(19) Евразийское патентное ведомство

(21) 202390955 (13) A1

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ЕВРАЗИЙСКОЙ ЗАЯВКЕ

(43) Дата публикации заявки
2023.12.13

(22) Дата подачи заявки
2023.04.21

(51) Int. Cl. ***G06F 17/40*** (2006.01)
G16Y 40/00 (2020.01)

(54) СИСТЕМА УПРАВЛЕНИЯ, СБОРА, ОБРАБОТКИ, ХРАНЕНИЯ И ПРЕДОСТАВЛЕНИЯ ДОСТУПА К ДАННЫМ УСТРОЙСТВ ПРОМЫШЛЕННОЙ АВТОМАТИЗАЦИИ И ИНТЕРНЕТА ВЕЩЕЙ

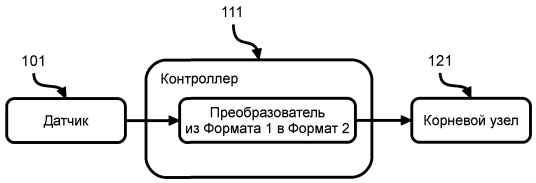
(31) **2022111385**
(32) **2023.04.26**
(33) **RU**

(71) Заявитель:
ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ "ЛАБОРАТОРИЯ ТЕХНОЛОГИЙ АВТОМАТИЗАЦИИ" (RU)

(72) Изобретатель:
Подольный Вадим Павлович, Зайцев Александр Валерьевич, Шелудько Сергей Дмитриевич (RU)

(74) Представитель:
Абраменко О.И. (RU)

(57) Данное изобретение относится к вычислительной технике и, в частности, к системам и способам промышленной автоматизации интернета вещей. Техническим результатом является повышение эффективности обработки данных, повышение отказоустойчивости, масштабируемости системы. Задачей предлагаемого изобретения является повышение эффективности сбора, обработки данных, отказоустойчивости системы.



202390955 A1

202390955 A1

СИСТЕМА УПРАВЛЕНИЯ, СБОРА, ОБРАБОТКИ, ХРАНЕНИЯ И ПРЕДОСТАВЛЕНИЯ ДОСТУПА К ДАННЫМ УСТРОЙСТВ ПРОМЫШЛЕННОЙ АВТОМАТИЗАЦИИ И ИНТЕРНЕТА ВЕЩЕЙ

ОБЛАСТЬ ТЕХНИКИ

Данное техническое решение относится к вычислительной технике и, в частности, к системам и способам промышленной автоматизации интернета вещей.

УРОВЕНЬ ТЕХНИКИ

Современные автоматизированные системы управления технологическими и производственными процессами (АСУ ТП, АСУ ПП), в том числе объектами критической информационной инфраструктуры (КИИ) размещаются в промышленных сетях, которые отличаются особой архитектурой и надежностью. Подразумевается, что в сети промышленного контура или Интернета вещей располагается совокупность датчиков и исполнительных механизмов, которые, в свою очередь, сопряжены с контроллерами или являются их составными частями. Контроллер, в свою очередь, представляет собой логическое устройство, способное осуществлять предварительную обработку данных и оперировать командами управления сопряженных исполнительных механизмов. Контроллер может содержать микропроцессор, управляться с помощью специализированной операционной системы (ОС) или встроенного программного обеспечения (ПО), или проще, состоять из простейших логических элементов и управляться, в свою очередь, более сложным контроллером. Более сложный контроллер может быть и программный и располагаться на промышленном сервере.

В настоящее время довольно остро стоит проблема в решениях, которые позволяют гибко собирать, хранить и управлять промышленными системами интернета вещей.

Задачей предлагаемого технического решения является повышение эффективности сбора, обработки данных, отказоустойчивости системы.

СУЩНОСТЬ ТЕХНИЧЕСКОГО РЕШЕНИЯ

Технический результат – повышение эффективности обработки данных, повышении отказустойчивости, масштабируемости системы.

Согласно одному из вариантов реализации, предлагается система управления, сбора, обработки, хранения и предоставления доступа к данным устройств промышленной автоматизации и Интернета вещей, включает: по крайней мере два датчика, каждый из которых связан по крайне мере с одним контроллером; по крайней мере два контроллера, причем каждый контроллер содержит один или более обрабатываемых портов ввода/вывода представляющих собой переменные и связан с одним или более датчиком; по крайней мере два корневых узла, выполненных с возможностью:

- получения и регистрации переменных от по крайней мере одного контроллера;
- синхронизации переменных;
- кэширования изменений переменных путем обновления текущего значения переменной, зарегистрированной в корневом узле, попадающего в него из очереди сообщений об изменениях
- синхронизации кэшей, путем рассылки изменений значений переменных различными способами, в зависимости от режима работы корневых узлов
- обеспечения консистентности при синхронизации кэшей путем приоритезации за счет применения специальной маркировки переменных, которые задают значимость их синхронизации, причем сначала синхронизируются более значимые параметры, и далее в порядке приоритета значимости;
- объединения корневых узлов, работающих в режиме мультимастера в вычислительный кластер;
- синхронизации кластерных сценариев;
- построения эффективных сценариев параллельных вычислений;

причем система выполнена с возможностью:

- обмена данными между контроллерами и корневыми узлами, используя очереди сообщений;
- резервирования значений переменных на корневых узлах;
- диагностики доступных корневых узлов, включая загруженность.

В некоторых вариантах реализации данные из портов ввода/вывода могут быть представлены в виде переменных, входящих в состав пары ключ-значение.

В некоторых вариантах реализации переменные могут быть оперативными или техническими.

В некоторых вариантах реализации регистрация переменных осуществляется путем ввода проектных данных в любой из доступных корневых узлов.

В некоторых вариантах реализации очереди обрабатываются по принципу LIFO или FIFO.

В некоторых вариантах реализации очереди обрабатываются по приоритетам.

В некоторых вариантах реализации данные из очереди сообщений распределенно записываются в несколько корневых узлов.

В некоторых вариантах реализации для синхронизации кэшей с низкой задержкой используется аппаратная конфигурация, в которой скорость шины серверного интерконнекта, по которой осуществляется синхронизация кэшей, будет выше или равна суммарной скорости всех портов.

В некоторых вариантах реализации синхронизация кэшей осуществляется в асинхронном режиме.

В некоторых вариантах реализации корневые узлы связаны друг с другом в режиме основной – резервный;

В некоторых вариантах реализации корневые узлы связаны друг с другом в режиме основной – основной;

В некоторых вариантах реализации диагностика осуществляется с использованием одной или более следующих метрик и их комбинациями: дедлайн, латентность, джиттер, накопительная метрика изменения значения переменной.

В некоторых вариантах реализации синхронизации кэшей осуществляется посредством мультивещания без резервных узлов в режиме работы корневых узлов основной-основной.

КРАТКОЕ ОПИСАНИЕ ГРАФИЧЕСКИХ МАТЕРИАЛОВ

ФИГ. 1 иллюстрирует примерный вариант преобразования данных на контроллере более низкого уровня по отношению к узлу путем интеграции в него преобразователя, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 2 иллюстрирует примерный вариант преобразования данных на узлах, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 3 иллюстрирует примерный вариант обмена данными корневых узлов с клиентами, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 4 иллюстрирует примерный вариант обмен данными между корневыми узлами, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 5 иллюстрирует примерный вариант обмена данными множества корневых узлов с клиентами, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 6 иллюстрирует примерный вариант обмена данными корневых узлов с клиентами с резервированием значений на корневых узлах, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 7 иллюстрирует примерный вариант распределенной записи данных в несколько корневых узлов, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 8 иллюстрирует примерный вариант распределенной записи данных в несколько корневых узлов с синхронизацией корневых узлов, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 9 иллюстрирует примерный вариант синхронизации кэшей с максимально низкой задержкой, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 10 иллюстрирует примерный вариант синхронизации переменных с более высокими метриками приоритета, далее в порядке уменьшения этого приоритета, согласно одному из вариантов осуществления настоящего изобретения;

ФИГ. 11 иллюстрирует пример синхронизации кэшей путем однонаправленной передачи данных, мультивещанием, в режиме функционирования корневых узлов «основной – несколько резервных», согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 12 иллюстрирует пример синхронизации кэшей посредством мультивещания без резервных узлов в режиме работы корневых узлов «несколько основных», согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 13 иллюстрирует пример синхронизации кэшей посредством мультивещания в режиме работы корневых узлов «несколько основных – несколько резервных (мультимастер - мультислейв)», согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 14 иллюстрирует пример управления приемными и передающими клиентскими очередями балансировщиком очередей клиентов, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 15 иллюстрирует пример управления очередями синхронизации корневых узлов балансировщиком очередей синхронизации, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 16 иллюстрирует вариант выполнения балансировщика нагрузки клиентских очередей на специально выделенном узле, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 17 иллюстрирует вариант выполнения балансировщика нагрузки клиентских очередей на нескольких специально выделенных узлах, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 18 иллюстрирует вариант встроенного балансировщика в каждый корневой узел в случае кластера, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 19 иллюстрирует примерный вариант обеспечения эффективного управления очередями сообщений балансировщиком нагрузки, согласно одному из вариантов настоящего технического решения;

ФИГ. 20 иллюстрирует примерный вариант балансировки аппаратных ресурсов, согласно одному из вариантов настоящего технического решения;

ФИГ. 21 иллюстрирует примерный вариант балансировки аппаратных ресурсов и обработки критических данных, согласно одному варианту осуществления настоящего технического решения;

ФИГ. 22 иллюстрирует примерный вариант балансировки при параллельных вычислениях, в частности, показана схема балансировки узлов, нацеленная на поддержание эффективности параллельных вычислений, согласно одному варианту осуществления настоящего технического решения;

ФИГ. 23 иллюстрирует примерный вариант параллельных вычислений, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 24 иллюстрирует примерный вариант указания переменных для узлов с целью синхронизации, согласно одному из вариантов настоящего технического решения;

ФИГ. 25 иллюстрирует примерный вариант сегментации переменных на корневых узлах, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 26 иллюстрирует примерный вариант сегментации переменных на корневых узлах в случае отказа одного из корневых улов, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 27 иллюстрирует примерный вариант переключения на другой корневой узел без задержек с сохранением подписки отказе корневого узла, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 28 иллюстрирует примерный вариант переключения клиента при перераспределении нагрузки между корневыми узлами с использованием балансировщика нагрузки, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 29 иллюстрирует примерный вариант переменных, отвечающих за диагностику, определение состояния и управления электростанцией;

ФИГ. 30 иллюстрирует примерный вариант резервирования межблочной сетью с планированием запаса по вычислительной мощности при высокой вероятности отказа всех резервов на блоке, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 31 иллюстрирует примерный вариант шести клиентских очередей сообщений обработки переменных на каждый сервер (из них три - в резерве), и шести очередей сообщений синхронизации, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 32 и **ФИГ. 33** иллюстрируют примерный вариант в случае отказа двух серверов системы верхнего уровня с переключением операторских рабочих станций на резервный сервер, согласно одному из вариантов осуществления настоящего технического решения;

ФИГ. 34 иллюстрирует примерный вариант в случае отказа системы верхнего уровня в целом, включая два сервера и операторские рабочие станции, с переходом оператором на блочный пульт с использованием рабочего места одного из операторов, согласно одному из вариантов осуществления настоящего технического решения;

ОПИСАНИЕ ВАРИАНТОВ ОСУЩЕСТВЛЕНИЯ

Объекты и признаки настоящего технического решения, способы для достижения этих объектов и признаков станут очевидными посредством отсылки к примерным вариантам осуществления. Однако настоящее техническое решение не ограничивается примерными вариантами осуществления, раскрытыми ниже, оно может воплощаться в различных видах. Сущность, приведенная в описании, является ничем иным, как конкретными деталями, обеспеченными для помощи специалисту в области техники в исчерпывающем понимании технического решения, и настоящее техническое решение определяется только в объеме приложенной формулы.

Используемые в настоящем описании настоящего технического решения термины «модуль», «компонент», «элемент» и подобные используются для обозначения

компьютерных сущностей, которые могут являться аппаратным обеспечением/оборудованием (например, устройством, инструментом, аппаратом, аппаратурой, составной частью устройства, например, процессором, микропроцессором, интегральной схемой, печатной платой, в том числе электронной печатной платой, макетной платой, материнской платой и т.д., микрокомпьютером и так далее), программным обеспечением (например, исполняемым программным кодом, скомпилированным приложением, программным модулем, частью программного обеспечения или программного кода и так далее) и/или микропрограммой (в частности, прошивкой). Так, например, компонент может быть процессом, выполняющемся на процессоре (процессором), объектом, исполняемым кодом, программным кодом, файлом, программой/приложением, функцией, методом, (программной) библиотекой, подпрограммой, сопрограммой и/или вычислительным устройством (например, микрокомпьютером или компьютером) или комбинацией программных или аппаратных компонентов.

На **ФИГ. 1** и **ФИГ. 2** показаны примерные варианты преобразования разнообразия форматов данных (протоколов) различных контроллеров (источников) в единообразный формат данных, в частности, на **ФИГ. 1** показан примерный вариант преобразования данных на контроллере более низкого уровня по отношению к узлу (корневому контроллеру) путем интеграции в него преобразователя, на **ФИГ. 2** показан примерный вариант преобразования данных на узлах.

Предлагаемое техническое решение относится к области вычислительной и контрольно-измерительной техники, может использоваться в системах цифровой обработки информации и управления, содержащих избыточные аппаратные и программные средства, с целью обеспечения отказоустойчивости данных систем, таких как автоматизированные системы управления технологическими и производственными процессами, промышленного интернета вещей, и прочих систем специального назначения.

Для обеспечения надежности и резервирования, датчики (**101**), или, по меньшей мере, их часть, и исполнительные механизмы, или, по меньшей мере, их часть, могут быть сопряжены с несколькими контроллерами (**111**), например, двумя, тремя или более. Такое сопряжение (такая схема), в частном случае, обеспечивает возможность повысить надежность сбора данных и управления в критически важных узлах объекта промышленной автоматизации, применяя схемы голосования, такие как, например, «два из трех», когда

достоверным является то значение, которое пришло (было получено) как минимум с двух контроллеров (**111**).

В частном случае, предлагаемое техническое решение обеспечивает возможность эффективно управлять большим числом контроллеров (**111**), десятков, сотен тысяч и более, обеспечивать обработку данных устройств, сопряженных сложными логическими взаимосвязями.

В частном случае, предлагаемое техническое решение обеспечивает надежный механизм сбора данных с датчиков (**101**) и управления исполнительными механизмами объектов автоматизации и Интернета вещей, используя преимущества распределенности, резервируемости, в совокупности, обеспечивая отказоустойчивость объекта автоматизации в целом.

В частном случае, предлагаемое техническое решение обеспечивает возможность обработки данных распределенных резервируемых контроллеров (**111**), обеспечивая общую распределенную сеть обработки данных, путем создания общего распределенного хранилища данных, и механизмов обеспечения к нему доступа.

В частном случае, предлагаемое техническое решение обеспечивает построение (формирование) распределенного хранилища и методы работы с ним. Каждый контроллер (**111**) содержит совокупность обрабатываемых портов ввода/вывода, которые, в данном техническом решении описаны в формате совокупности переменных, которые в свою очередь представляют собой пару, параметр и значение. В свою очередь, переменные разделяются на два типа, оперативные (динамические переменные, часто меняются) и технические (статические переменные, редко меняются). Оперативные данные, обладают приоритетной доступностью для обработки относительно технических, следовательно требования к обеспечению записи, хранения и доступности, в частном случае, отличаются. В настоящем техническом решении оперативные данные хранятся в оперативной памяти, а технические хранятся по мере доступности аппаратных ресурсов, например, на скоростных твердотельных накопителях (SSD). В частном случае, предлагаемое техническое решение решают задачу сбора, хранения, обработки и предоставления доступа большого количества оперативных данных с максимально низкой задержкой (латентностью). Для этого предлагаемое техническое решение, в частном случае, содержит набор специальных методов, которые описываются далее.

В промышленных сетях присутствует большое число контроллеров (**111**) различных производителей, которые взаимодействуют по самым разным протоколам.

В рамках настоящего технического решения осуществляется преобразование разнообразия форматов данных (протоколов) различных контроллеров (источников) в единообразный формат данных. В частном случае, для повышения эффективности преобразования такое преобразование может осуществляться как можно на более низком уровне контроллера (**111**), в частности, устройства, источника, путем интеграции в него соответствующего преобразователя (в частности, кода, алгоритма, программы) в прошивку или операционную систему контроллера **111**, как показано на **ФИГ. 1**. После преобразования данные в принятом, единообразном, формате передаются (поднимаются) до уровня корневых узлов (корневых контроллеров) **121**. В частном случае в рамках настоящего технического решения осуществляется преобразование данных непосредственно на корневых узлах (корневых контроллерах) **121**, как показано на **ФИГ. 2** В частном случае, преобразование данных на корневых узлах (корневых контроллерах) **121** требует дополнительных ресурсов корневого узла (корневого контроллера) **121**, что может снизить общую производительность целевой системы, являющейся частным случаем системы, описываемой в настоящем техническом решении или созданной с использованием технологий, описываемых в рамках настоящего технического решения.

На корневых узлах (корневых контроллерах) **121** регистрируются все известные переменные (теги, сигналы), используемые в автоматизируемой системе. Регистрация переменных осуществляется путем ввода проектных данных в автоматизированную систему в любой из доступных корневых узлов **121**, далее они автоматически синхронизируются, в частности, друг с другом (между собой), например, если на одном из узлов изменилось значение какой-либо переменной, оно должно обновиться на всех узлах, причем в частном случае рассылаются сообщения, содержащие такие изменения, которые доставляются на синхронизируемые узлы. Данные переменных кэшируются на корневых узлах **121**, где под кэшированием понимается запись переменных в узловой узел **121**, в связи с тем, что ранее в системе (на контроллере **111**, датчике **101**) уже эти переменные встречались (присутствовали). Кэширование изменений переменных происходит путем обновления их текущих (актуальных) значений, зарегистрированных в корневом узле (в частном случае, переданных на корневой узел) **121**, попадающих в него из других, не корневых узлов (клиентских узлов), в частном случае, из контроллеров (**111**).

На **ФИГ. 3** и **ФИГ. 4** показаны примерные варианты обмена данными, в частности, на **ФИГ. 3** показан примерный вариант обмена данными корневых узлов с клиентами (не корневыми узлами, то есть клиентскими узлами, которыми могут являться контроллеры

(**111**)), на **ФИГ. 4** показан примерный вариант обмен данными между корневыми узлами (в частности, синхронизация).

Для обмена данными в описываемой системе используются очереди сообщений. В частном случае, в системе может быть выделено четыре вида очередей сообщений, применяемых для задач обработки обмена данными между узлами:

- очередь приемная корневого узла **121** от источников **203**, в частности, от клиентов (**ФИГ. 3**) с использованием очереди сообщений (**213**);

- очередь, передающая от корневого узла клиентам;

- очередь приемная **224** синхронизации корневых узлов (**121**);

- очередь, передающая **214** синхронизации корневых узлов (**121**).

Очереди сообщений могут обрабатываться различными способами) с точки зрения приоритета), одним из основных (базовых) которых является FIFO (от англ. First Input First Output), предполагая, что очереди не будут задерживаться. При задержках могут использоваться варианты обработки очередей согласно разнообразным приоритетам (как описано далее). Так, например, может применяться способ LIFO (от англ. Last Input First Output), предполагая, что первичны и важнее свежие (последние данные), а все устаревшие обрабатываются с приоритетом ниже.

На **ФИГ. 5** показан примерный вариант обмена данными множества корневых узлов (**121**) с клиентами, в частности источниками (**203**).

Корневых узлов (**121**) может быть один, два и более, в зависимости от сложности и масштабности объекта автоматизации. Таким образом может осуществляться функция резервирования данных, или самих узлов целиком.

На **ФИГ. 6** показан примерный вариант обмена данными корневых узлов (**121**) с клиентами, в частности источниками (**203**), с резервированием значений на корневых узлах (**121**).

Резервирование данных может осуществляться путем резервирования значений на узлах, в частности, корневых узлах (**121**), с сохранением копии на дополнительном узле (узлах), двух, трех и более. Количество узлов задается (в частности, определяется) на основе требований (рекомендаций) проектировщика и опыта администратора автоматизированной системы.

На **ФИГ. 7** показан примерный вариант распределенной записи данных в несколько корневых узлов (**121**).

В частном случае, описываемое техническое решение использует (в том числе для ускорения обработки данных внешних источников) распределенную запись в несколько

корневых узлов (**121**), что в свою очередь, в частном случае, обеспечивает существенное ускорение обработки информации. Часть данных записываются на один корневой узел (**121**), другая часть - на второй корневой узел.

На **ФИГ. 8** показан примерный вариант распределенной записи данных в несколько корневых узлов (**121**) с синхронизацией (**808**) корневых узлов (**121**).

В частном случае, описываемое техническое решение использует распределенную запись в несколько корневых узлов (**121**), так что часть данных записываются на один корневой узел (**121**), другая часть - на второй корневой узел, при этом, асинхронно и параллельно обработке клиентских очередей сообщений, корневые узлы (**121**) могут синхронизироваться (**808**), в частности, могут быть синхронизированы.

На **ФИГ. 9** показан примерный вариант синхронизации кэшей с максимально низкой задержкой, согласно одному из вариантов осуществления настоящего технического решения.

В частном случае, предлагаемая система выполнена с возможностью обеспечения функционала синхронизации кэшей, с максимально низкой задержкой. При этом задержки синхронизации кэшей могут (в частном случае, должны) быть ниже, чем задержки обработки клиентских очередей сообщений (**909**), для чего выбирается (задается) соответствующая аппаратную конфигурация, в которой скорость шины серверного интерконнекта, по которой осуществляется синхронизация кэшей, будет выше или равна суммарной скорости всех клиентских соединений (портов). В частном случае синхронизация кэшей осуществляется в асинхронном режиме.

В отношении топологии корневых узлов: корневые узлы могут находиться в различных связях друг с другом (могут быть связаны различными способами). Пример простейших схем из двух узлов:

- основной (мастер, от англ. master) - резервный (слейв, от англ. slave);

- мастер - мастер.

Предлагаемая система выполнена с возможностью управления более сложными схемами (топологиями), в том числе для обработки высоких нагрузок и повышения надежности целевых систем:

- основной – несколько резервных (мультислейв);

- несколько основных (мультимастер);

- несколько основных – несколько резервных.

