УДК 004.032.26
doi:10.31799/1684-8853-2024-6-15-26
EDN: HNIROD

Научные статьи
Articles

Алгоритм динамического распределения обработки изображений в облачных системах интеллектуального видеонаблюдения

Н. А. Жукова[а], *доктор техн. наук, профессор, orcid.org/0000-0001-5877-4461, nazhukova@mail.ru*
А. Н. Субботин[б], *аспирант, orcid.org/0000-0002-4823-6288*
[а]*Санкт-Петербургский Федеральный исследовательский центр РАН, 14-я линия В. О., 39, Санкт-Петербург, 199178, РФ*
[б]*Санкт-Петербургский государственный электротехнический университет «ЛЭТИ», Профессора Попова ул., 5, Санкт-Петербург, 197376, РФ*

Введение: *наличие мощных серверов в облачной среде позволяет облачным системам видеонаблюдения выполнять сложные процессы обработки изображений с применением методов машинного обучения искусственных нейронных сетей, включая нейросетевые методы; кроме того, появляется возможность строить такие процессы в динамике.* ***Цель:*** *разработать алгоритм динамической распределенной обработки изображений для интеллектуальной системы видеонаблюдения в облачной среде.* ***Результаты:*** *предложен новый алгоритм динамического распределения обработки изображений, позволяющий выполнять сложные синтезированные процессы обработки данных систем видеонаблюдения в облаке, в динамике строить маршруты передачи изображений в облако, а также распределять нагрузку между многими виртуальными машинами. Алгоритм распределения обработки выполняется разработанным оркестратором, который может быть размещен в облаке или в тумане. Полученные результаты обеспечили новые возможности в динамике распределять выполнение синтезированных процессов обработки изображений по разнотипным виртуальным машинам, размещенным на разных физических серверах.* ***Практическая значимость:*** *применение полученных результатов на практике при решении задач интеллектуального видеонаблюдения в метрополитене позволило за счет динамического распределения обработки изображений повысить скорость обработки данных (снизить время) более чем в 2,5 раза и за счет применения сложных синтезированных процессов обработки повысить точность определения событий более чем на 11 %.*

Ключевые слова *— распределенная обработка изображений, синтез процессов, облачная среда, логическая модель.*

Для цитирования: Жукова Н. А., Субботин А. Н. Алгоритм динамического распределения обработки изображений в облачных системах интеллектуального видеонаблюдения. *Информационно-управляющие системы*, 2024, № 6, с. 15–26. doi:10.31799/1684-8853-2024-6-15-26, EDN: HNIROD
For citation: Zhukova N. A., Subbotin A. N. Dynamic distribution algorithm for image processing in cloud-based intelligent video surveillance systems. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2024, no. 6, pp. 15–26 (In Russian). doi:10.31799/1684-8853-2024-6-15-26, EDN: HNIROD

Введение

Повышение быстродействия современных вычислительных средств предоставляет большие возможности для интеллектуальной распределенной обработки информации, в частности обработки видеопоследовательностей [1]. При обработке изображений в настоящее время широко применяются туманные технологии. Обработка видеопоследовательностей с использованием туманных вычислительных сред (туманных сред) рассмотрена в статьях [2–4]. Туманные вычислительные среды имеют ряд преимуществ перед другими вычислительными технологиями (снижение задержек в передаче данных, обеспечение конфиденциальности информации и пр.). Однако обработка данных в туманных вычислительных средах требует значительных финансовых затрат, при этом быстродействие туманных сред напрямую зависит от их стоимости. В то же время стоимость аренды вычислительных средств в облаке является относительно низкой. Для примера, стоимость владения сервером Cisco UCS C480 ML M5 Deep Learning Rack Server составляет около $ 15 000 с накладными расходами (электроэнергия и пр.), а стоимость аренды на Microsoft Azure — примерно $ 100 в месяц. В облаках, как правило, используются дорогие и мощные серверы. Таким образом, в облаке изображения обрабатываются несравнимо быстрее, чем в туманной вычислительной среде, что позволяет выполнять сложные процессы обработки с применением методов машинного обучения искусственных нейронных сетей, включая нейросетевые методы [5–8]. Однако на практике применение заранее построенных процессов обработки оказывается не всегда достаточным при обработке видеопоследовательностей, что обусловлено частым изменением условий, в которых ведется видеонаблюдение. Требуется выполнять

