenie v imitatsionnoe modelirovanie i yazyk SLAM II = Introduction to Simulation and the SLAM II Language*. Moscow: Mir, 1987:646. (In Russ.)
40. Lomakina L.S., Bazin A.S., Vigura A.N., Kiselev A.V. *Teoriya i praktika strukturnogo testirovaniya programmnykh system = Theory and practice of structural testing of software systems*. Voronezh: Nauchnaya kniga, 2013:220. (In Russ.)
41. Kulagin V.P. Formation of information resources based on parallel computing. *Perspektivy nauki i obrazovaniya = Prospects of Science and Education*. 2013;(6):26–31. (In Russ.)
42. Kotov V.E. *Seti Petri = Petri net.* Moscow: Nauka, 1984:160. (In Russ.)
43. Polikarpova N.I., Shalyto A.A. *Avtomatnoe programmirovanie = Automata-based programming*. Saint Petersburg: Izd-vo Sankt-Peterburgskogo gosudarstvennogo universiteta informatsionnykh tekhno-logiy, mekhaniki i optiki, 2008:167. (In Russ.)
44. Yushchenko E.L., Tseytlin G.E., Gritsay V.P., Terzyan T.K. *Mnogourovnevoe strukturnoe proektirovanie programm: Teoreticheskie osnovy, instrumentariy = Multilevel structural design of programs: Theoretical foundations, tools.* Moscow: Finansy i statistika, 1989:208. (In Russ.)
45. Volchikhin V.I., Zinkin S.A. Logical-algebraic models and methods in designing the functional architecture of distributed data storage and processing systems. *Izvestiya vysshikh uchebnykh zavedeniy. Povolzhskiy region. Tekhnicheskie nauki = University proceedings. Volga region. Engineering sciences*. 2012;(2):3–16. (In Russ.)

46. Zinkin S.A. Elements of a new object-oriented technology for modeling and implementing data storage and processing systems and networks. *Informatsionnye tekhnologii = Information technology*. 2008;(10):20–27. (In Russ.)
47. *Mikroekonomika = Microeconomics*. (In Russ.). Available at: https://ru.wikipedia.org/wiki/Mikroekonomika (accessed 02.03.2025).
48. *Makroekonomika = Microeconomics*. (In Russ.). Available at: https://ru.wikipedia.org/wiki/Makroekonomika (accessed 02.03.2025).
49. Petushkov G.V., Sigov A.S. Technical and economic analysis of servers as computing modules of computing systems of the class WSC. *Russian Technological Journal*. 2025;13(1):49–58. (In Russ.). doi: 10.32362/2500-316X-2025-13-1-49-58 EDN: JQICRJ
50. Evreinov E.V., Kosarev Yu.G. *Odnorodnye universal'nye sistemy vysokoy proizvoditel'nosti = Homogeneous universal high-performance systems*. Novosibirsk: Nauka, Sibirskoe otdelenie. 1966:308. (In Russ.)
51. Baytser B. *Mikroanaliz proizvoditel'nosti vychislitel'nykh system = Microanalysis of computing system performance.* Moscow: Radio i svyaz', 1983:360. (In Russ.)
52. Kapitonova Yu.V., Krivoy S.L., Letichevskiy A.A., Lutskiy G.M. *Lektsii po diskretnoy matematike = Lectures on Discrete Mathematics.* Saint Petersburg: BKhV-Peterburg, 2004:624. (In Russ.)
53. Zinkin S.A. *Intellektualizatsiya i integratsiya sistem i setey khraneniya i obrabotki dannykh = Intellectualization and integration of data storage and processing systems and networks.* Penza: Izd-vo PGU, 2023:416. (In Russ.)
54. Kudryavtsev E.M. *GPSS World. Osnovy imitatsionnogo modelirovaniya razlichnykh system = GPSS World. Fundamentals of simulation modeling of various systems*. Moscow: DMK Press, 2004:320. (In Russ.)

Информация об авторах / Information about the authors

Григорий Валерьевич Петушков
проректор, МИРЭА – Российский технологический университет (Россия, г. Москва, пр-кт Вернадского, 78)

E-mail: petushkov@mirea.ru

Grigory V. Petushkov
Vice-Rector, MIREA – Russian Technological University (78 Vernadskogo avenue, Moscow, Russia)

Александр Сергеевич Сигов
доктор физико-математических наук, профессор, президент МИРЭА – Российский технологический университет, академик Российской академии наук (Россия, г. Москва, пр-кт Вернадского, 78)

E-mail: sigov@mirea.ru

Alexander S. Sigov
Doctor of physical and mathematical sciences, professor, President, MIREA - Russian Technological University, Academician, Russian Academy of Sciences (78 Vernadskogo avenue, Moscow, Russia)

Авторы заявляют об отсутствии конфликта интересов / The authors declare no conflicts of interests.

Поступила в редакцию / Received 16.05.2025

Поступила после рецензирования и доработки / Revised 29.08.2025

Принята к публикации / Accepted 10.09.2025