В контексте метрик (диагностических показателей), предлагаемое решение обеспечивает выполнение диагностики доступных корневых узлов, в том числе их

загруженности. Для описания диагностических показателей (метрик) задаются диагностические параметры, которые, в свою очередь, используются при балансировке нагрузки системы. Диагностические показатели могут использоваться в проектных сценариях, используемых для управления поведением системы.

В частном случае могут использоваться различные, как системные (заранее встроенные), так и проектные метрики (задаваемые пользователем в проекте, в частности, в целевой системе, причем проектные данные целевой системы задаются в конфигурации, которая хранится, например, в формате файлов конфигурации, в СУБД и т. д.). Например, определены системные метрики для систем (псевдо) реального времени:

- дедлайн (deadline) — предельный задаваемый срок операции, в частном случае операции с данными, например, доставка и обработка данных, сообщений, где сроком операции в частном случае является, например, время прохождения и/или обработки данных;

- латентность (latency) — время отклика операции (время задержки (реакции));

- джиттер (jitter) — разброс значений времени отклика, являющийся показателем качества работы системы реального времени, дисперсией времени реакции, задержек.

В качестве проектных метрик, например, определяются метрики качества данных. Например, накопительная метрика изменения значения переменной. Данная метрика используется (служит) для оптимизации нагрузки целевых систем. По умолчанию для каждого значения метрики устанавливается в значение «0». Если значение переменной не меняется с обновлением данных, значение метрики увеличивается на проектно задаваемый шаг (например от 0 до 1 (шаг 0.1); от 0-100 (шаг 1)), до достижения предельного значения (например «1»), при достижении которого, обновляется только метка времени переменной (на текущую, в частности, если переменная не менялась, то изменяется время, чтобы не тратилось время на синхронизацию того-же самого; в частности, если не осуществляется слежение за временем, в какой то момент «забывается» истинный статус ее, например включен ли выключатель или нет), но не само его значение, что снижает объем операций при обновлении данных, когда их много. Накопительная метрика изменения переменной, может уточняться с использованием различных статистических методов, например расчетом дисперсии, применением фильтра Калмана, применение методов машинного обучения обработки сигналов и специальных аппаратных DSP-устройств.

Касательно консистентности корневых узлов, описываемое техническое решение обеспечивает консистентность данных корневых узлов при их синхронизации (синхронизации кэшей). Консистентность обеспечивается различными способами, в

зависимости от режима работы корневых узлов, с той или иной эффективностью скорость синхронизации, количество ошибок, повторений отправки. Синхронизация кэшей обеспечивается путем рассылки изменений значений переменных различными способами, в зависимости от режима работы корневых узлов (unicast, multicast, broadcast). Таким образом обеспечивая на всех синхронизируемых узлах максимально актуальную информацию на текущий момент времени.

На **ФИГ. 10** показан примерный вариант синхронизации переменных с более высокими метриками приоритета, далее в порядке уменьшения этого приоритета, согласно одному из вариантов осуществления настоящего изобретения.

При обеспечении консистентности используются (применяются) схемы приоритета синхронизации (**1020**) кэшей. Для этого используются специальные параметры переменных, которые задают приоритет их обработки - метрики приоритета (метрики значимости). Так, для переменной «v4» метрика приоритета (метрика значимости) равна единице («1»), для переменной «v3» метрика приоритета (метрика значимости) равна четырем («4»), для переменной «v2» метрика приоритета (метрика значимости) равна трем («3»), для переменной «v1» метрика приоритета (метрика значимости) равна двум («2»). Сначала синхронизируются переменные с более высокими метриками приоритета, далее в порядке уменьшения этого приоритета. Таким образом в информационном потоке (в частном случае, потоке данных), не затеряются критически важные изменения, такие как, например команды управления, критические значения переменных и др. Проектно задаваемые метрики приоритета переменных, могут быть заменены на аналогичные расчетные метрики качества (например, средняя скорость обработки сообщений на все ядра одного узла; или, например, общая средняя скорость обработки переменных, например 10 миллионов на узел в секунду; или, например при использовании разных процессоров и сетевых карт с различной скоростью метрика может являться относительной скоростью сетевой узловой обработки данных на ядро процессора), причем такая схема может применяться в менее критических системах управления.

В классической схеме в автоматизированной системе управления (АСУ) используются два узла, «основной – резервный», «основной – основной». В первом случае, всю нагрузку обрабатывает основной узел, а при его отказе, резервный узел становится основным, и берет на себя всю нагрузку. Во втором случае обрабатывать нагрузку могут два узла одновременно, при отказе одного из них, оставшийся берет (принимает) на себя всю нагрузку. Стоит отметить, что в такой схеме, при отказе одного узла, резервных узлов

не остается без ручного вмешательства. Синхронизация кэшей (кешей) осуществляется путем однонаправленной передачи данных (unicast).

На **ФИГ. 11** показан пример синхронизации кешей путем однонаправленной передачи данных, мультивещанием (multicast), в режиме функционирования (работы) корневых узлов «основной – несколько резервных», согласно одному из вариантов осуществления настоящего технического решения.

В режиме функционирования (работы) корневых узлов «основной – несколько резервных» при отказе или проблемах на основном узле, один из резервных узлов становится основным (выбирается любой, или согласно заранее заданному приоритету), при этом остаются резервные узлы. Синхронизация кэшей осуществляется путем однонаправленной передачи данных, мультивещанием (multicast).

На **ФИГ. 12** показан пример синхронизации кэшей посредством мультивещания без резервных узлов в режиме работы корневых узлов «несколько основных (мультимастер)», согласно одному из вариантов осуществления настоящего технического решения.

В режиме работы корневых узлов «несколько основных (мультимастер)» при отказе или проблемах на одном узле, другие узлы обеспечивают обслуживание систем, узлов становится меньше, нагрузка распределяется между узлами и на каждый из них растет. Резервных узлов нет. Синхронизация кэшей осуществляется мультивещанием.

На **ФИГ. 13** показан пример синхронизации кэшей посредством мультивещания в режиме работы корневых узлов «несколько основных – несколько резервных (мультимастер - мультислейв)», согласно одному из вариантов осуществления настоящего технического решения.

В режиме работы корневых узлов «несколько основных – несколько резервных (мультимастер - мультислейв)» при отказе (**1318**) или проблемах на одном (или нескольких) корневом узле, остальные корневые узлы обеспечивают обслуживание систем. Рабочих (основных) корневых узлов (**1313**) при этом меньше не становится, нагрузка перераспределяется между корневыми узлами, не вызывая рост нагрузки на каждом отдельном корневом узле. Резервных (**1316**) узлов становиться меньше. Синхронизация кэшей – мультивещание.

В частном случае режим работы узлов мультимастер обеспечивает ряд преимуществ для распределенных систем управления с большим количеством устройств, и клиентов.

Режим работы узлов мультимастер подразумевает проектную избыточность корневых узлов, которые разделяют общую нагрузку на систему в целом, часть из которых могут находиться в резерве.

Управление нагрузкой корневыми узлами обеспечивает специальные балансировщики нагрузки (**1010**), которые по заданному алгоритму осуществляет распределение обработки данных между узлами (в частности, балансировку нагрузки).

В частном случае выделяется два вида балансировщиков нагрузки (балансировщики): балансировщик очередей клиентов (балансировщик клиентов, балансировщик клиентских очередей), который управляет приемными и передающими клиентскими очередями, как показано на **ФИГ. 14**, и балансировщик очередей синхронизации (балансировщик синхронизации), который управляет очередями синхронизации корневых узлов, как показано на **ФИГ. 15**.

На **ФИГ. 14** показан пример управления приемными и передающими клиентскими очередями балансировщиком очередей клиентов (**1414**), согласно одному из вариантов осуществления настоящего технического решения.

На **ФИГ. 15** показан пример управления очередями синхронизации корневых узлов балансировщиком очередей синхронизации (**1515**), согласно одному из вариантов осуществления настоящего технического решения.

Балансировщик нагрузки клиентских очередей может выполняться на специально выделенном узле, как показано на **ФИГ. 16**, или нескольких узлах, как показано на **ФИГ. 17**. Балансировщик может быть встроен в каждый корневой узел, в случае кластера, как показано на **ФИГ. 18**. Конфигурация задается администратором целевой системы.

На **ФИГ. 16** показан вариант выполнения балансировщика нагрузки клиентских очередей (**1616**) на специально выделенном узле, согласно одному из вариантов осуществления настоящего технического решения.

На **ФИГ. 17** показан вариант выполнения балансировщика (**1616**) нагрузки клиентских очередей на нескольких специально выделенных узлах, согласно одному из вариантов осуществления настоящего технического решения.

На **ФИГ. 18** показан вариант встроенного балансировщика в каждый корневой узел в случае кластера, согласно одному из вариантов осуществления настоящего технического решения.

В частном случае балансировщик очередей синхронизации (**1515**) встроен в каждый корневой узел (**121**), и не может быть использован отдельно, в виде отдельного узла.

На **ФИГ. 19** показан примерный вариант обеспечения эффективного управления очередями сообщений балансировщиком нагрузки, согласно одному из вариантов настоящего технического решения.

В частном случае алгоритм балансировки задается на этапе технического проектирования, например, каждое новое значение переменной записывается на следующий по списку узел, инкрементно, при достижении максимального значения, инкремент начинается заново. В алгоритме могут учитываться загруженность корневых узлов, близость источника данных и их потребителя к корневому узлу. Балансировщик (**1010**) нагрузки (**1919**), сформированной из узла 1 (**2021**), узла 2 (**2022**) и узла 3 (**2023**), обеспечивает эффективное управление очередями сообщений.

Балансировщик нагрузки, может учитывать метрики приоритета и качества. В таком случае все данные в очередях будут обрабатываться в соответствии с приоритетом, сначала будут обрабатываться более важные данные, а потом менее важные. Балансировщик может учитывать загруженность узлов. Могут использоваться различные схемы работы с метриками приоритета и нагрузки.

На **ФИГ. 20** показан примерный вариант балансировки аппаратных ресурсов, в частности, схема балансировки узлов, нацеленная на равномерное использование аппаратных ресурсов, где все переменные направляются в обработку на наименее загруженный узел до тех пор, пока его нагрузка не станет равна предельному значению, например значению максимально нагруженного узла, по умолчанию.

В приведенном на **ФИГ. 20** примере случае переменной «v9» соответствует приоритет, равный «1» ,переменной «v8» соответствует приоритет, равный «4», переменной «v7» соответствует приоритет, равный «1», переменной «v6» соответствует приоритет, равный «2», переменной «v5» соответствует приоритет, равный «3», переменной «v4» соответствует приоритет, равный «1», где «4» - наиболее значимое. Так, в приведенном на **ФИГ. 20** примере случае значение переменной «v8», в порядке наивысшего приоритета («4»), направляется на обработку в наименее загруженный третий узел (**2023**), Далее, значение переменной «v5», в порядке следующего приоритета («3»), направляется на обработку в наименее загруженный второй узел (**2022**), выбирается в порядке увеличения идентификатора узла или случайно. Таким же образом обрабатывается значение «v6» (с приоритетом «2») на третьем узле (**2023**). Все переменные, направляются в обработку на наименее загруженный узел, до тех пор, пока его нагрузка не станет равна предельному значению (например, значению, максимально нагруженному узла, по умолчанию).

На **ФИГ. 21** показан примерный вариант балансировки аппаратных ресурсов и обработки критических данных, согласно одному варианту осуществления настоящего технического решения.

В приведенном на **ФИГ. 21** примере случае переменной «v9» соответствует приоритет, равный «1» ,переменной «v8» соответствует приоритет, равный «4», переменной «v7» соответствует приоритет, равный «1», переменной «v6» соответствует приоритет, равный «2», переменной «v5» соответствует приоритет, равный «3», переменной «v4» соответствует приоритет, равный «1», где «4» - наиболее значимое.

Схема балансировки узлов, нацеленная на равномерное использование аппаратных ресурсов, но при этом максимально быстрой обработки критических значений переменных. В примере, приведенном на **ФИГ. 21**, значение переменной «v8», в порядке наивысшего приоритета («4»), направляется на обработку в наименее загруженный третий узел (**2023**). Далее, значение переменной «v5», в порядке следующего приоритета («3»), направляется на обработку в наименее загруженный второй узел (**2022**). Таким же образом обрабатывается значение «v6» на первом узле (**2021**). Далее, переменные, направляются в наименее загруженный узел, до тех пор, пока его нагрузка не станет равна предельному значение (например, значению, максимально нагруженному узла, по умолчанию).

На **ФИГ. 22** показан примерный вариант балансировки при параллельных вычислениях, в частности, показана схема балансировки узлов, нацеленная на поддержание эффективности параллельных вычислений.

В приведенном на **ФИГ. 22** примере случае переменной «v9» соответствует приоритет, равный «1» ,переменной «v8» соответствует приоритет, равный «4», переменной «v7» соответствует приоритет, равный «1», переменной «v6» соответствует приоритет, равный «2», переменной «v5» соответствует приоритет, равный «3», переменной «v4» соответствует приоритет, равный «1», где «4» - наиболее значимое.

В примере, приведенном на **ФИГ. 22**, в частности, в случае схемы балансировки узлов, нацеленной на поддержание эффективности параллельных вычислений, алгоритм попытается максимально распределить обработку переменных по узлам, но начнет с наименее загруженного узла. В данном случае значение переменной «v8», в порядке наивысшего приоритета («4»), направляется на обработку в наименее загруженный третий узел (**2023**). Далее, значение переменной «v5», в порядке следующего приоритета («3»), направляется на обработку в наименее загруженный второй узел (**2022**). Таким же образом обрабатывается значение «v6» на первом узле (**2021**). Далее, переменные, имеющие равный приоритет, обрабатываются в порядке обычной очереди в приоритете загрузки узлов.

На **ФИГ. 23** показан примерный вариант параллельных вычислений, согласно одному из вариантов осуществления настоящего технического решения.

Описываемая система выполнена с возможностью поддерживания режима объединения корневых узлов, работающих в режиме мультимастера в вычислительный кластер. Для этого корневые узлы должны обладать, одинаковой (или примерно одинаковой, приблизительно одинаковой и т. д.) аппаратной конфигурацией (процессоры, оперативная память, сетевые компоненты и т. д.). В частном случае перечень кластерных узлов задает (определяет) администратор системы вручную. В таком случае кластер может решать типичные для кластеризации задачи, в том числе задачи параллельных вычислений. Выбор и конфигурацию режима работы кластера осуществляет пользователь.

При достаточном объеме кластерных узлов, описываемое техническое решение позволяет формировать (построить) эффективные сценарии (в частности, алгоритмы) параллельных вычислений, например задач, которые нужно вычислить (считать) в режиме реального времени. Такими задачами могут быть, например типичных задач для АСУ ТП (автоматизированных систем управления технологическим процессом), вычисления плавающих уставок, аппроксимации, прогнозирования и т. д.

Предлагаемая система выполнена с возможностью обеспечения синхронизации кластерных сценариев. Кластерный сценарий является интегрированной в узел (корневой узел) программой, которая может использовать прикладной программный интерфейс окружения (в т. ч. операционной системы). Сценарий может выполняться узловым вычислителем, например, в скомпилированном виде (в частности, в виде сгенерированных кодов), так что осуществляется соблюдение всех оптимизаций для работы (функционирования) системы в максимально быстром режиме. Такие сценарии могут быть синхронизированы кластером, их текст или бинарный код, как переменные. Актуальный код сценариев, копируется на все узлы без остановки работы кластера. Если в текущий момент времени узел обрабатывает тот или иной сценарий, то ожидается его завершение, осуществляется замена кода, и перезапускается расчет при достижении консистентности синхронизации сценариев.

При отказе узлов, и наличии резервных, кластерный сценарий без остановки продолжит осуществлять вычисления на этих узлах без задержек и потери производительности, а клиенты, владельцы сценария, будут переподключены к этим узлам. Такая схема, в частном случае реализует модель распределённых вычислений (например, MapReduce), в случае отказа узлов, система или по меньшей мере одна ее часть не останавливается на перерасчет, ожидая восстановление утраченных узлов.

Узловой сценарий может быть описан в виде внешнего скомпилированного приложения, сценария на транслируемом языке, доступным в окружении, для чего может

быть использован прикладной программный интерфейс корневого узла. В некоторых случаях, сценарии могут быть вынесены в клиентские узлы, для чего используется прикладной программный интерфейс клиентского узла.

Узловые сценарии могут работать и в обычном, не кластерном режиме.

На **ФИГ. 24** показан примерный вариант указания переменных для узлов с целью синхронизации, согласно одному из вариантов настоящего технического решения.

На мультимастере синхронизация кэшей может осуществляться не целиком, а частично, в частном случае снижая общую нагрузку на каждый узел и, следовательно, на систему (в частном случае, сеть) в целом. Параметры частичной синхронизации могут быть заданы вручную или автоматически. Вручную могут быть указаны какие переменные (или группа переменных) с каким узлом должны синхронизироваться. Таким образом, может быть определен шардинг.

На **ФИГ. 25** показан примерный вариант сегментации переменных на корневых узлах, согласно одному из вариантов осуществления настоящего технического решения.

Автоматическая синхронизация может осуществляться в связке с балансировщиком нагрузки с формированием (выстраиванием) оптимальной схемы хранения переменных в зависимости от близости источника данных и (или) их потребителя к корневому узлу. Переменные «v1», «v2», необходимые для работы первому клиенту (**2515**), сегментированы на первом корневом узле (**2511**) и втором корневом узле (**2512**). В свою очередь, переменные «v3», «v4», необходимые для работы второму клиенту (**2525**), сегментированы на третьем узле (**2513**) и на четвертом узле (**2514**).

На **ФИГ. 26** показан примерный вариант сегментации переменных на корневых узлах в случае отказа одного из корневых улов, согласно одному из вариантов осуществления настоящего технического решения.

В случае отказа одного корневого узла, например, первого корневого узла (**2511**), нужные первому клиенту (**2515**) копии (реплики) данных (переменных), остались в единственном экземпляре на втором корневом узле (**2512**). В случае отсутствия в системе «горячего» резерва (заменяющего устройства), для продолжения обеспечения резервирования данные могут быть скопированы на (ближайший) третий корневой узел (**2513**), на котором, в свою очередь, в случае утраты второго корневого узла (**2512**), будут находиться данные, необходимые первому клиенту (**2515**).

На **ФИГ. 27** показан примерный вариант переключения на другой корневой узел без задержек с сохранением подписки отказе корневого узла, согласно одному из вариантов осуществления настоящего технического решения.

Доступ к корневым узлам осуществляется путем специализированных программных интерфейсов, в частном случае один из которых обеспечивает взаимодействие с корневыми узлами в режиме корневого узла, другой в режиме клиентского узла, подключенному к корневому узлу (мультимастеру, кластеру).

Прикладной программный интерфейс корневого узла обеспечивает возможность собрать (в частности, скомпилировать, сформировать и т. д.) компонент программного обеспечения (например, библиотеку, модуль, программу и т. д.), который будет содержать функционал корневого узла системы, и другие узлы (в частности, не корневые), входящие в состав компонента, будут иметь прямой доступ к данным этого корневого узла путем синхронизации, с поддержкой консистентности.

Прикладной программный интерфейс клиентского узла обеспечивает возможность собрать компонент программного обеспечения (библиотеку, модуль, программу), который будет взаимодействовать с корневыми узлами путем подписки на изменения ограниченного набора данных. При отказе корневого узла переключение на другой корневой узел осуществляется без задержек, подписка на изменения сохраняется.

На **ФИГ. 28** показан примерный вариант переключения клиента при перераспределении нагрузки между корневыми узлами с использованием балансировщика нагрузки, согласно одному из вариантов осуществления настоящего технического решения.

Интерфейс клиентского узла обеспечивает возможность осуществить переключение клиента при перераспределении нагрузки между корневыми узлами, например к наименее нагруженному, с помощью балансировщика нагрузки.

В примерном варианте осуществления (в частности, базового примера) может быть приведено предприятие генерации, например, тепловая электростанция (ТЭС). Так, в 1980 году была построена ТЭС с двумя энергоблоками с мощностью по 100 МВт. ТЭС была автоматизирована контроллерами и устройствами телемеханики, органы диагностики и управления были выведены на блочный щит управления в виде аналоговых показометров (световые индикаторы, стрелочные приборы) и механических средств (кнопки, переключатели, тумблеры). Первый блок (Б1) был введен сразу. В связи с задержкой ввода в эксплуатацию второй блок (Б2) был выведен позже, в 1982 году, и в связи с изменением на рынке компонентов, средства автоматизации на нем отличаются от первого. В 2000 году было принято решение достроить еще один блок мощностью 200 МВт. В качестве средств автоматизации было принято решение использовать программно-логические контроллеры, а вывод диагностики и управления на операторские станции с графическим пользовательским интерфейсом и реализованными в нем встроенными элементами

управления. В 2020 году было принято решение модернизировать первые два блока, а с ними и средства автоматизации, и для унификации на третьем блоке для единообразия.

Важным требованием при проектировании модернизированной АСУ была заявлено возможность резервирования диагностики и управления энергоблоками, при отказе систем верхнего уровня на уровне блоков и отдельных подсистем, например отказе блочного пульта целиком. Такое резервирование необходимо не только для предотвращения потенциальных аварийных инцидентов, но и для возможности передачи управления между блочными пультами, в случае проведения ремонтов и иных ситуаций. Таким образом, в случае отказа одной из подсистем верхнего уровня энергоблока, или целиком, оператор должен иметь возможность перейти на блочный пульт управления другого блока, и используя его часть, перевести на него управление отказавшие системы, при этом не нарушив управляемость АСУ того блока, на который переводят управление.

Оказалось, что для экономической эффективности, не теряя при этом инженерный функционал, выгоднее заменить на первых двух блоках все средства автоматизации, а на третьем только верхний уровень, т. к. низовая автоматика, в целом работает нормально, и необходимо только обновить часть контроллеров, не меняя топологии сети. В качестве контроллеров низовой автоматики первого и второго блока было принято решение закупить решения не одного вендора а нескольких, так как с точки зрения экономической эффективности такое решение выгоднее, чем автоматизировать все одним вендором. Для того, чтобы осуществить данный план, требуется универсальное решение верхнего уровня. В качестве такого решения для проекта может использоваться предлагаемое техническое решение.