синтез процессов обработки данных в динамике. Динамический синтез процессов позволяет значительно повышать точность обработки за счет учета особенностей данных и условий, в которых обработка выполняется [9]. Однако синтез процессов требует дополнительных вычислительных ресурсов, что может привести к задержкам в обработке данных.

Проблему задержки в обработке данных можно решить за счет распределения обработки между виртуальными машинами (ВМ), используя для этого оркестратор, который в динамике строит маршруты и распределяет нагрузку между многими ВМ. В настоящее время обработка, как правило, выполняется на одном сервере. Существующие алгоритмы динамического распределения данных (оркестровки) между компьютерами (или виртуальными машинами, или контейнерами), такие как Docker, Open VZ, VirtualBox, LXC, Containerd, Vagrant, Microsoft Azure, Kubernetes (K8), Podman, ZeroVM и др., обладают существенным недостатком: невозможностью работать с контейнерами (виртуальными машинами) другого типа, отличного от программы оркестрации [10]. Также следует отметить, что значительное большинство алгоритмов оркестрации представляют коммерческую тайну. Динамическое распределение обработки по многим машинам позволит в разы сократить время, необходимое на обработку изображений в облаке.

Таким образом, с учетом сложности процессов обработки в системах интеллектуального видеонаблюдения, большого объема данных, требующих обработки, а также высокой стоимости использования туманных сред представляется целесообразным значительную часть обрабатывать в облаке. При этом в туманной среде могут выполняться отдельные процессы, в частности предобработка данных, предусматривающая удаление цветности, изменение размерности, глубины и пр., которые должны проходить в непосредственной близости от камер видеонаблюдения.

В статье рассматривается модель данных и логическая модель обработки изображений в облаке, реализованные в виде программных компонент единой системы распределенных вычислений [11], обеспечивающей синтез и выполнение процессов обработки изображений в облачной среде. Предложенное решение отличается от существующих тем, что позволяет в динамике синтезировать процессы обработки и в динамике распределять их проведение по разным ВМ. Поддерживается возможность синтезировать сложные процессы обработки с применением различных алгоритмов машинного обучения, и в том числе искусственных нейронных сетей. При распределении обработки предоставляется возможность работать с разнотипными ВМ и физическими серверами, а не контейнерами одного типа.

Целью исследования является повышение эффективности облачных систем интеллектуального видеонаблюдения за счет выполнения синтезированных процессов обработки данных на многих ВМ.

В общем виде задача синтеза процессов обработки состоит в нахождении процесса P_0 на заданный момент времени t, в результате выполнения которого достигается экстремум основного показателя эффективности построения процессов E_0 при ограничениях на вспомогательные показатели E_k, $1 \le k \le n$, где n — общее число показателей эффективности.

В данной работе для оценки эффективности использованы показатели, разработанные авторами в сотрудничестве со специалистами в области облачного анализа изображений и на основании исследований, представленных в работах [12, 13].

Сравнение методов обработки изображений, применяемых в условиях ограничений производительности пользовательского компьютера (устройства)

В работах [13, 14] рассматриваются различные подходы к обработке видеоданных в облачной среде. Применяются следующие способы передачи видео: постоянное TCP/IP-соединение; постоянное UDP-соединение; использование туманной среды.

При постоянном TCP/IP-соединении чаще всего используется протокол HTTP/HTTPS, видео передается фрагментами от одной секунды до 10 мин и более.

UDP-соединение предполагает трансляцию видеопотока без контроля ошибок, что может привести к снижению качества. Для решения проблемы задержек в канале связи используется буферизация данных.