В качестве примера используются (будем оперировать) три типа переменных:

- диагностические источники, данные с датчиков и контроллеров;
- расчетные переменные, вычисляемые переменные в системе;
- управляющие сигналы, команды оператора и прочие воздействия.

В частном случае приоритет обработки управляющих сигналов выше, чем диагностических, которые в свою очередь, выше расчетных, однако стоит отметить, что в различных (в том числе реальных) системах приоритеты могут распределяться по более сложному алгоритму.

Для настоящего примера на первом и втором блоках - по три тысячи диагностических сигналов, по шесть тысяч расчетных сигналов и по тысяче управляющих сигналов. На третьем блоке четыре тысячи диагностических сигналов, десять тысяч расчетных сигналов, и две тысячи управляющих сигналов, как показано в таблице 1.

Таблица 1. Распределения сигналов по блокам.

Переменные	Диагностические	Расчетные	Управляющие
Блок 1 [A]	3 000 [A1]	6 000 [A2]	1 000 [A3]
Блок 2 [B]	3 000 [B1]	6 000 [B2]	1 000 [B3]
Блок 3 [C]	4 000 [C1]	10 000 [C2]	2 000 [C3]
Итого	10 000	22 000	4 000

Таким образом, требуется тридцать шесть тысяч переменных, отвечающих за диагностику, определение состояния и управления электростанцией, как показано на **ФИГ. 29**.

На **ФИГ. 29** показан примерный вариант переменных, отвечающих за диагностику, определение состояния и управления электростанцией.

Для резервирования на каждом блоке требуется разместить по два сервера (основной-резервный). Сравнительно базовый современный сервер, при доступности оперативной памяти, способен вместить по тридцать шесть тысяч переменных. Мощность серверов планируется таким образом, чтобы в пиковых нагрузках, а они возникают в основном при переходных режимах, не достигала пятидесяти процентов. В частном случае, резервируемые сервера, не должны превышать десяти процентов нагрузки на задачи поддержки резервирования.

В приведенном примере не учитываются оборудование подсистем верхнего уровня, таких как серверы преобразования протоколов, серверы связи с объектом и т.д, однако они также должны резервироваться. В случае отказа всех резервов такого оборудования осуществляются те же действия, что и в рассматриваемом примере. При высокой вероятности отказа всех резервов на блоке осуществляется резервирование межблочной сетью с аналогичным оборудованием, с планированием запаса по вычислительной мощности, как показано на **ФИГ. 30**.

На **ФИГ. 30** показан примерный вариант резервирования межблочной сетью с планированием запаса по вычислительной мощности при высокой вероятности отказа всех резервов на блоке, согласно одному из вариантов осуществления настоящего технического решения.

В приведенном примере присутствует (имеется) шесть клиентских очередей сообщений обработки переменных на каждый сервер (из них три - в резерве), и шесть очередей сообщений синхронизации, как показано на **ФИГ. 31**.

На **ФИГ. 31** показан примерный вариант шести клиентских очередей сообщений обработки переменных на каждый сервер (из них три - в резерве), и шести очередей сообщений синхронизации.

В случае отказа одного основного сервера («Б1 С1»), системы верхнего уровня («Б1»), в частности, клиенты, переключаются на резервный сервер («Б1 С2»). «Б1» остается без резервного сервера.

В случае отказа двух серверов (основного «Б1 С1» и резервного «Б1 С2») системы верхнего уровня, операторские рабочие станции переключаются на резервный север «Б2 С2», при этом «Б2» остается без резервного сервера, до восстановления систем «Б1», как показано на **ФИГ. 32**.

На **ФИГ. 32** и **ФИГ. 33** показан примерный вариант в случае отказа двух серверов (основного «Б1 С1» и резервного «Б1 С2») системы верхнего уровня, операторские рабочие станции переключаются на резервный север «Б2 С2», при этом «Б2» остается без резервного сервера, до восстановления систем «Б1».

В данном случае клиентские очереди «Б2 С1» принимаются обрабатывать данные Б1. Осуществляется ввод резервной очереди «Б2 С2» для обслуживания данных «Б2», следовательно нагрузка на узел вырастает.

В случае отказа системы верхнего уровня в целом, включая два сервера (основного «Б1 С1» и резервного «Б1 С2»), включая операторские рабочие станции, оператор «Б1» может физически перейти на блочный пульт «Б2» заняв рабочее место одного из операторов «Б2», при этом «Б2» остается без резервного сервера и резервной рабочий станции оператора, до восстановления систем «Б1», как показано на **ФИГ. 34**.

На **ФИГ. 34** показан примерный вариант в случае отказа системы верхнего уровня в целом, включая два сервера (основного «Б1 С1» и резервного «Б1 С2»), включая операторские рабочие станции, с переходом оператором «Б1» на блочный пульт «Б2» заняв рабочее место одного из операторов «Б2», при этом «Б2» остается без резервного сервера и резервной рабочий станции оператора, до восстановления систем «Б1».

Предлагаемое техническое решение выполнено с возможностью сбора, обработки, хранения и предоставления доступа к данным устройств, в том числе устройств промышленной автоматизации и интернета вещей, управления ими, в том числе для объектов критической информационной инфраструктуры (КИИ), обеспечивающий высокие характеристики надежности, отказоустойчивости (катастрофоустойчивости).

В заключение следует отметить, что приведенные в описании сведения являются примерами, которые не ограничивают объем настоящего технического решения,

определенного формулой. Специалисту в данной области становится понятным, что могут существовать и другие варианты осуществления настоящего технического решения, согласующиеся с сущностью и объемом настоящего технического решения.

ФОРМУЛА

1. Система управления, сбора, обработки, хранения и предоставления доступа к данным устройств промышленной автоматизации и Интернета вещей, включает:

    - по крайней мере два датчика, каждый из которых связан по крайне мере с одним контроллером;

    - по крайней мере два контроллера, причем каждый контроллер содержит один или более обрабатываемых портов ввода/вывода представляющих собой переменные и связан с одним или более датчиком;

    - по крайней мере два корневых узла, выполненных с возможностью:

        - получения и регистрации переменных от по крайней мере одного контроллера;
        - синхронизации переменных;
        - кэширования изменений переменных путем обновления текущего значения переменной, зарегистрированной в корневом узле, попадающего в него из очереди сообщений об изменениях
        - синхронизации кэшей, путем рассылки изменений значений переменных различными способами, в зависимости от режима работы корневых узлов
        - обеспечения консистентности при синхронизации кэшей путем приоритезации за счет применения специальной маркировки переменных, которые задают значимость их синхронизации, причем сначала синхронизируются более значимые параметры, и далее в порядке приоритета значимости;
        - объединения корневых узлов, работающих в режиме мультимастера в вычислительный кластер;
        - синхронизации кластерных сценариев;

    - причем система выполнена с возможностью:

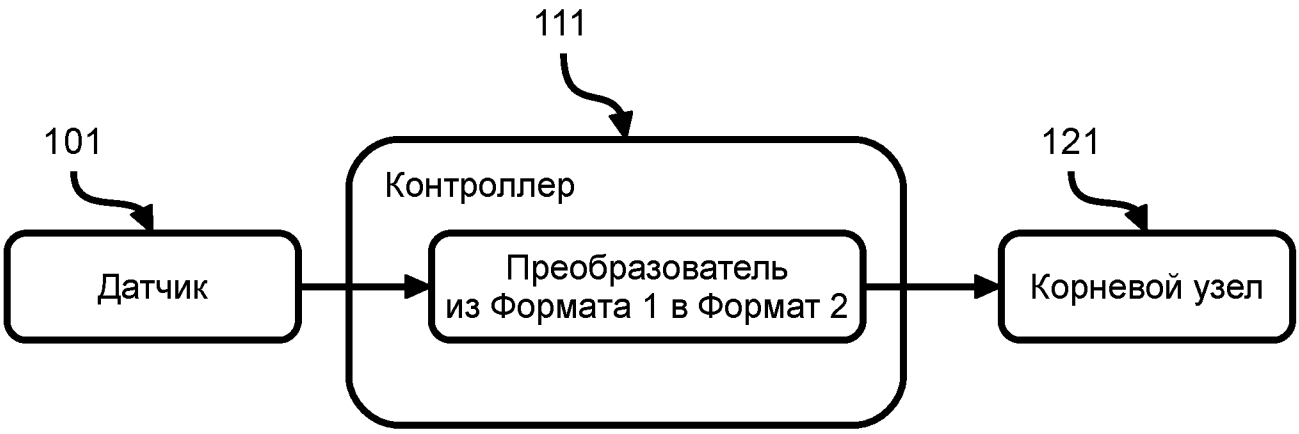
        - обмена данными между контроллерами и корневыми узлами, используя очереди сообщений;
        - резервирования значений переменных на корневых узлах;
        - диагностики доступных корневых узлов, включая загруженность.

2. Система по п.1, в которой для синхронизации кэшей с задержкой ниже, чем задержки обработки клиентских очередей сообщений используется аппаратная конфигурация, в которой скорость шины серверного интерконнекта, по которой осуществляется синхронизация кэшей, будет выше или равна суммарной скорости всех портов.

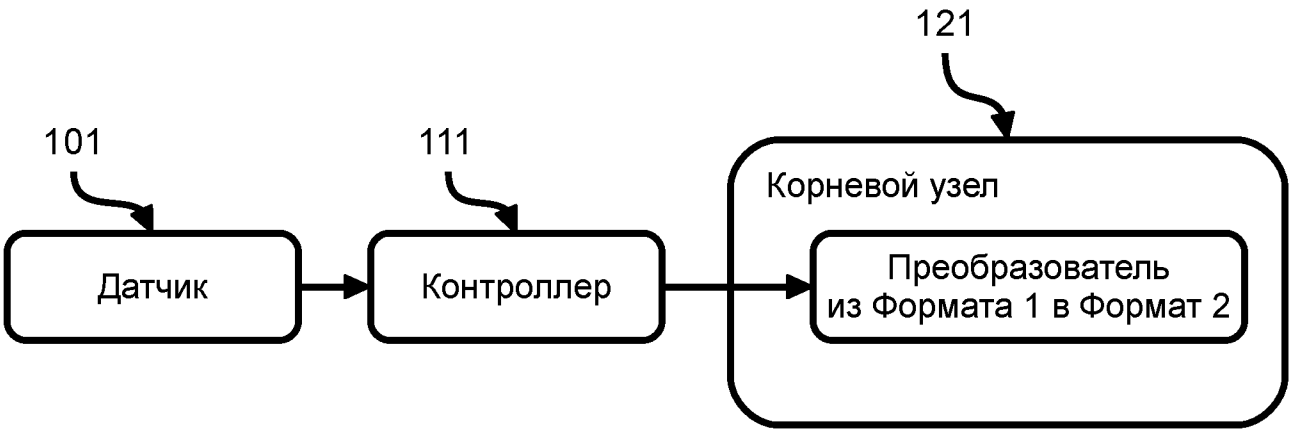
3. Система по п.1, в которой корневые узлы связаны друг с другом в режиме основной – резервный;

4. Система по п.1, в которой корневые узлы связаны друг с другом в режиме основной – основной;

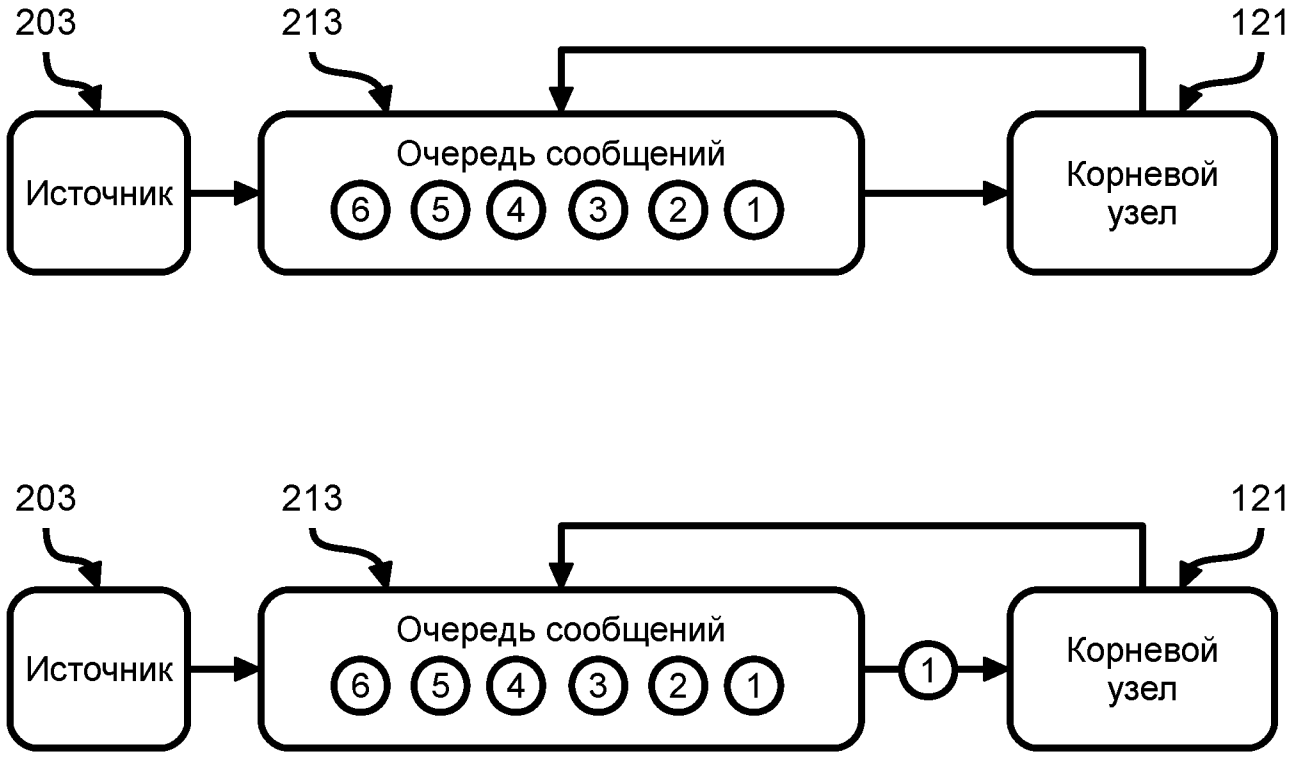
5. Система по п.1, в которой синхронизации кэшей осуществляется посредством мультивещания без резервных узлов в режиме работы корневых узлов основной-основной.


111
101
Контроллер
121
Датчик
Преобразователь
из Формата 1 в Формат 2
Корневой узел

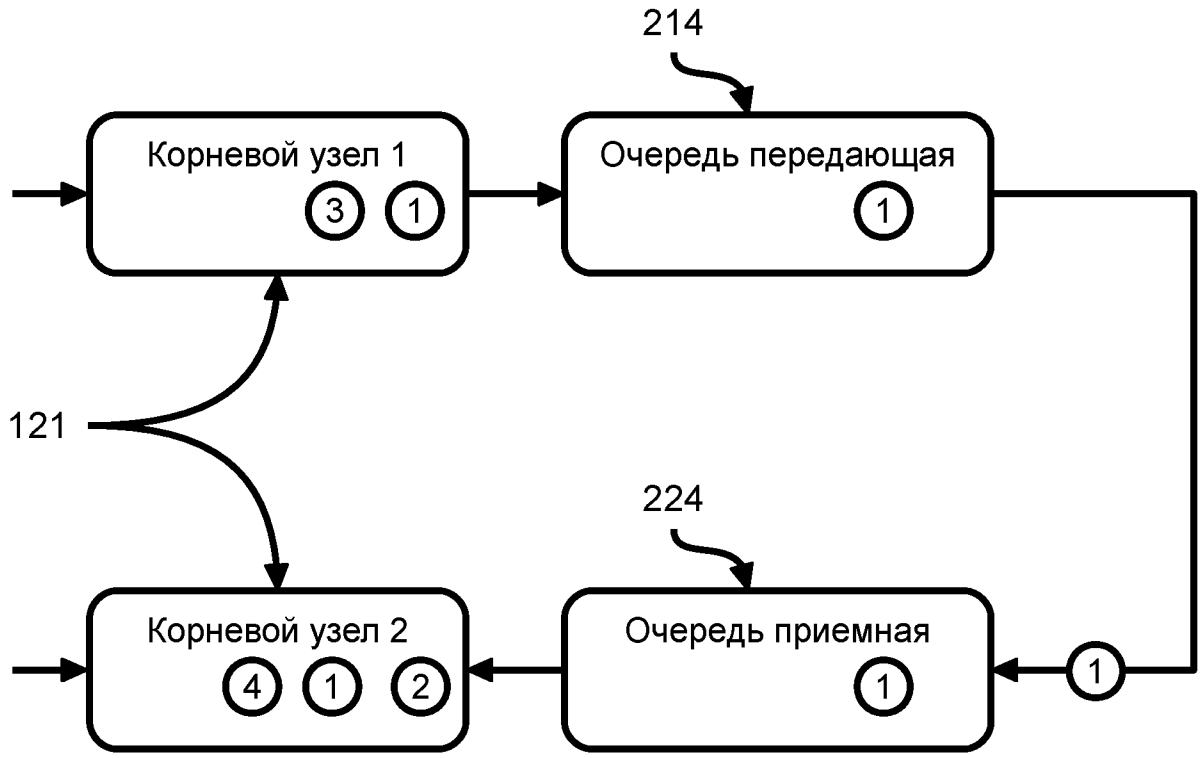
ФИГ. 1


121
101
111
Корневой узел
Датчик
Контроллер
Преобразователь
из Формата 1 в Формат 2

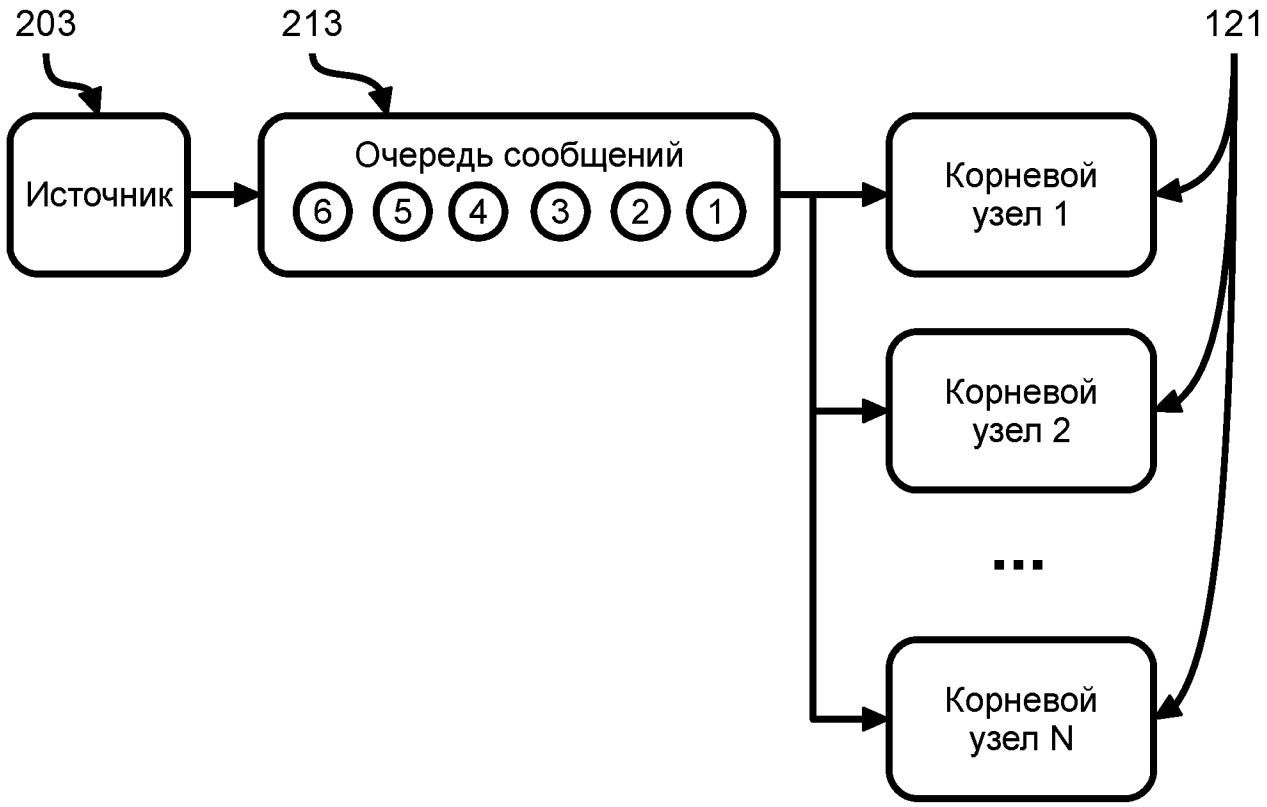
ФИГ. 2


203
213
121
Источник
Очередь сообщений
6
5
4
3
2
1
Корневой узел
203
213
121
Источник
Очередь сообщений
6
5
4
3
2
1
1
Корневой узел

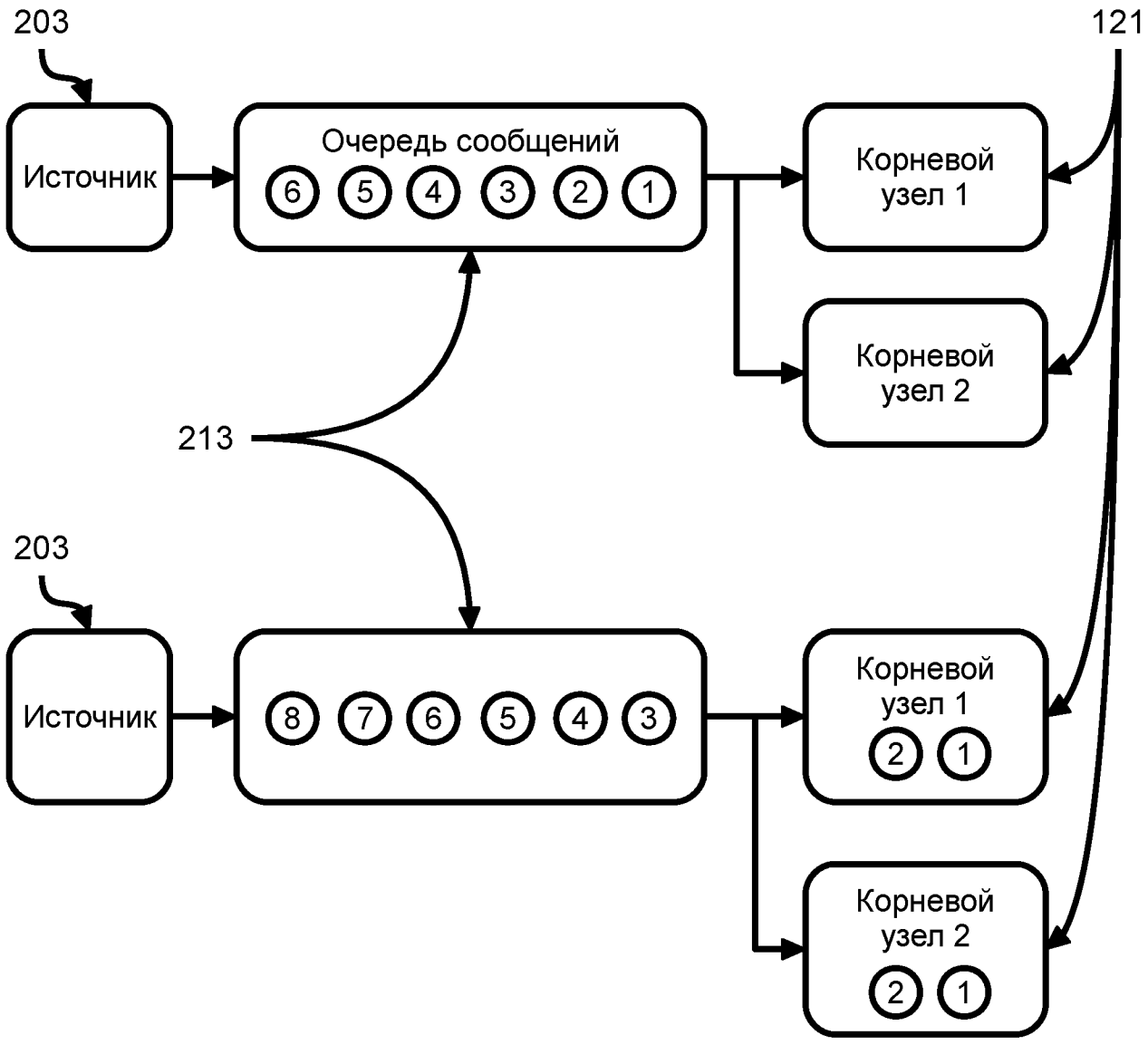
ФИГ. 3


214
Корневой узел 1
3
1
Очередь передающая
1
121
224
Корневой узел 2
4
1
2
Очередь приемная
1
1

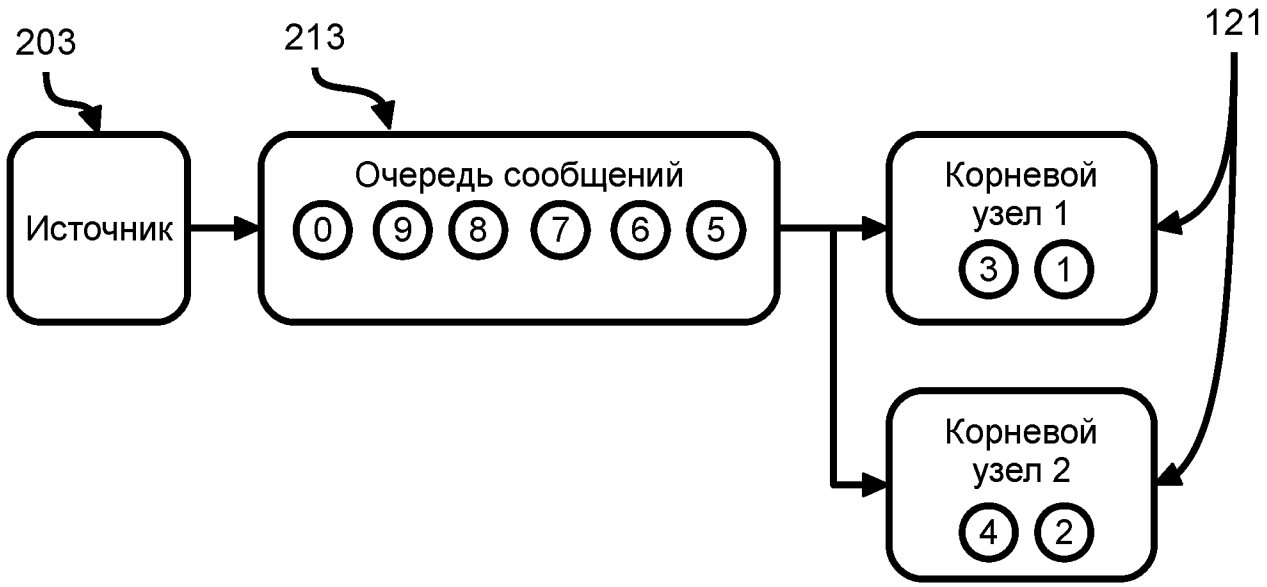
ФИГ. 4


203
213
121
Источник
Очередь сообщений
6
5
4
3
2
1
Корневой узел 1
Корневой узел 2
...
Корневой узел N

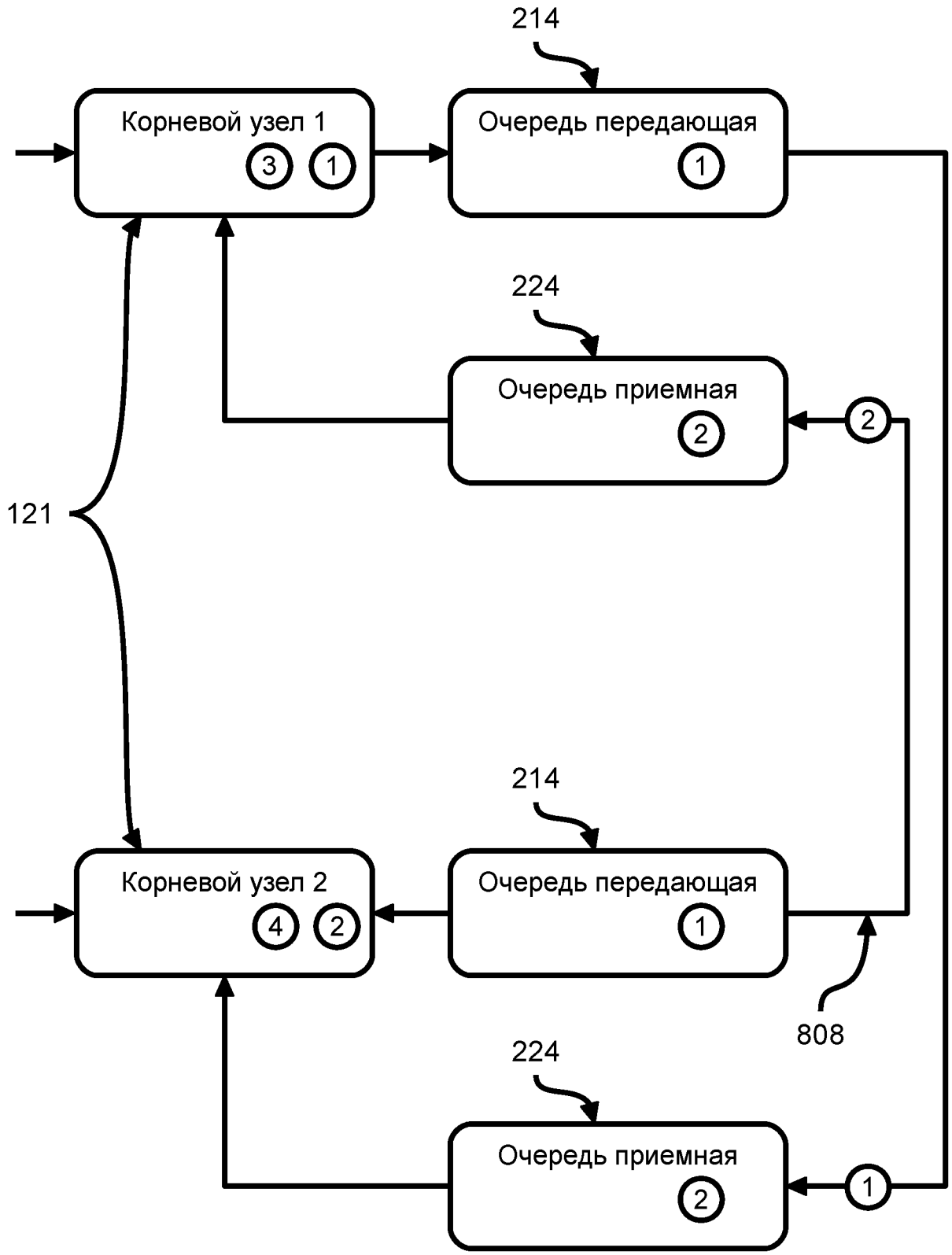
ФИГ. 5


203
121
Источник
Очередь сообщений
6
5
4
3
2
1
Корневой узел 1
Корневой узел 2
213
203
Источник
8
7
6
5
4
3
Корневой узел 1
2
1
Корневой узел 2
2
1

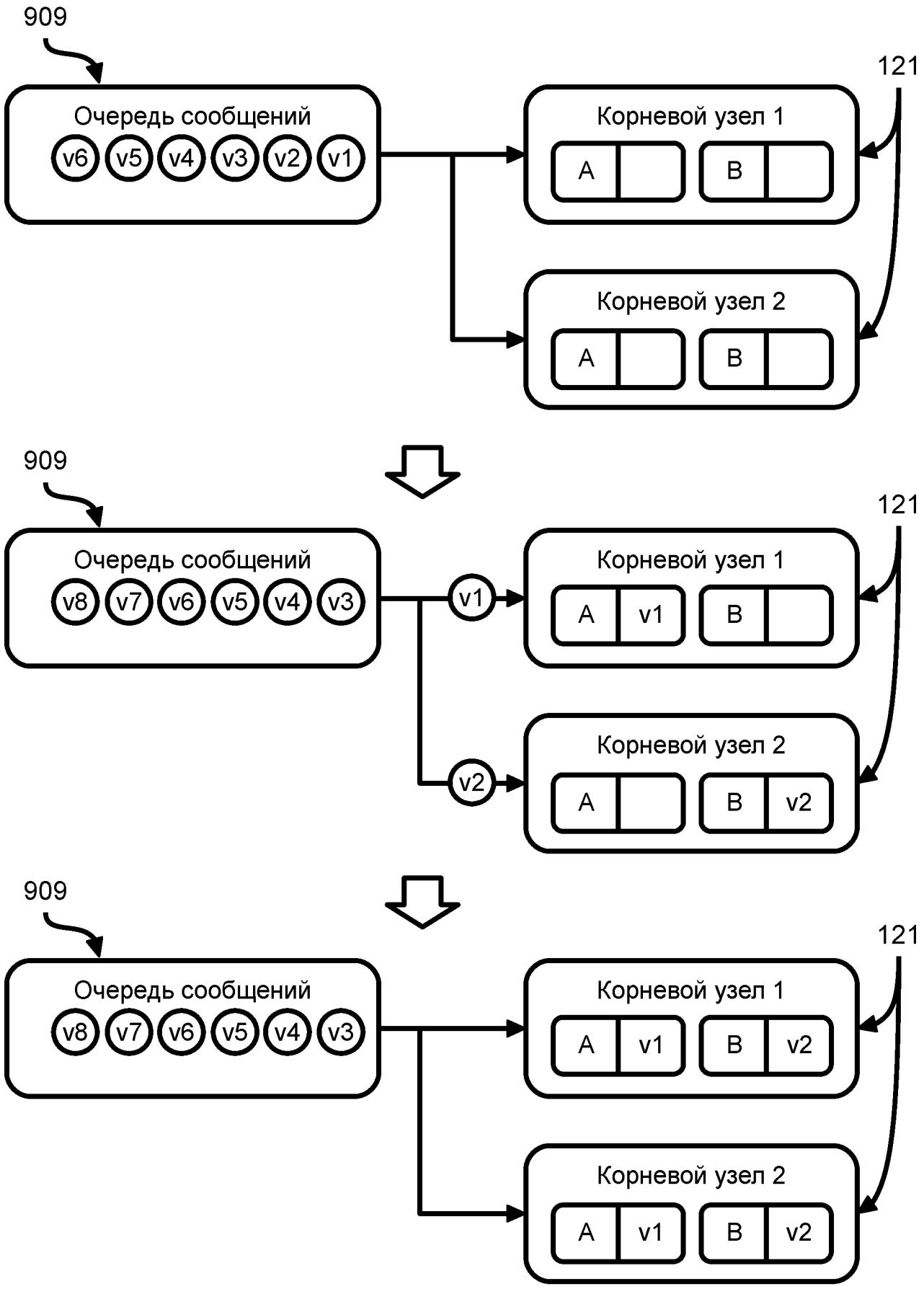
ФИГ. 6


203
213
121
Источник
Очередь сообщений
0
9
8
7
6
5
Корневой
узел 1
3
1
Корневой
узел 2
4
2

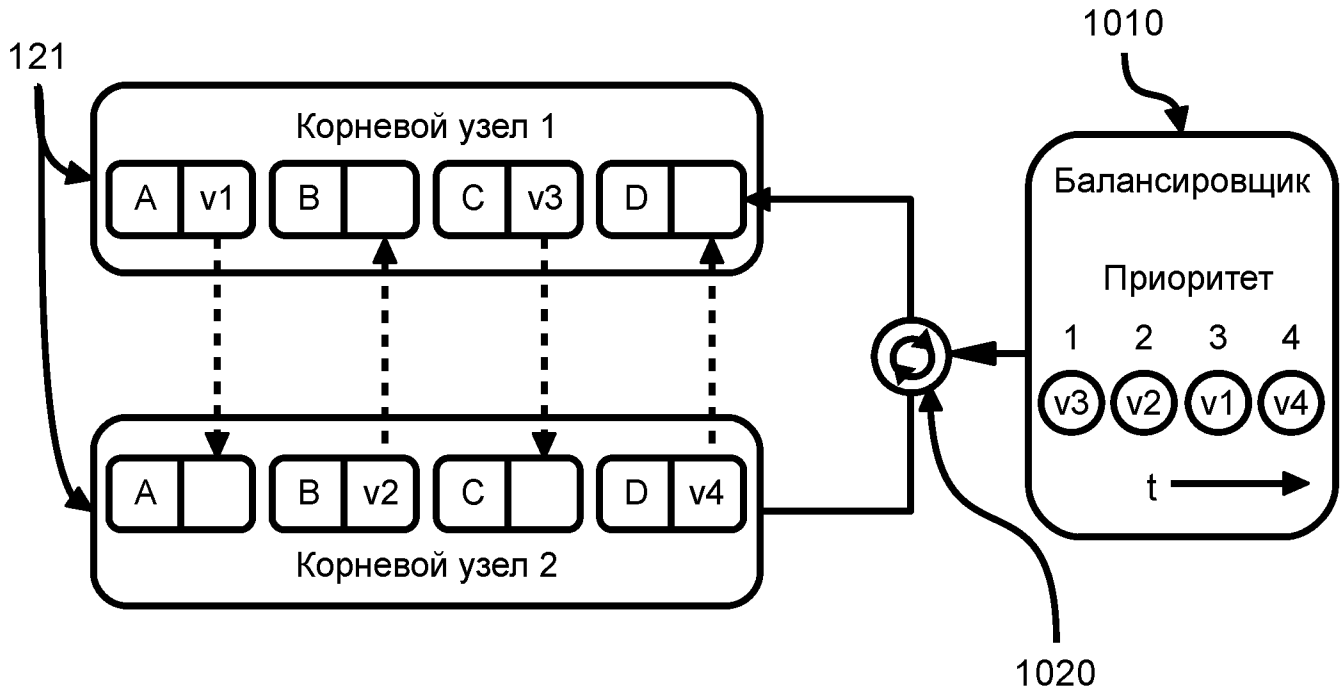
ФИГ. 7


214
Корневой узел 1
3
1
Очередь передающая
1
224
Очередь приемная
2
2
121
214
Корневой узел 2
4
2
Очередь передающая
1
808
224
Очередь приемная
2
1

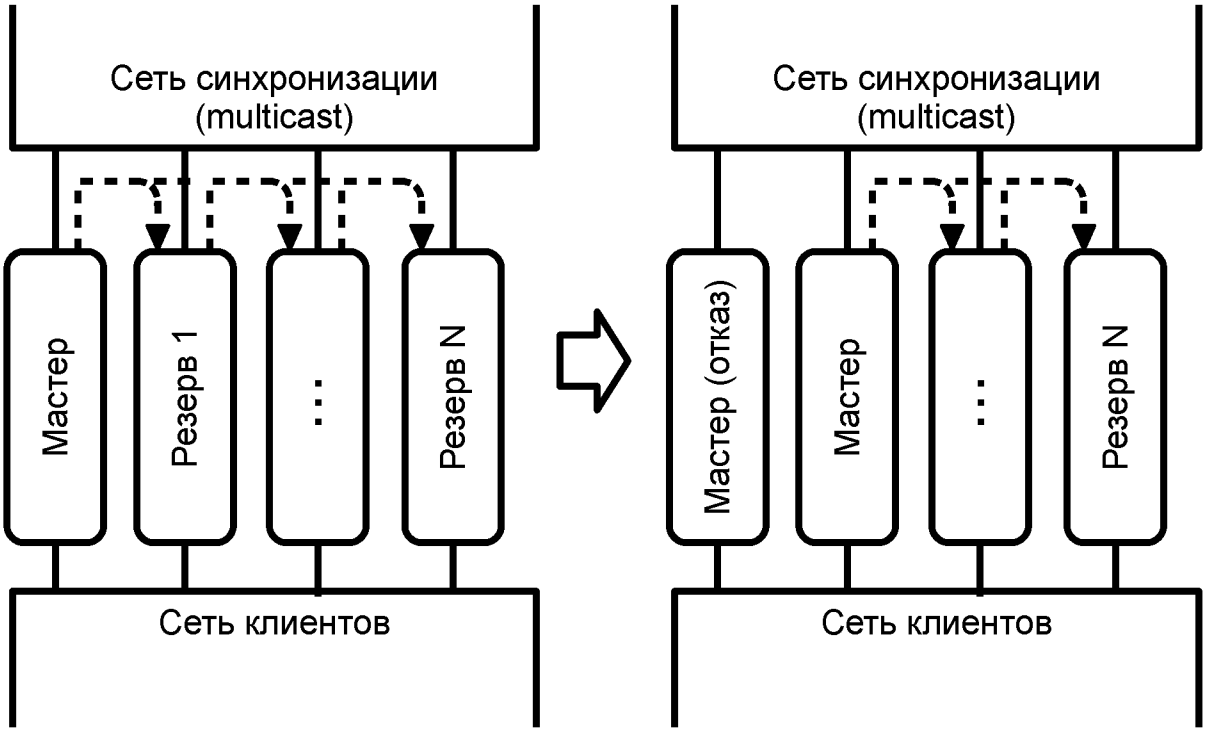
ФИГ. 8


909
121
Очередь сообщений
v6
v5
v4
v3
v2
v1
Корневой узел 1
A
B
Корневой узел 2
A
B
909
121
Очередь сообщений
v8
v7
v6
v5
v4
v3
v1
v2
Корневой узел 1
A
v1
B
Корневой узел 2
A
B
v2
909
121
Очередь сообщений
v8
v7
v6
v5
v4
v3
Корневой узел 1
A
v1
B
v2
Корневой узел 2
A
v1
B
v2

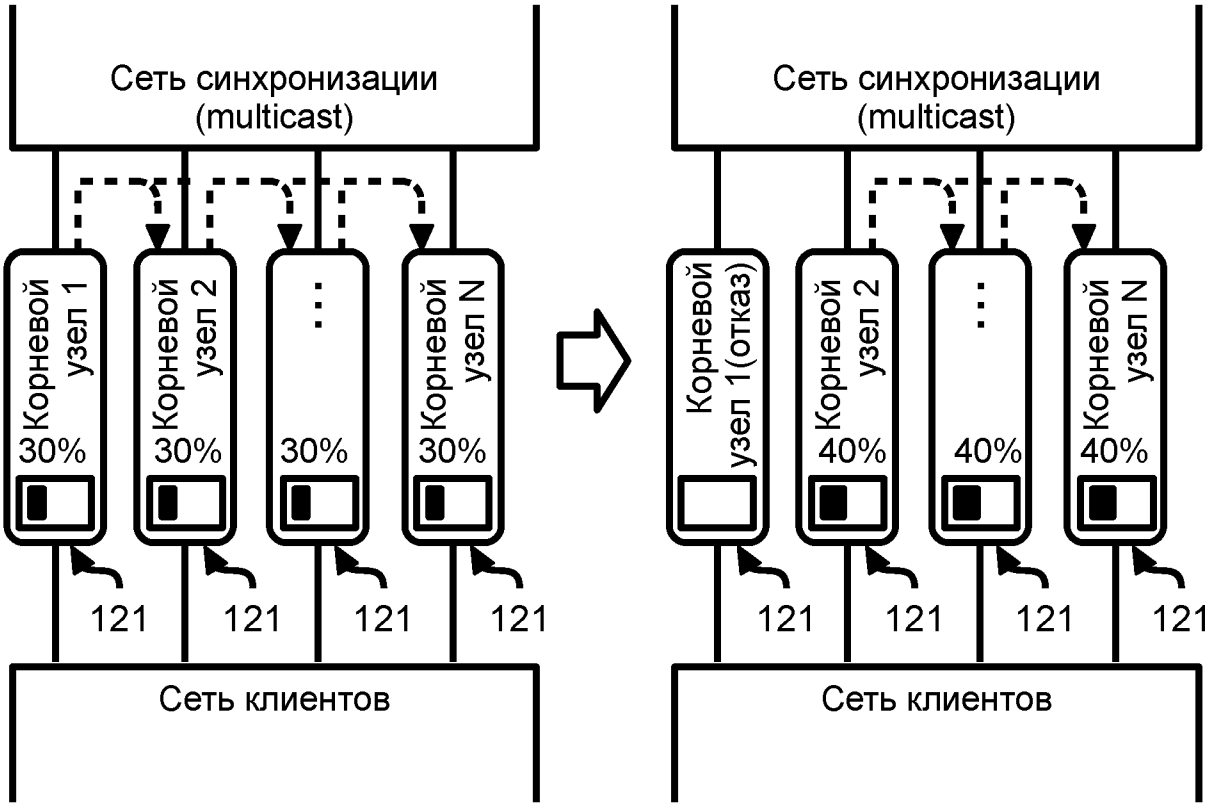
ФИГ. 9


121
1010
Корневой узел 1
A
v1
B
C
v3
D
Балансировщик
Приоритет
1
2
3
4
v3
v2
v1
v4
t
A
B
v2
C
D
v4
Корневой узел 2
1020