Благодаря применению туманной среды обеспечиваются возможности буферизировать видео, выполнять синхронизацию кадров из различных источников, кроме того, в туманных средах, как правило, выполняется предварительная обработка данных. Также в туманной среде обеспечивается маршрутизация, необходимая для передачи видео облачному серверу или кластеру серверов по доступным каналам связи.

Обработку данных можно также выполнять и на облачных платформах, предоставляемых такими корпорациями, как Microsoft, Google, Amazon и др., на которых, помимо прочего, доступны вычислительные мощности.

Основными преимуществами использования таких платформ являются возможность установки операционных систем Windows и Linux, подключения по SSH и удаленного администрирования, а также возможность использовать необходимые программы и скрипты. К основным недостаткам относятся ограниченный выбор версий операционной системы Linux, а также доступ по протоколам RDP.

Также ряд компаний предоставляет посреднический доступ к облачным серверам Microsoft, Amazon и др. с собственными настройками.

В целом, можно выделить следующие основные отличия облачных вычислительных сред от туманных. Облачные среды обеспечивают более высокую производительность, предоставляют возможности кластеризации и выделения ресурсов, а также программное обеспечение для управления.

На эффективность обработки изображений с применением машинного обучения оказывают влияние следующие факторы:

— адаптация сервера под задачи машинного обучения (специальный GPU, платы расширения);

— наличие программного обеспечения для оптимизации вычислений и необходимых библиотек (OpenCV, Emgu CV, TensorFlow, Keras);

— возможность установки совместимого программного обеспечения: интерпретатора Python, среды разработки, операционной системы.

В табл. 1 представлен список существующих систем интеллектуального видеонаблюдения и их возможностей: $p1$ — работа в туманной среде, $p2$ — работа в облаке, $p3$ — возможность выполнять обработку на многих машинах, $p4$ — поддержка видеокамер по коаксиальному кабелю, отличному от TCP/IP, $p5$ — ограничения на количество камер (не более пяти-восьми, например), $p6$ — поддержка мобильных устройств (смартфонов и пр.), $p7$ — устойчивость к низким и высоким температурам, $p8$ — возможности системы применять методы машинного обучения, включая искусственные нейронные сети. Все системы имеют видеорегистраторы, но небольшой емкости по сравнению с туманным сервером или облачным хранилищем. Все системы имеют программу или веб-сайт для удаленного наблюдения. Сравнительный анализ возможностей существующих систем и новой предлагаемой авторами системы SubWatch приведен в табл. 1.

Дополнительная информация о системах доступна на сайтах: Ginzzu (http://ginzzu.com/), Ezviz (https://www.ezviz.com/), Camdrive (https://www.camdrive.com/), PS-link (https://ps-link.ru/), Kvadro (https://4vision.ru/), Ivue (https://ivuecamera.com/), HiWatch (https://ru-hiwatch.com/).

■ ***Таблица 1.*** Сравнение возможностей систем интеллектуального видеонаблюдения

■ ***Table 1.*** Comparison of capabilities of intelligent video surveillance systems

Система	Возможность							
	$p1$	$p2$	$p3$	$p4$	$p5$	$p6$	$p7$	$p8$
1. Ginzzu	–	–	–	–	–	–	–	–
2. Ezviz	–	–	–	–	+	+	–	–
3. Camdrive	–	+	–	–	+	+	–	–
4. PS-link	+	+	+	–	+	+	+	+
5. Kvadro	+	–	–	–	+	+	+	+
6. Ivue	–	–	–	–	–	+	+	–
7. HiWatch	+	–	–	–	+	+	+	+
8. SubWatch	+	+	+	+	+	+	+	+

Модели обработки изображений в облачных системах

При описании предлагаемого решения рассматриваются обобщенная и логическая модели обработки изображений в облачных средах. Предлагается новый алгоритм распределенной обработки данных в системах видеонаблюдения, обеспечивающий динамическую маршрутизацию и перенаправление видеопотоков при передаче данных в облако для их последующей обработки.