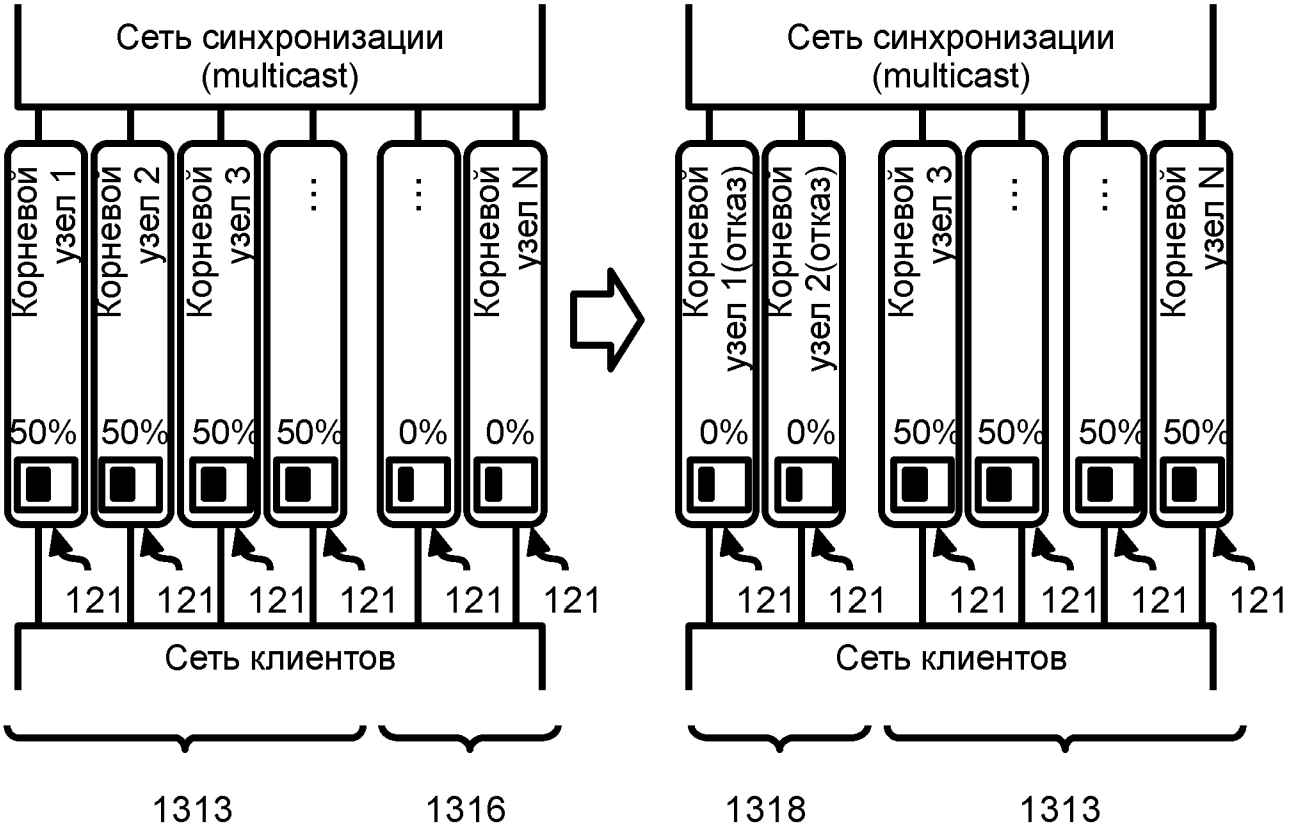
ФИГ. 10


Сеть синхронизации
(multicast)
Мастер
Резерв 1
...
Резерв N
Сеть клиентов
Сеть синхронизации
(multicast)
Мастер (отказ)
Мастер
...
Резерв N
Сеть клиентов

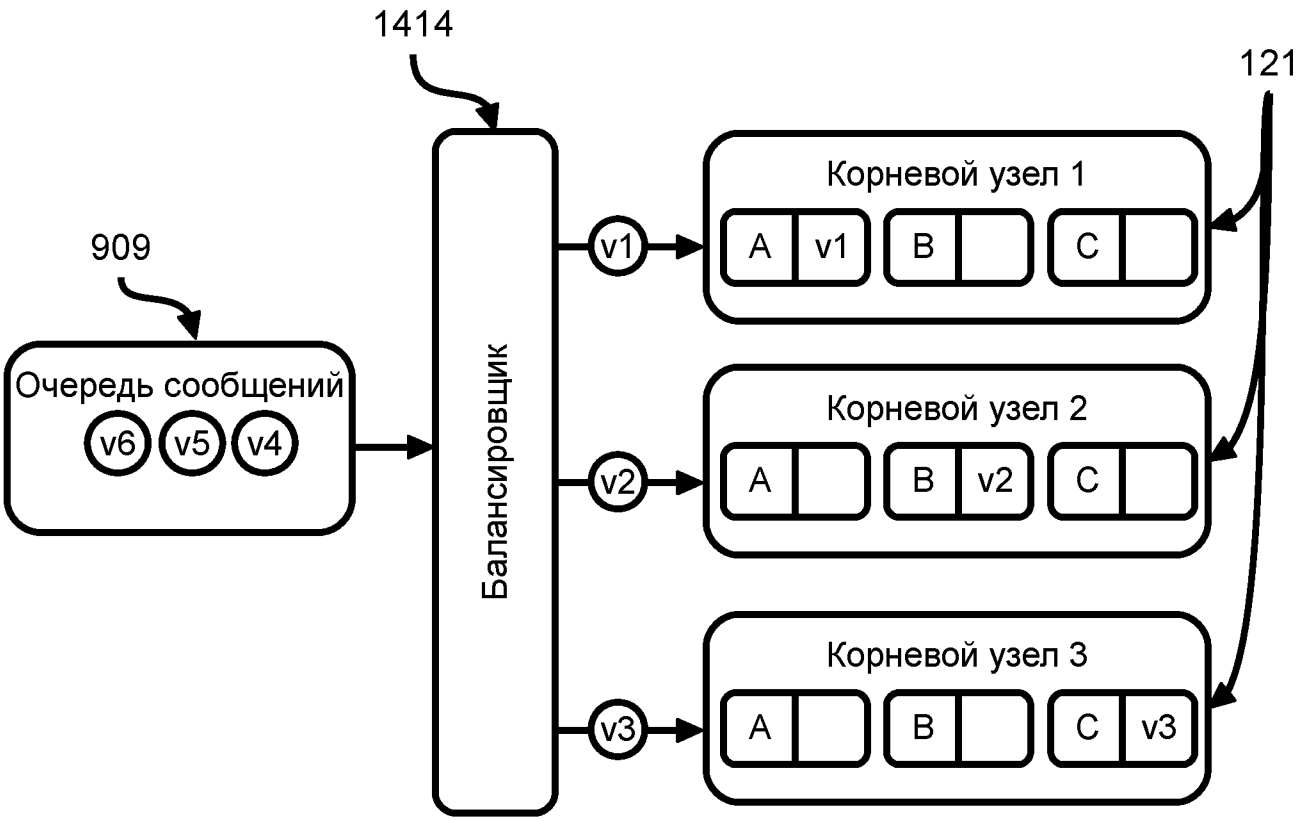
ФИГ. 11


Сеть синхронизации
(multicast)
Корневой узел 1
30%
Корневой узел 2
30%
...
Корневой узел N
30%
121
121
121
121
Сеть клиентов
Сеть синхронизации
(multicast)
Корневой узел 1(отказ)
Корневой узел 2
40%
...
Корневой узел N
40%
121
121
121
121
Сеть клиентов

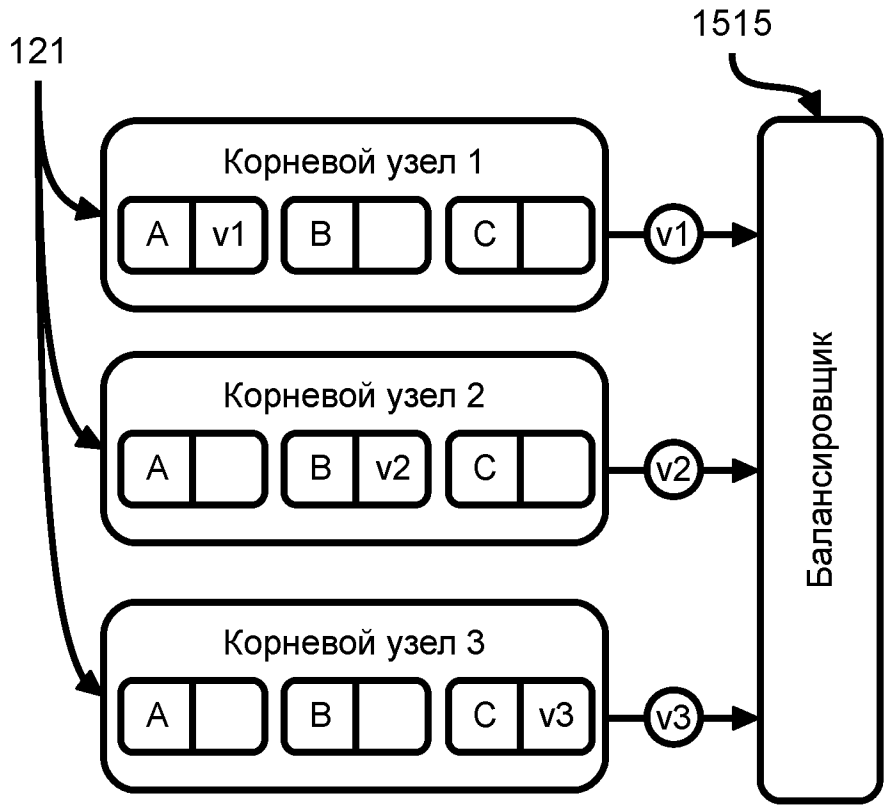
ФИГ. 12


Сеть синхронизации
(multicast)
Корневой узел 1
Корневой узел 2
Корневой узел 3
...
...
Корневой узел N
50%
50%
50%
50%
0%
0%
121
121
121
121
121
121
Сеть клиентов
1313
1316
Сеть синхронизации
(multicast)
Корневой узел 1(отказ)
Корневой узел 2(отказ)
Корневой узел 3
...
...
Корневой узел N
0%
0%
50%
50%
50%
50%
121
121
121
121
121
121
Сеть клиентов
1318
1313

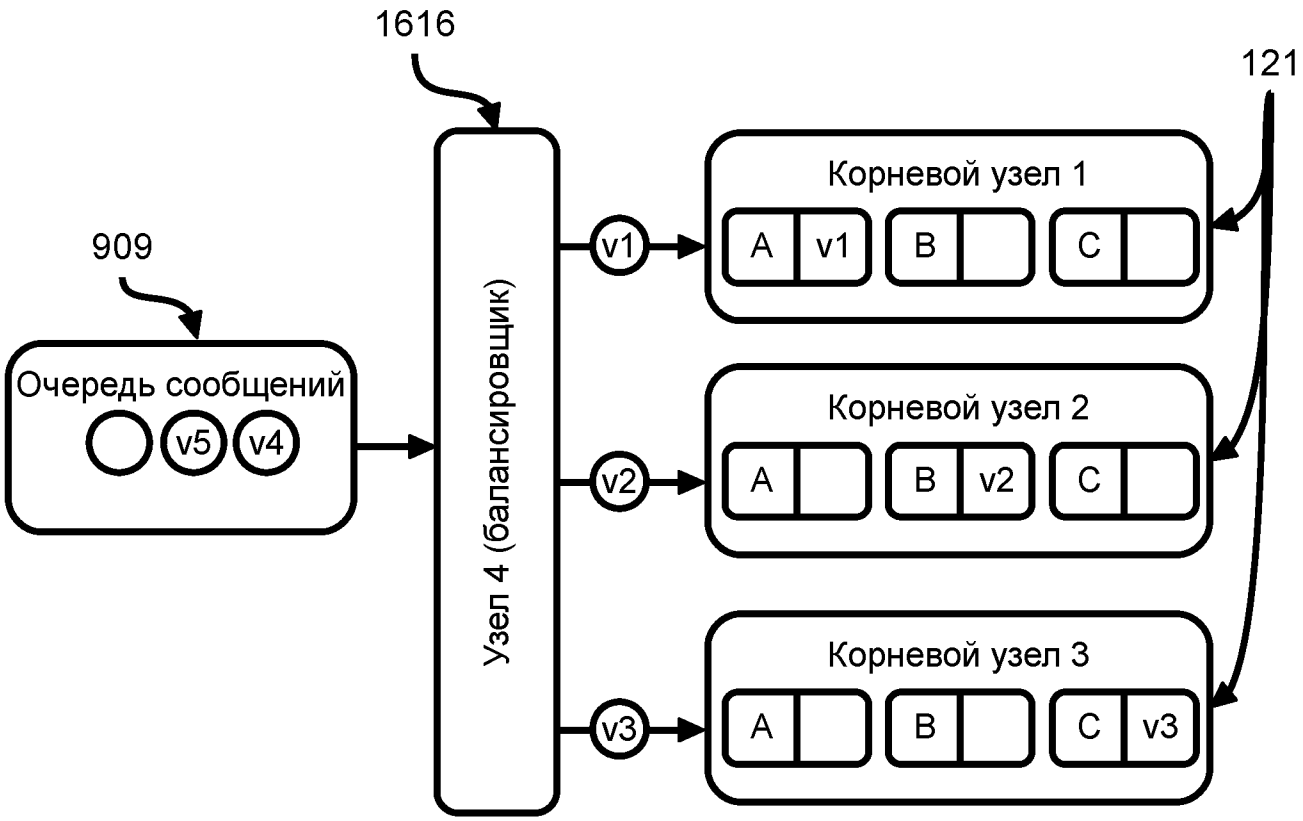
ФИГ. 13


1414
121
909
Очередь сообщений
v6
v5
v4
Балансировщик
v1
v2
v3
Корневой узел 1
A
v1
B
C
Корневой узел 2
A
B
v2
C
Корневой узел 3
A
B
C
v3

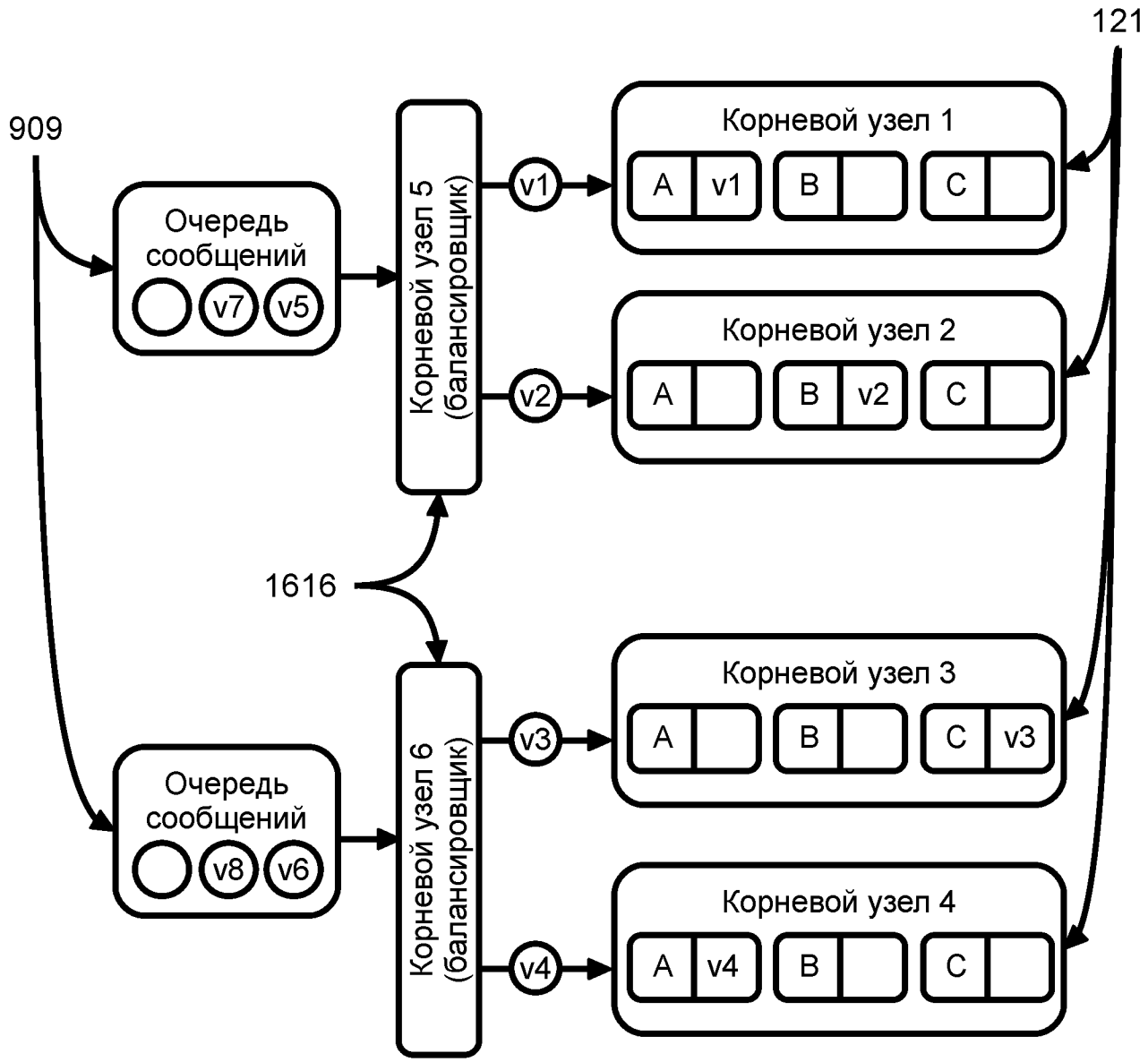
ФИГ. 14


121
1515
Корневой узел 1
A
v1
B
C
v1
Корневой узел 2
A
B
v2
C
v2
Корневой узел 3
A
B
C
v3
v3
Балансировщик

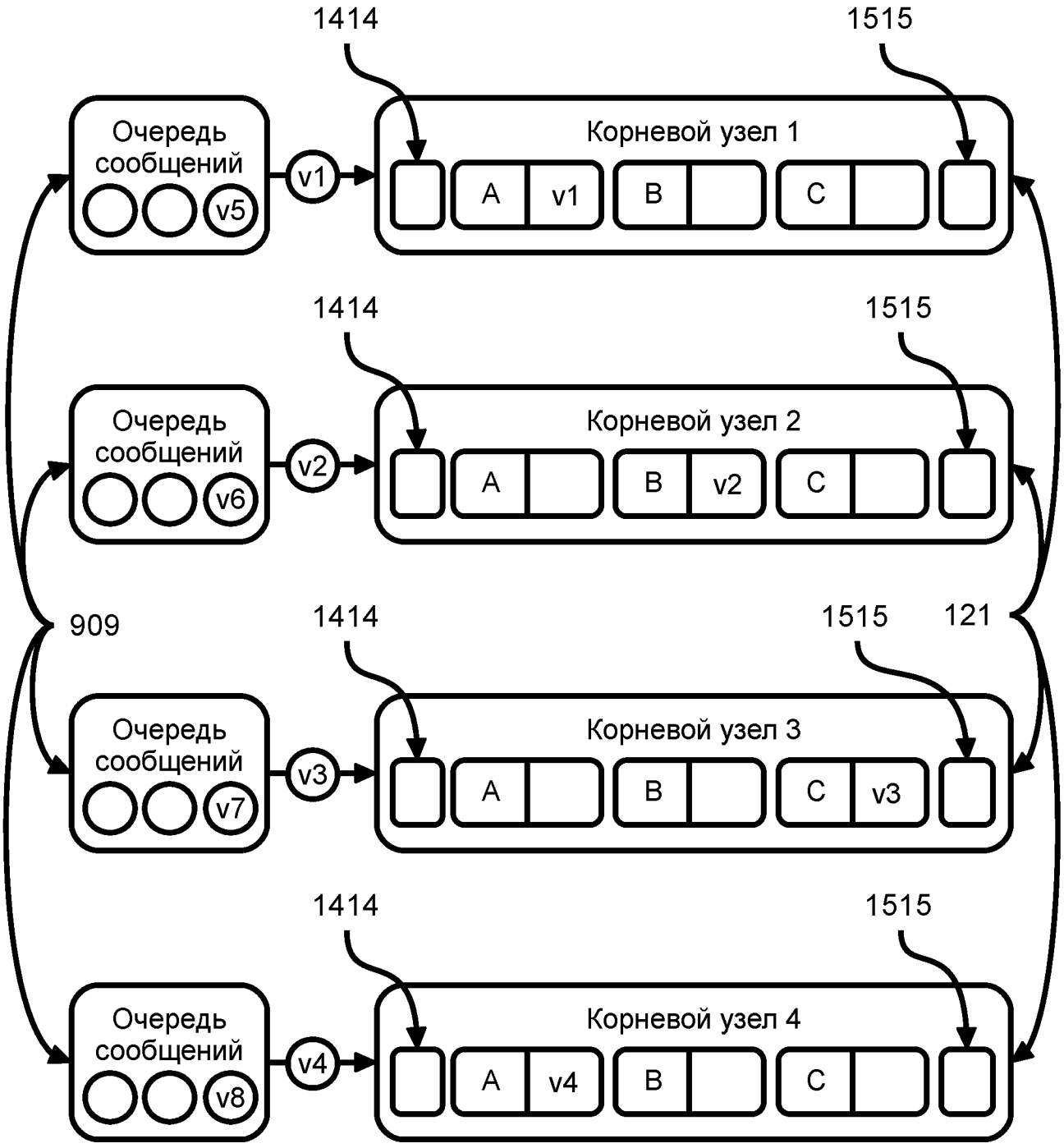
ФИГ. 15


1616
121
909
Очередь сообщений
v5
v4
Узел 4 (балансировщик)
v1
v2
v3
Корневой узел 1
A
v1
B
C
Корневой узел 2
A
B
v2
C
Корневой узел 3
A
B
C
v3

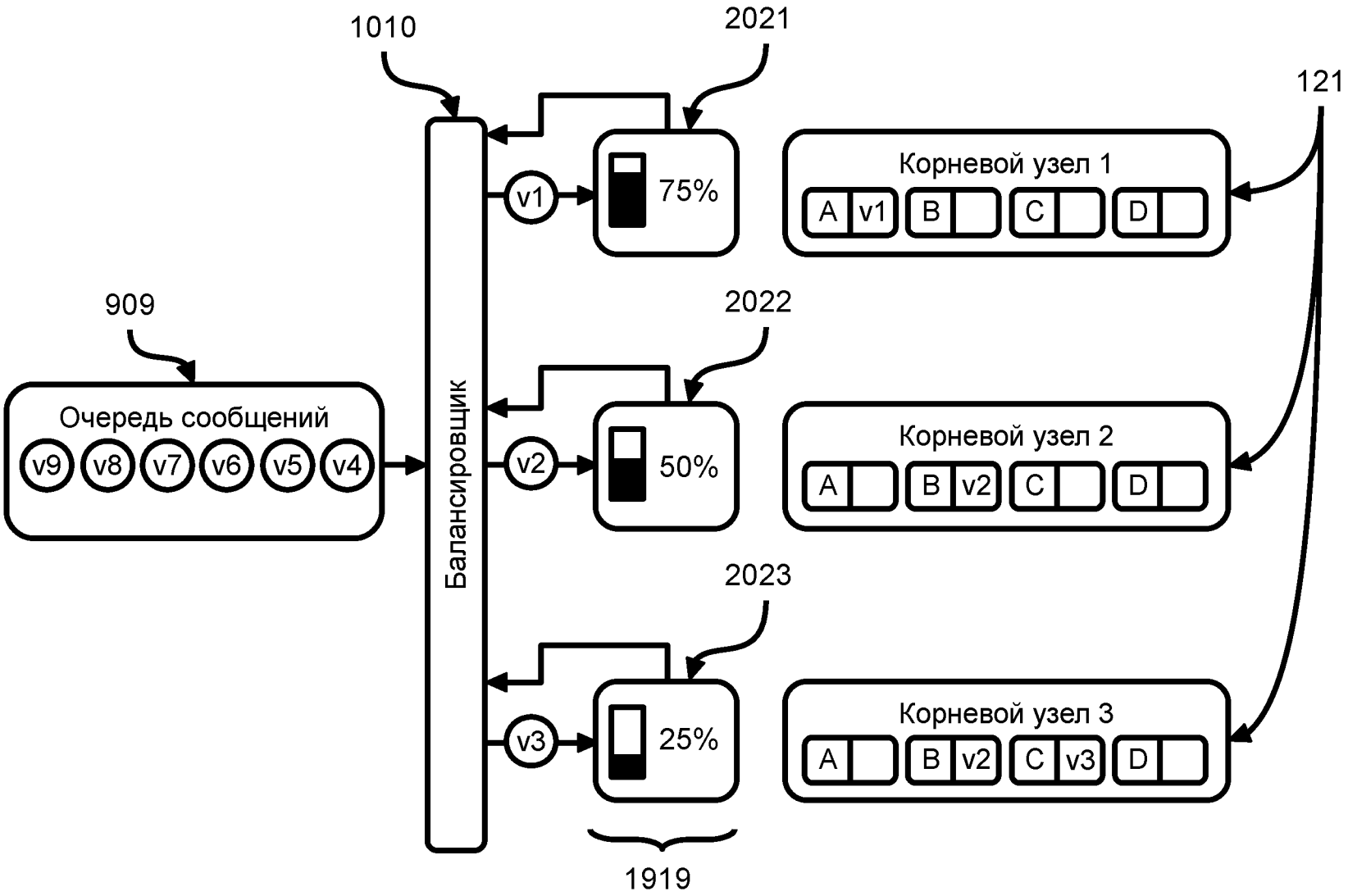
ФИГ. 16


121
909
Очередь сообщений
v7
v5
Корневой узел 5 (балансировщик)
v1
v2
Корневой узел 1
A
v1
B
C
Корневой узел 2
A
B
v2
C
1616
Очередь сообщений
v8
v6
Корневой узел 6 (балансировщик)
v3
v4
Корневой узел 3
A
B
C
v3
Корневой узел 4
A
v4
B
C

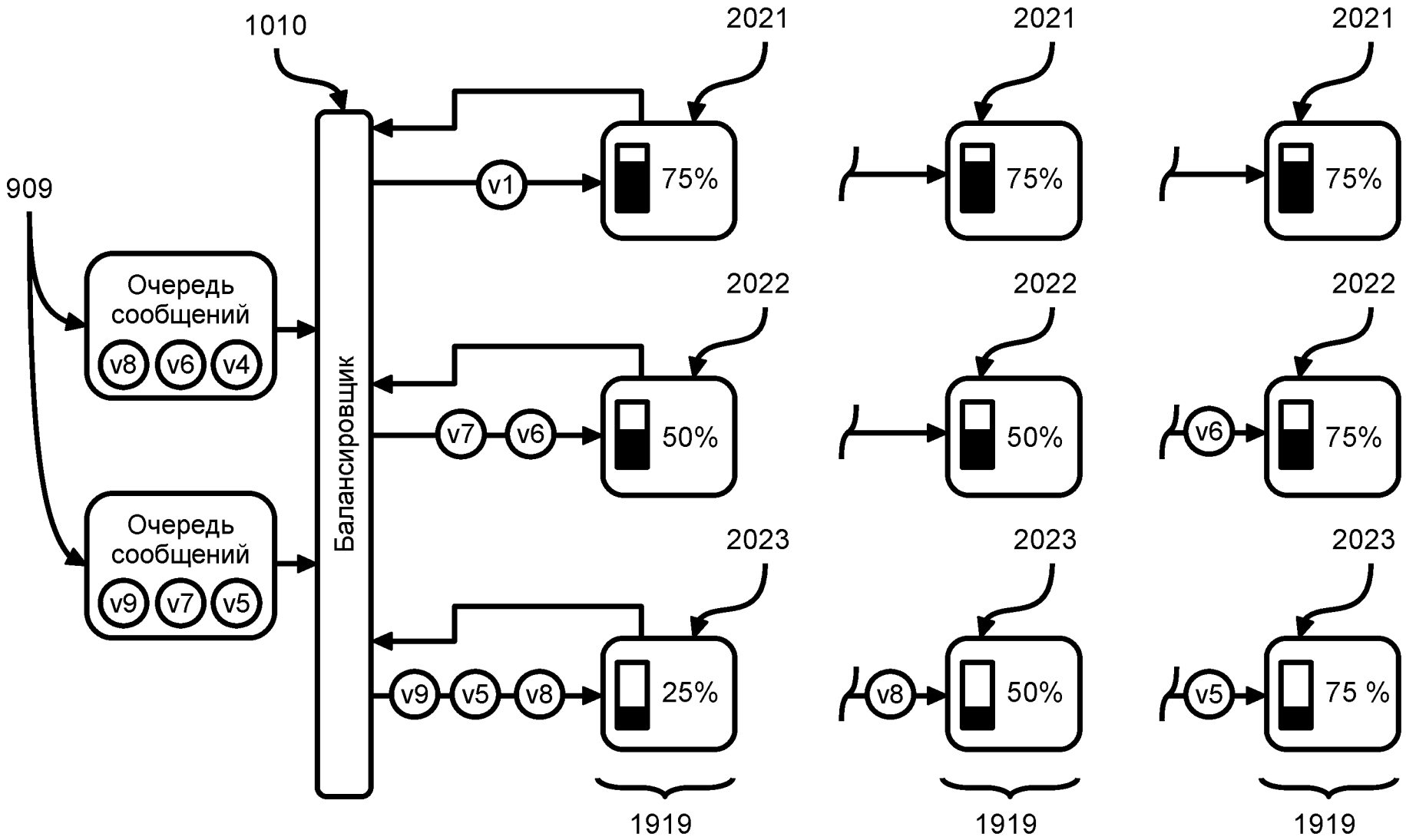
ФИГ. 17


1414
1515
Очередь сообщений
v5
v1
Корневой узел 1
A
v1
B
C
1414
1515
Очередь сообщений
v6
v2
Корневой узел 2
A
B
v2
C
909
1414
1515
121
Очередь сообщений
v7
v3
Корневой узел 3
A
B
C
v3
1414
1515
Очередь сообщений
v8
v4
Корневой узел 4
A
v4
B
C

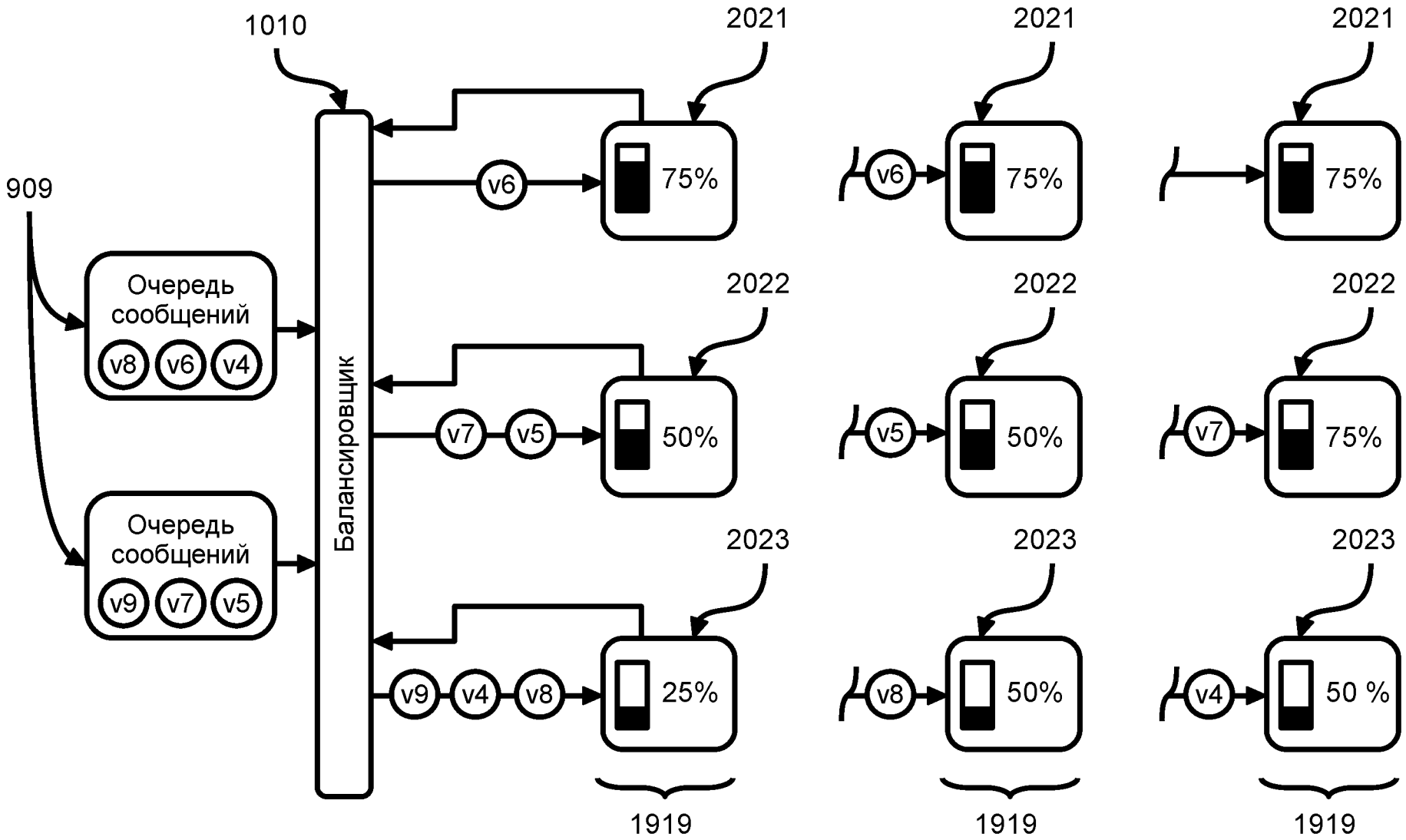
ФИГ. 18


1010
2021
121
909
2022
2023
Очередь сообщений
v9
v8
v7
v6
v5
v4
Балансировщик
v1
75%
v2
50%
v3
25%
1919
Корневой узел 1
A
v1
B
C
D
Корневой узел 2
A
B
v2
C
D
Корневой узел 3
A
B
v2
C
v3
D

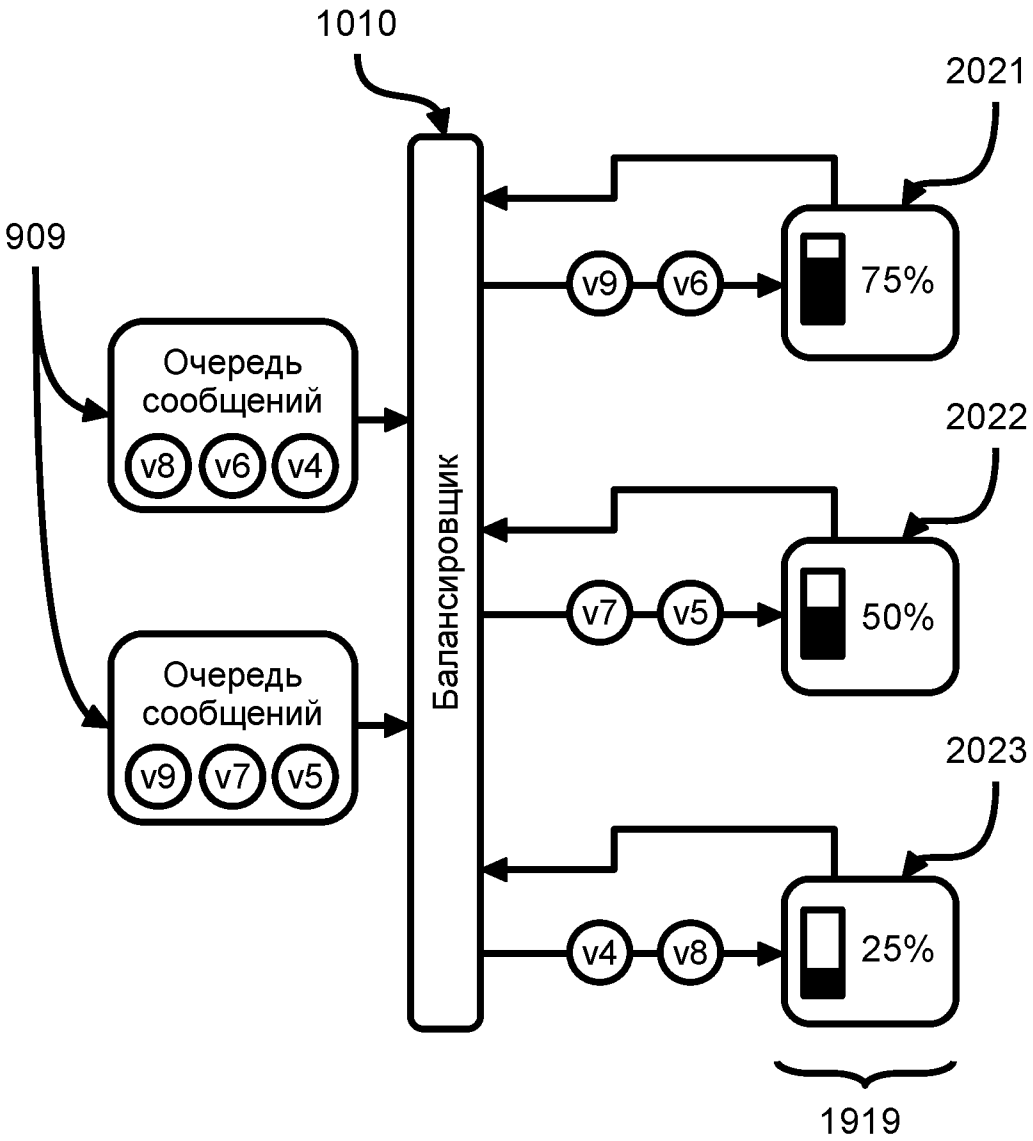
ФИГ. 19


1010
2021
2021
2021
909
Очередь сообщений
v8
v6
v4
Очередь сообщений
v9
v7
v5
Балансировщик
v1
75%
75%
75%
2022
2022
2022
v7
v6
50%
50%
v6
75%
2023
2023
2023
v9
v5
v8
25%
v8
50%
v5
75 %
1919
1919
1919

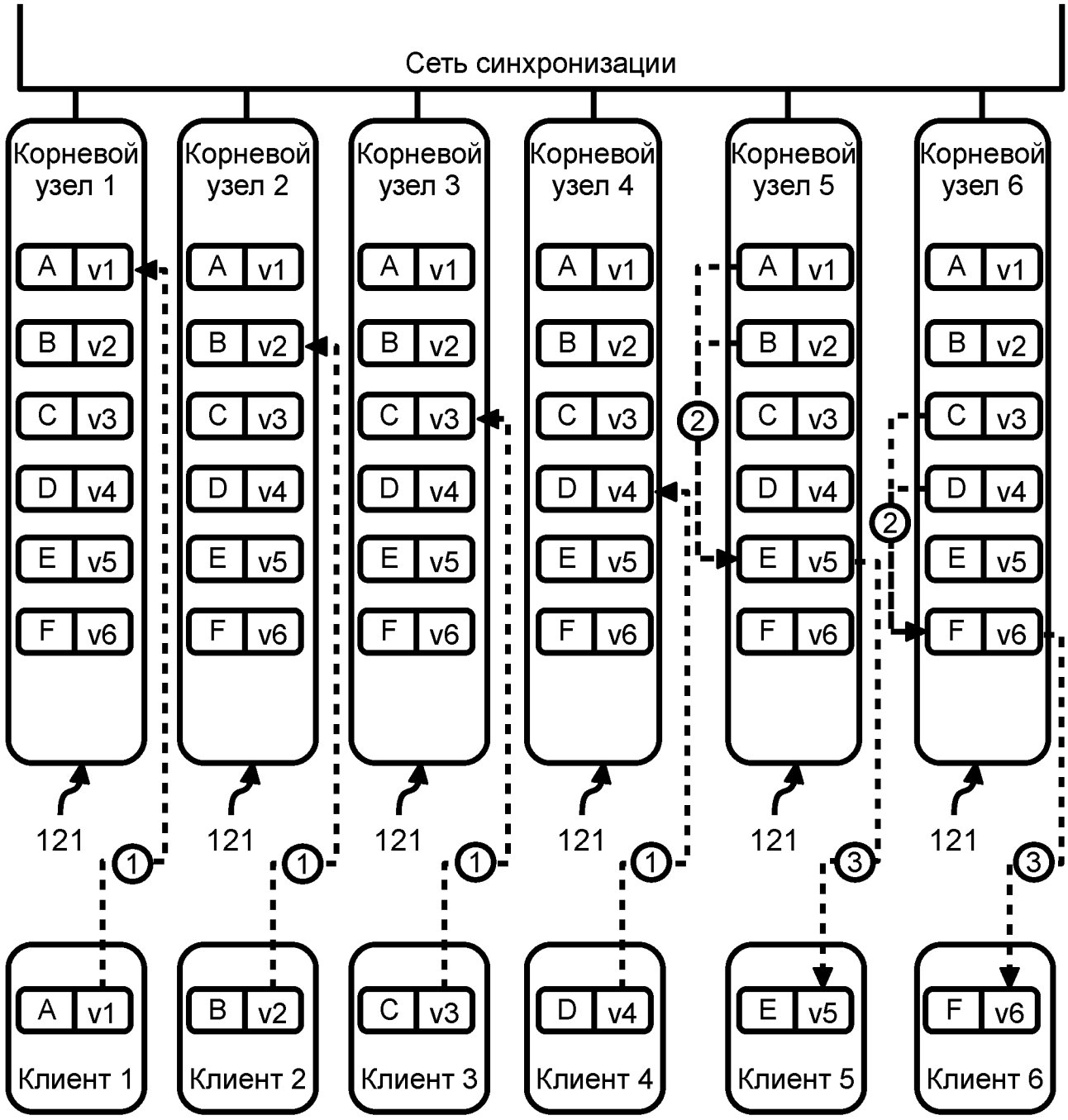
ФИГ. 20


1010
2021
2021
2021
909
Очередь сообщений
v8
v6
v4
v6
75%
v6
75%
75%
2022
2022
2022
v7
v5
50%
v5
50%
v7
75%
Балансировщик
Очередь сообщений
v9
v7
v5
2023
2023
2023
v9
v4
v8
25%
v8
50%
v4
50 %
1919
1919
1919

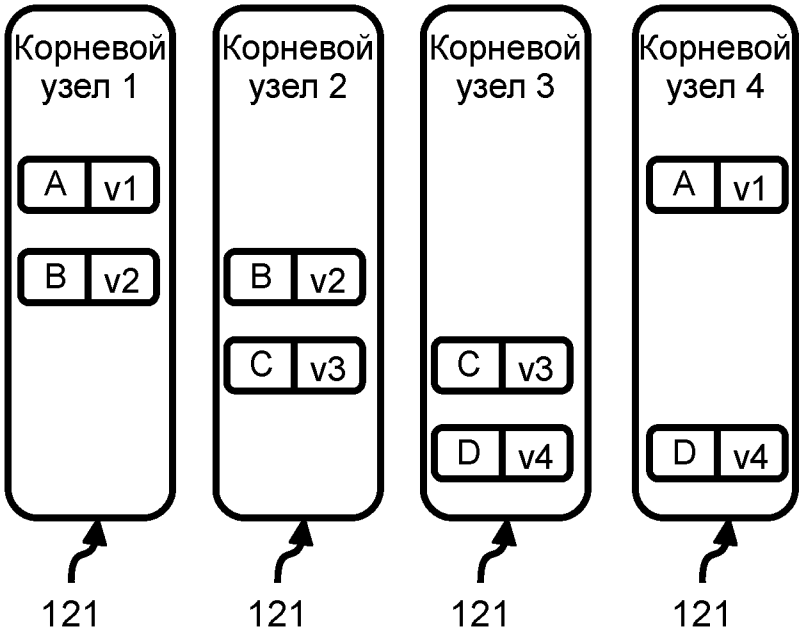
ФИГ. 21


1010
2021
909
v9
v6
75%
Очередь сообщений
v8
v6
v4
2022
Балансировщик
v7
v5
50%
Очередь сообщений
v9
v7
v5
2023
v4
v8
25%
1919

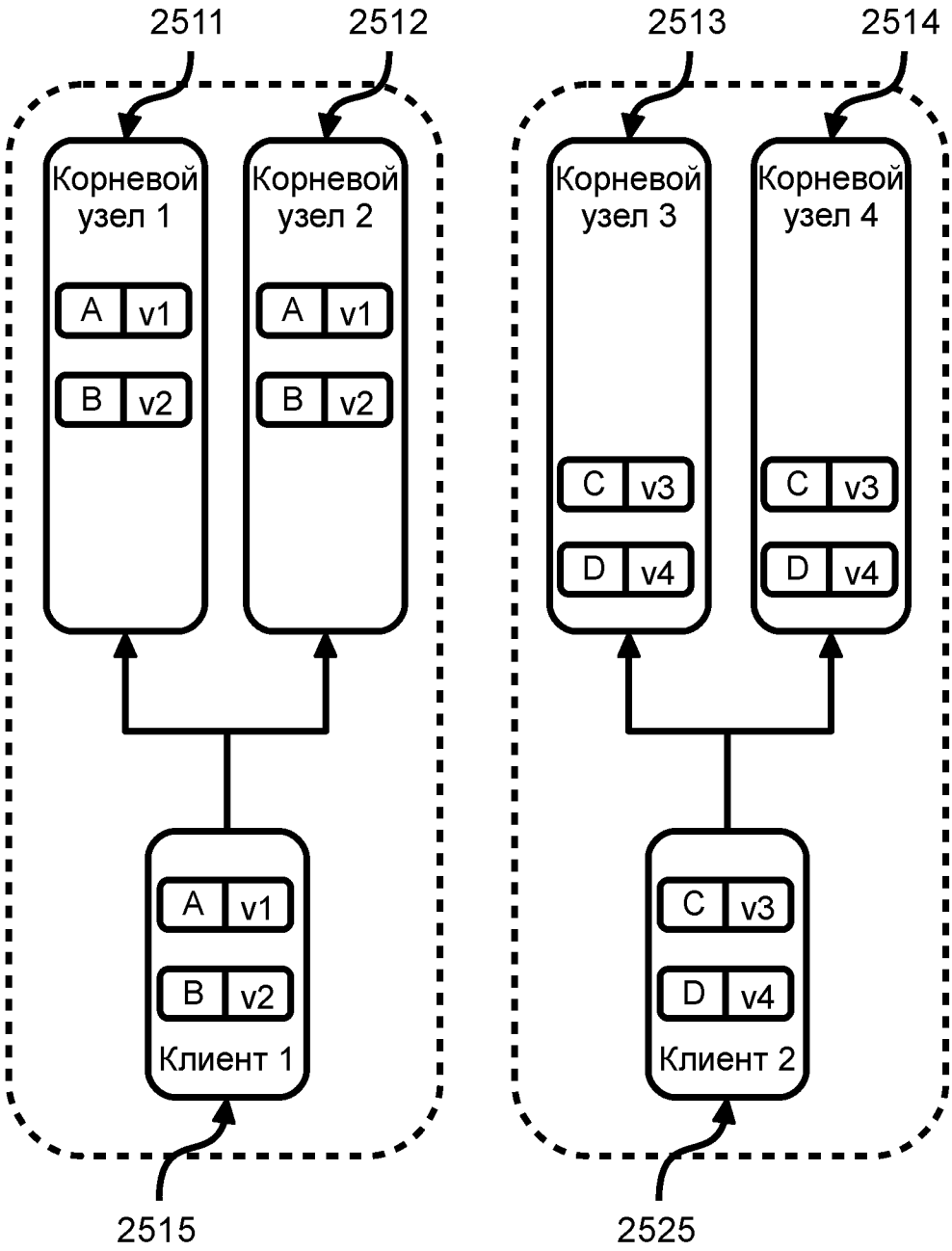
ФИГ. 22


Сеть синхронизации
Корневой узел 1
Корневой узел 2
Корневой узел 3
Корневой узел 4
Корневой узел 5
Корневой узел 6
A v1
B v2
C v3
D v4
E v5
F v6
121
1
2
3
Клиент 1
Клиент 2
Клиент 3
Клиент 4
Клиент 5
Клиент 6

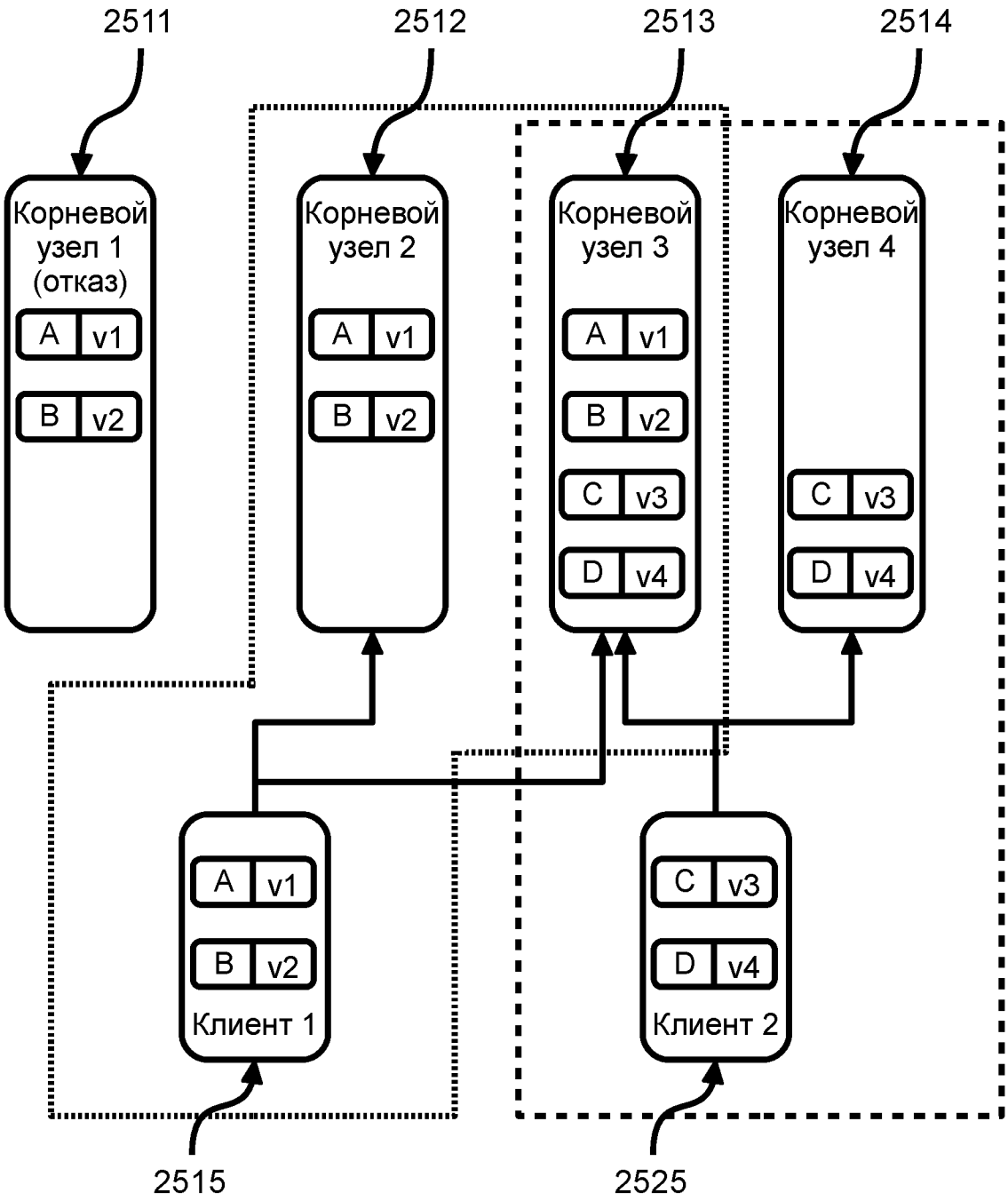
ФИГ. 23


Корневой узел 1
A
v1
B
v2
121
Корневой узел 2
B
v2
C
v3
121
Корневой узел 3
C
v3
D
v4
121
Корневой узел 4
A
v1
D
v4
121

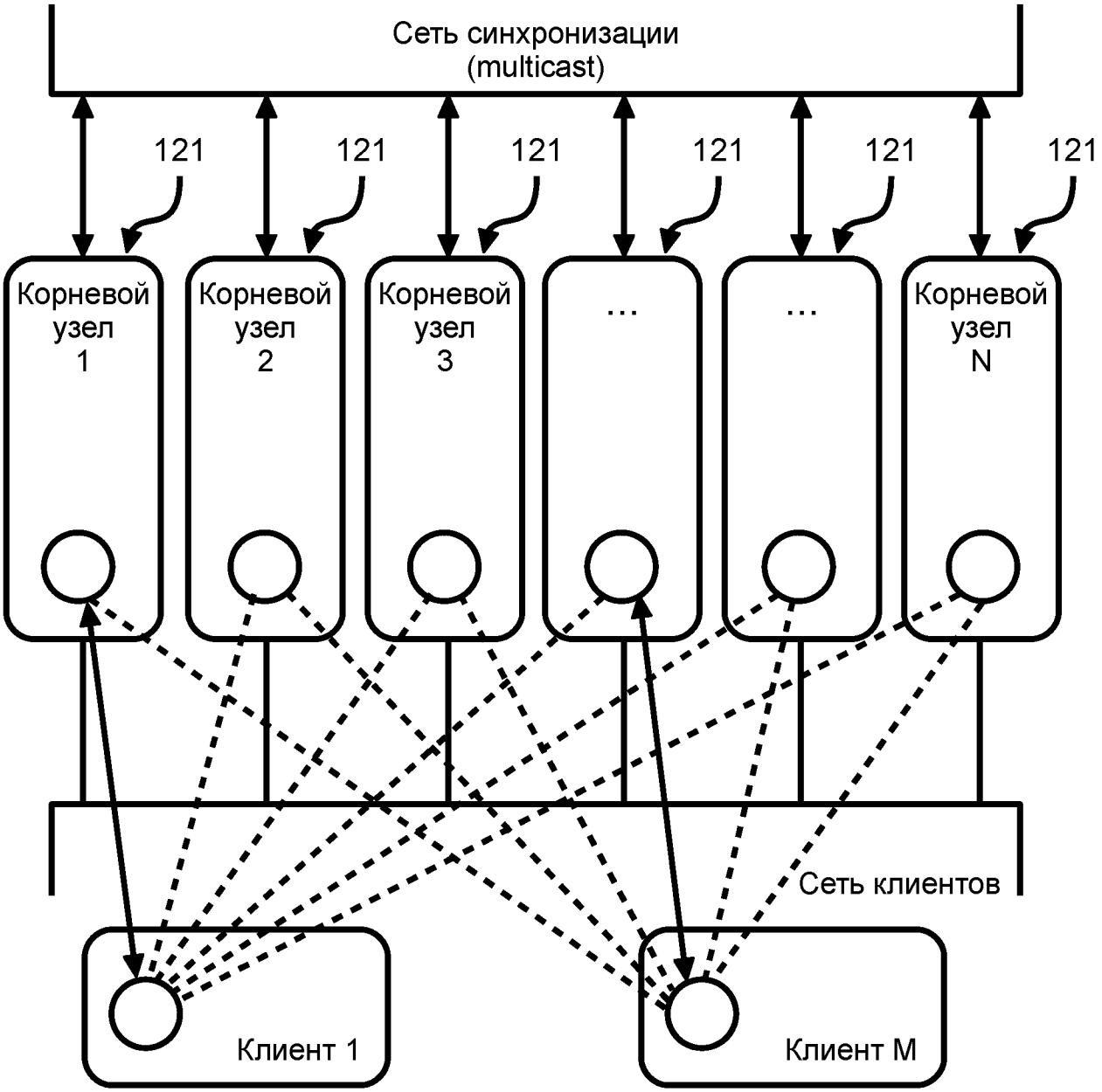
ФИГ. 24


2511
2512
2513
2514
Корневой узел 1
Корневой узел 2
Корневой узел 3
Корневой узел 4
A
v1
B
v2
C
v3
D
v4
Клиент 1
Клиент 2
2515
2525

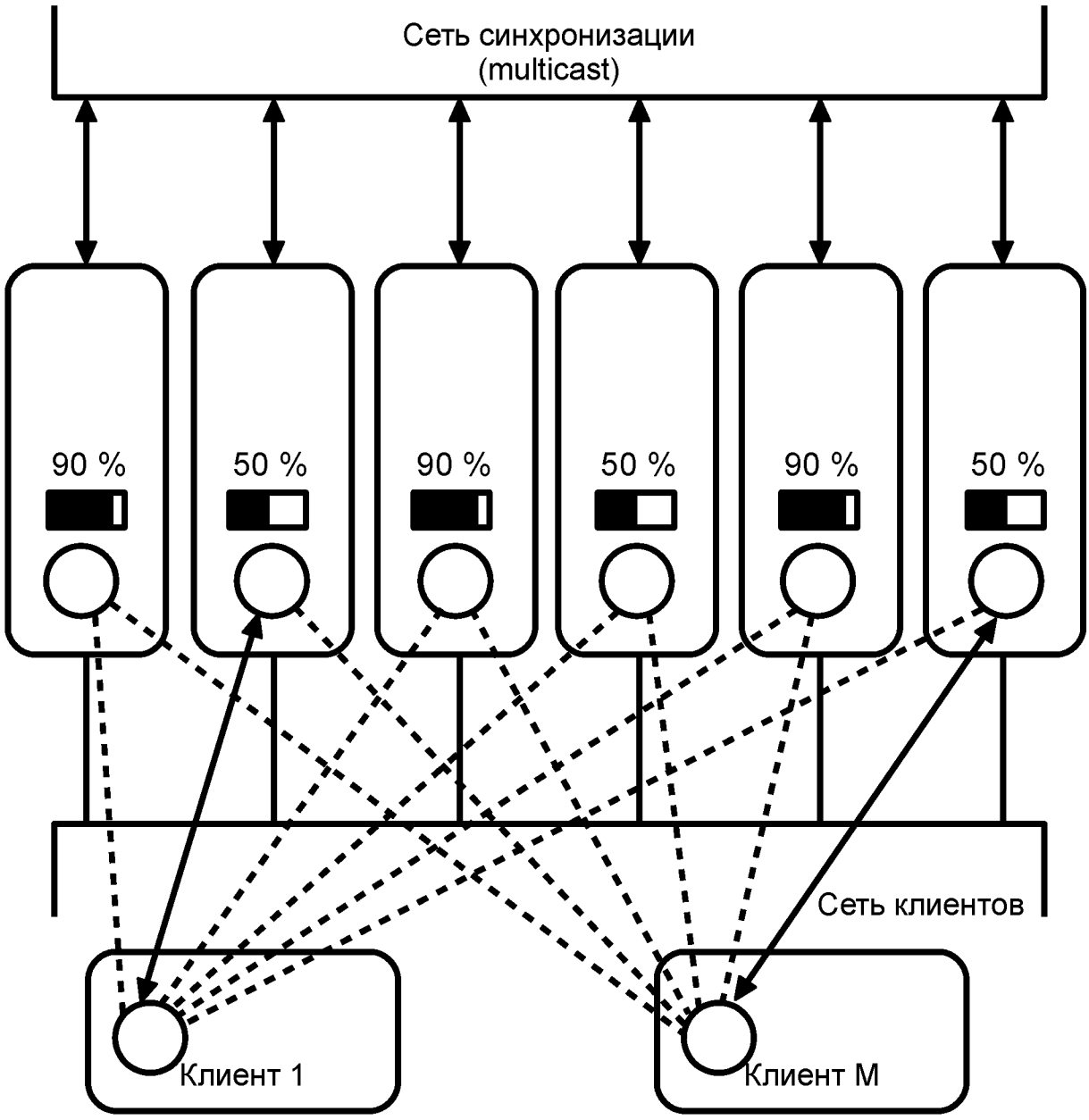
ФИГ. 25


2511
2512
2513
2514
Корневой узел 1 (отказ)
A v1
B v2
Корневой узел 2
A v1
B v2
Корневой узел 3
A v1
B v2
C v3
D v4
Корневой узел 4
C v3
D v4
A v1
B v2
Клиент 1
C v3
D v4
Клиент 2
2515
2525

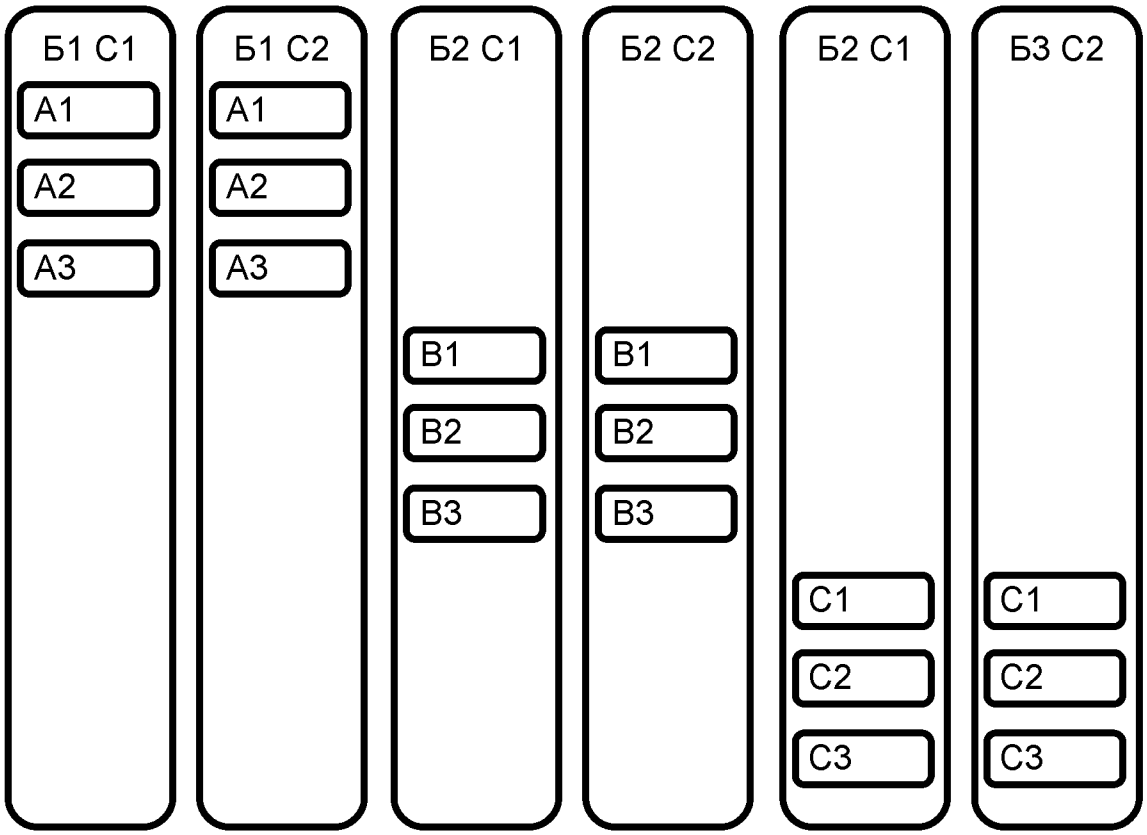
ФИГ. 26


Сеть синхронизации
(multicast)
121
121
121
121
121
121
Корневой
узел
1
Корневой
узел
2
Корневой
узел
3
...
...
Корневой
узел
N
Сеть клиентов
Клиент 1
Клиент M

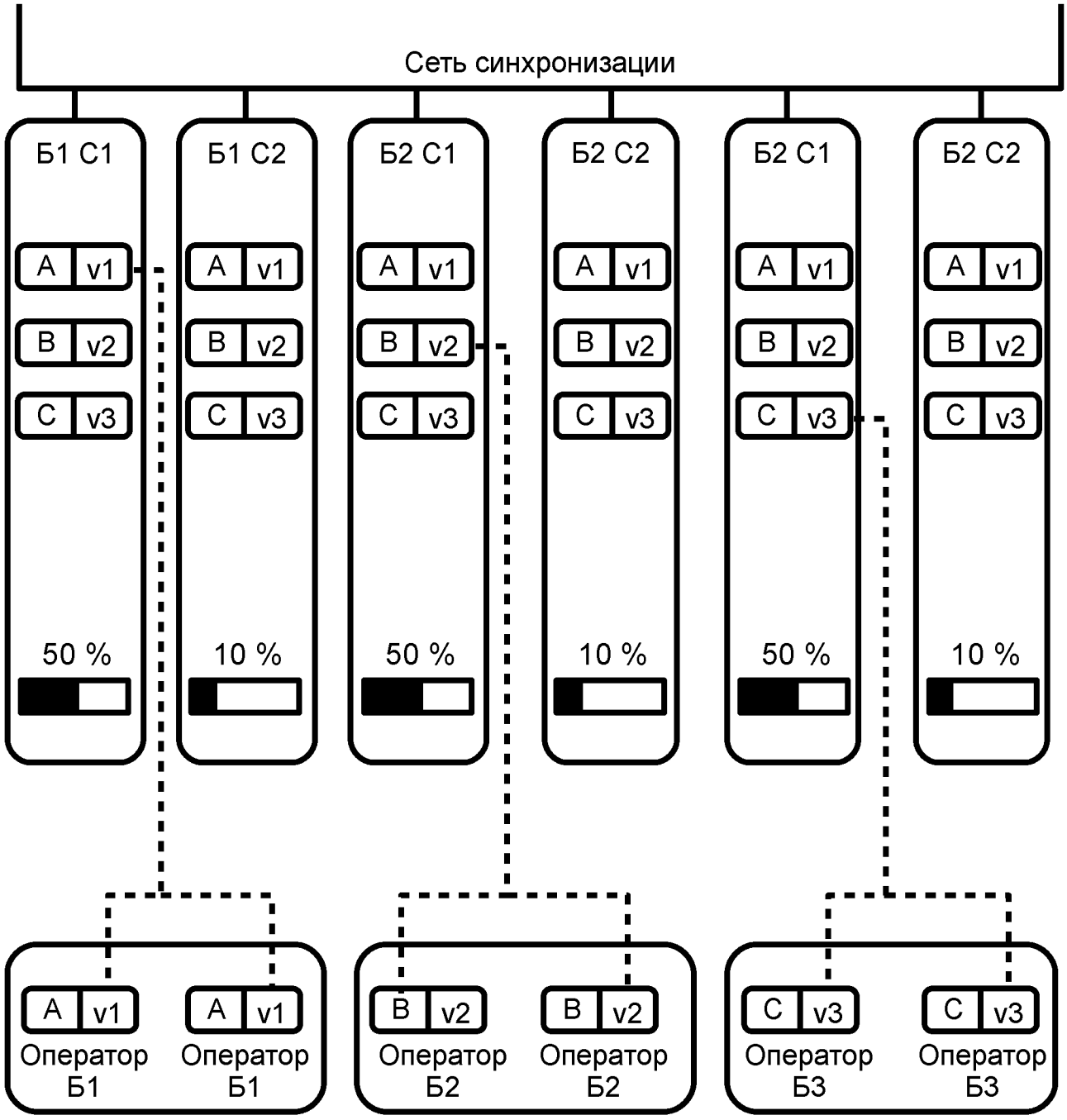
ФИГ. 27


Сеть синхронизации
(multicast)
90 %
50 %
90 %
50 %
90 %
50 %
Сеть клиентов
Клиент 1
Клиент M

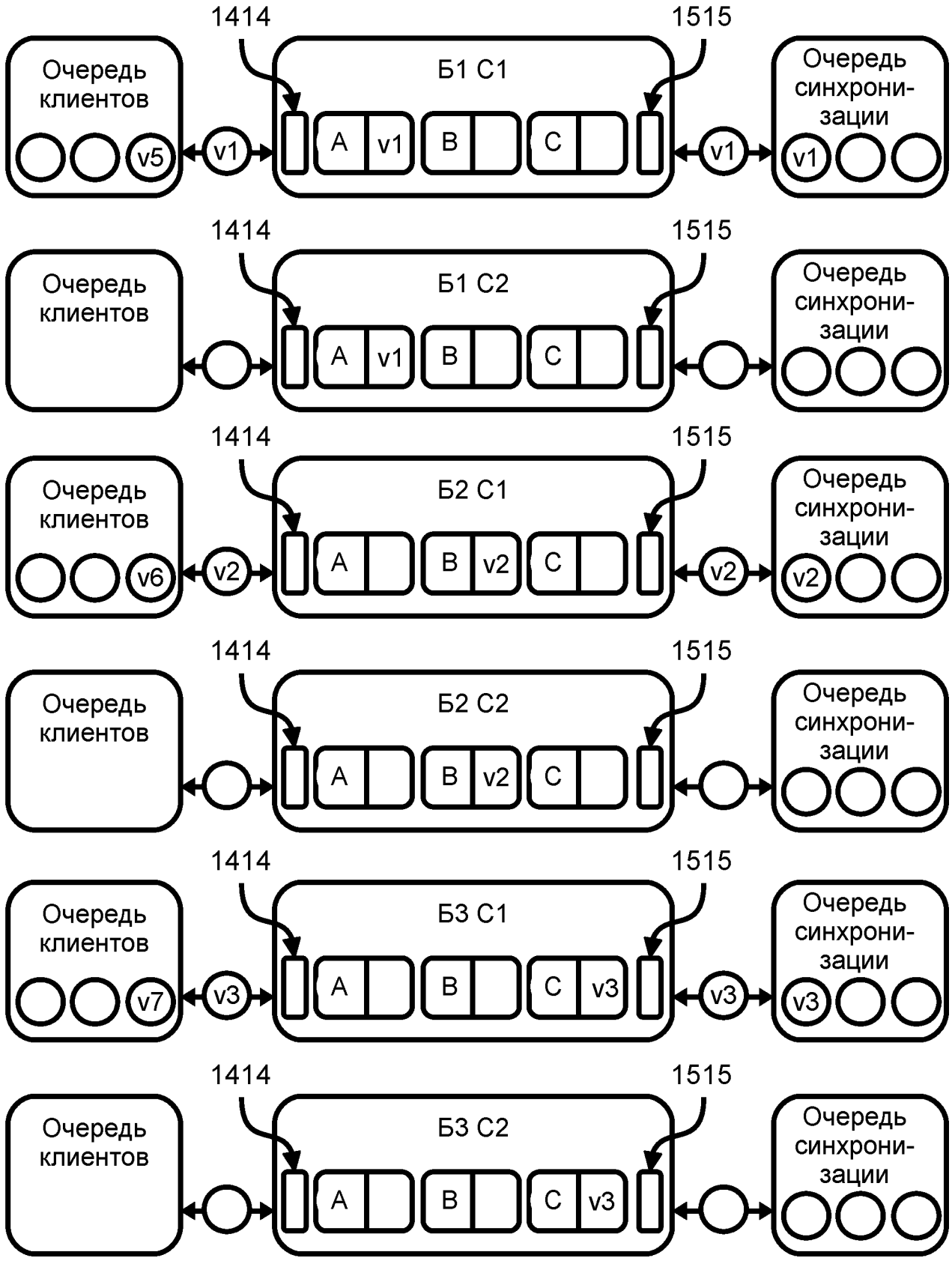
ФИГ. 28


Б1 С1
A1
A2
A3
Б1 С2
A1
A2
A3
Б2 С1
B1
B2
B3
Б2 С2
B1
B2
B3
Б2 С1
C1
C2
C3
Б3 С2
C1
C2
C3

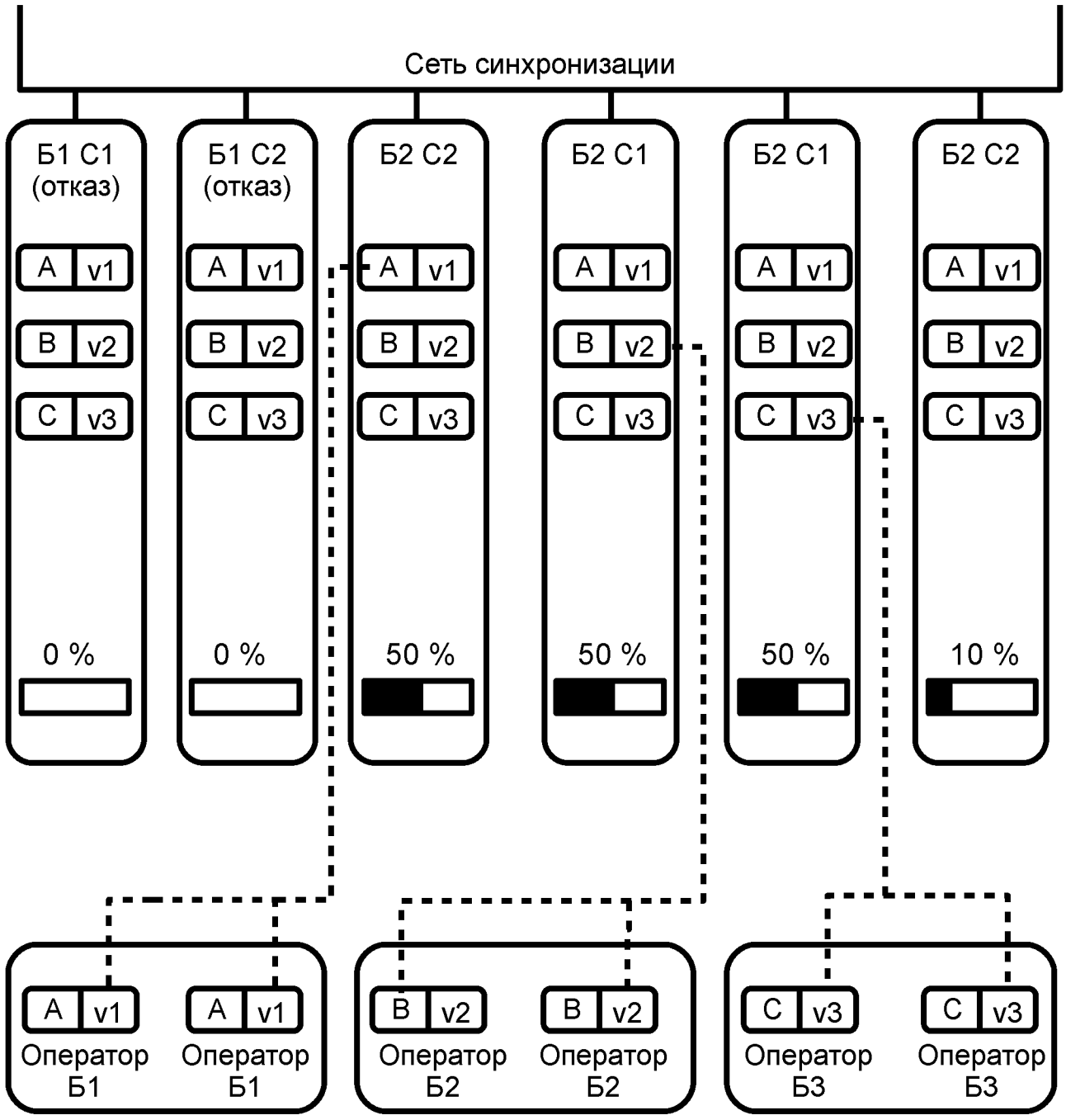
ФИГ. 29


Сеть синхронизации
Б1 С1
Б1 С2
Б2 С1
Б2 С2
Б2 С1
Б2 С2
A v1
B v2
C v3
50 %
10 %
50 %
10 %
50 %
10 %
Оператор Б1
Оператор Б1
Оператор Б2
Оператор Б2
Оператор Б3
Оператор Б3

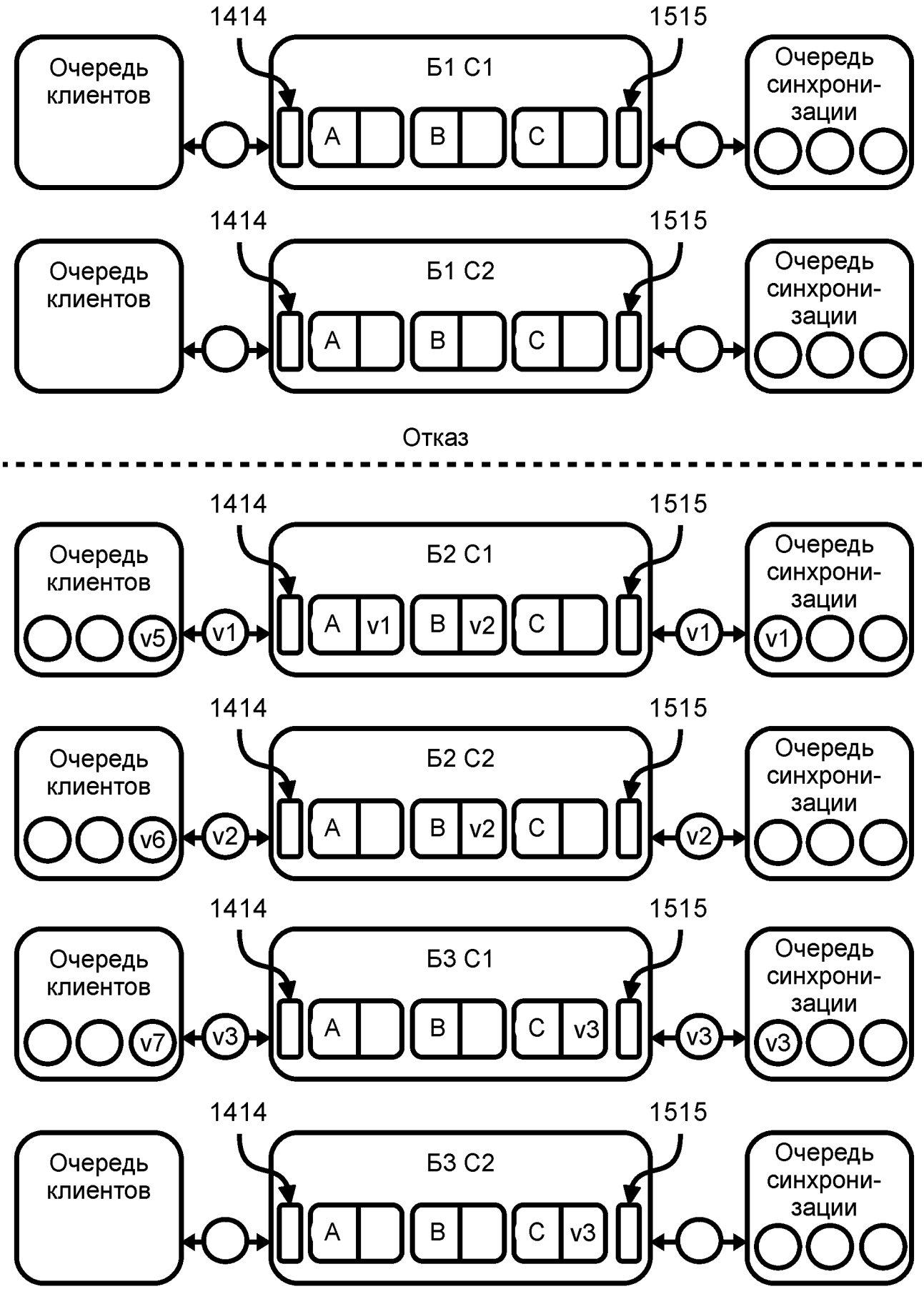
ФИГ. 30


1414
1515
Очередь клиентов
v5
v1
Б1 С1
A
v1
B
C
v1
Очередь синхрони-зации
v1
Очередь клиентов
Б1 С2
A
v1
B
C
Очередь синхрони-зации
Очередь клиентов
v6
v2
Б2 С1
A
B
v2
C
v2
Очередь синхрони-зации
v2
Очередь клиентов
Б2 С2
A
B
v2
C
Очередь синхрони-зации
Очередь клиентов
v7
v3
Б3 С1
A
B
C
v3
v3
Очередь синхрони-зации
v3
Очередь клиентов
Б3 С2
A
B
C
v3
Очередь синхрони-зации

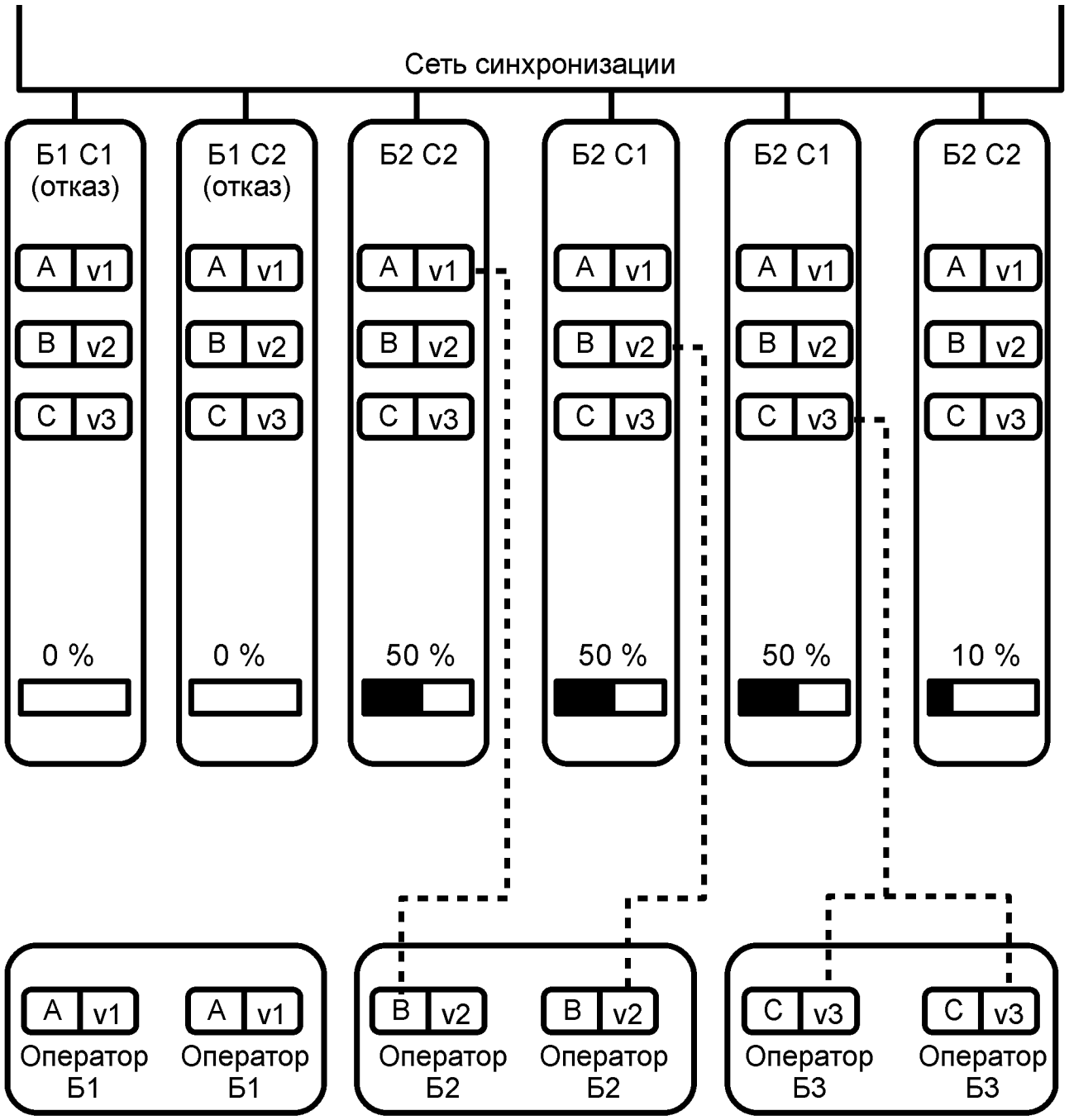
ФИГ. 31


Сеть синхронизации
Б1 С1 (отказ)
Б1 С2 (отказ)
Б2 С2
Б2 С1
Б2 С1
Б2 С2
A v1
B v2
C v3
0 %
0 %
50 %
50 %
50 %
10 %
Оператор Б1
Оператор Б1
Оператор Б2
Оператор Б2
Оператор Б3
Оператор Б3

ФИГ. 32


1414
1515
Очередь клиентов
Б1 С1
A
B
C
Очередь синхрони-зации
1414
1515
Очередь клиентов
Б1 С2
A
B
C
Очередь синхрони-зации
Отказ
1414
1515
Очередь клиентов
v5
v1
Б2 С1
A
v1
B
v2
C
v1
Очередь синхрони-зации
v1
1414
1515
Очередь клиентов
v6
v2
Б2 С2
A
B
v2
C
v2
Очередь синхрони-зации
1414
1515
Очередь клиентов
v7
v3
Б3 С1
A
B
C
v3
v3
Очередь синхрони-зации
v3
1414
1515
Очередь клиентов
Б3 С2
A
B
C
v3
Очередь синхрони-зации

ФИГ. 33


Сеть синхронизации
Б1 С1 (отказ)
Б1 С2 (отказ)
Б2 С2
Б2 С1
Б2 С1
Б2 С2
A v1
B v2
C v3
0 %
0 %
50 %
50 %
50 %
10 %
Оператор Б1
Оператор Б1
Оператор Б2
Оператор Б2
Оператор Б3
Оператор Б3

ФИГ. 34

ОТЧЕТ О ПАТЕНТНОМ ПОИСКЕ

(статья 15(3) ЕАПК и правило 42 Патентной инструкции к ЕАПК)

Номер евразийской заявки:

202390955

А. КЛАССИФИКАЦИЯ ПРЕДМЕТА ИЗОБРЕТЕНИЯ:

G06F 17/40 *(2006.01)*
G16Y 40/00 *(2020.01)*

Согласно Международной патентной классификации (МПК)

Б. ОБЛАСТЬ ПОИСКА:

Просмотренная документация (система классификации и индексы МПК)
G06F 17/00-17/40, G16Y 40/00

Электронная база данных, использовавшаяся при поиске (название базы и, если, возможно, используемые поисковые термины)
Google Patents, Espacenet, (ИС «Поисковая платформа» Роспатент), ЕАПАТИС

В. ДОКУМЕНТЫ, СЧИТАЮЩИЕСЯ РЕЛЕВАНТНЫМИ

Категория*	Ссылки на документы с указанием, где это возможно, релевантных частей	Относится к пункту №
A	US 2018/0299849 A1 (SCHNEIDER ELECTRIC SYSTEMS USA, INC), 18.10.2018	1-5
A	WO 2018/004843 A1 (FISHER CONTROLS INTERNATIONAL LLC), 04.01.2018	1-5
A	US 2018/0300437 A1 (ROCKWELL AUTOMATION TECHNOLOGIES, INC), 18.10.2018	1-1
A	US 2017/0038754 A1 (SIEMENS AKTIENGESELLSCHAFT), 09.02.2017	1-5

☐ последующие документы указаны в продолжении

* Особые категории ссылочных документов:
«A» - документ, определяющий общий уровень техники
«D» - документ, приведенный в евразийской заявке
«E» - более ранний документ, но опубликованный на дату подачи евразийской заявки или после нее
«O» - документ, относящийся к устному раскрытию, экспонированию и т.д.
"P" - документ, опубликованный до даты подачи евразийской заявки, но после даты испрашиваемого приоритета"
«T» - более поздний документ, опубликованный после даты приоритета и приведенный для понимания изобретения
«X» - документ, имеющий наиболее близкое отношение к предмету поиска, порочащий новизну или изобретательский уровень, взятый в отдельности
«Y» - документ, имеющий наиболее близкое отношение к предмету поиска, порочащий изобретательский уровень в сочетании с другими документами той же категории
«&» - документ, являющийся патентом-аналогом
«L» - документ, приведенный в других целях

Дата проведения патентного поиска: **12/09/2023**

Уполномоченное лицо:
Начальник отдела механики,
физики и электротехники

Д.Ф. Крылов