Обобщенная модель обработки данных в облачных системах

Обобщенная модель обработки видеопоследовательностей представлена на рис. 1. Установленные на объектах видеокамеры передают изображения по кабелю или оптоволокну через физическую сеть маршрутизаторов с использованием протокола TCP/IP, что позволяет обеспечить низкую нагрузку на каналы связи.

Передаваемые в облако данные размещаются на выделенном сервере с дисковым массивом, обработку данных выполняет центральный облачный сервер. В качестве серверов могут использоваться серверы для установки в стойки, предоставляемые компаниями Cisco, Dell EMC, Fujitsu, Hewlett Packard Enterprise, IBM, Intel, Lenovo, Supermicro, HUAWEI и др. В облачной среде разработчик программного обеспечения работает с «Озером данных» (Data Lake), которое представляет собой абстрактное дисковое пространство. В отличие от других облачных хранилищ, «Озеро данных» обладает способностью к динамическому расширению, которое выполняется по мере необходимости. В настоящее время «Озера данных» применяются в основном

в случаях, когда требуется использовать модели и методы машинного обучения.

После поступления данных на сервер дальнейшая их обработка реализуется в два условных потока. В первом потоке обрабатываются неструктурированные данные, в большинстве случаев это данные из «Озер данных» [15–18], при этом выполняется обработка первичных видеозаписей в их исходном формате и качестве. Во втором потоке проводятся аналитика и статистическая обработка последовательностей кадров [19], хранящихся в базах данных (MariaDB, MongoDB, PostgreSQL, Microsoft SQL, RavenDB, CouchDB и др.): информация о найденных объектах, выявленных событиях, уровне опасности, принятые меры операторами видеонаблюдения и вспомогательными службами.

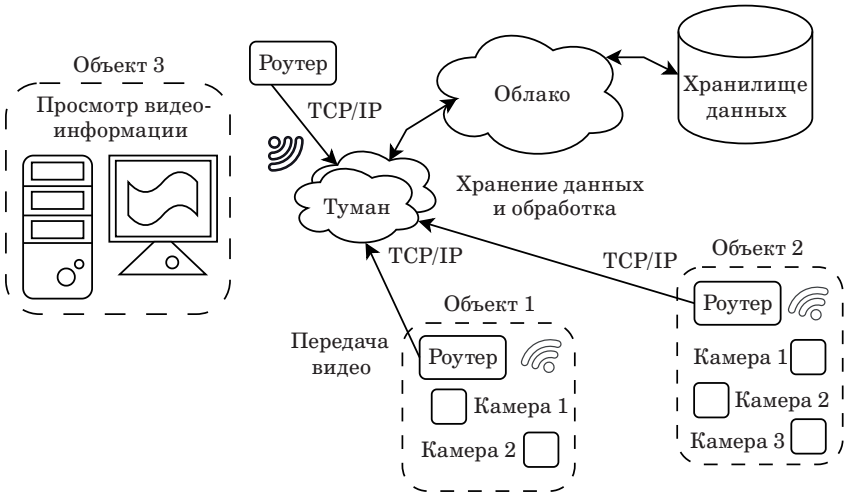
Полученные результаты и фрагменты исходных данных предоставляются пользователям или операторам видеонаблюдения в доступном формате через веб-сайт [20], через приложения для мобильных устройств iOS или Android Lnx [21, 22] и автономные приложения, подключаемые к облачному серверу [23, 24]. Веб-сайт позволяет настраивать права доступа к данным для различных пользователей, предоставляет возможность просмотра видеоархива, получения статистики по обнаруженным объектам. При анализе статистических данных может применяться фильтрация по событиям и выполняться построение графиков.

Приложения могут постоянно размещаться в оперативной памяти смартфона [25], обеспечивая своевременное уведомление пользователей о выявленных событиях.

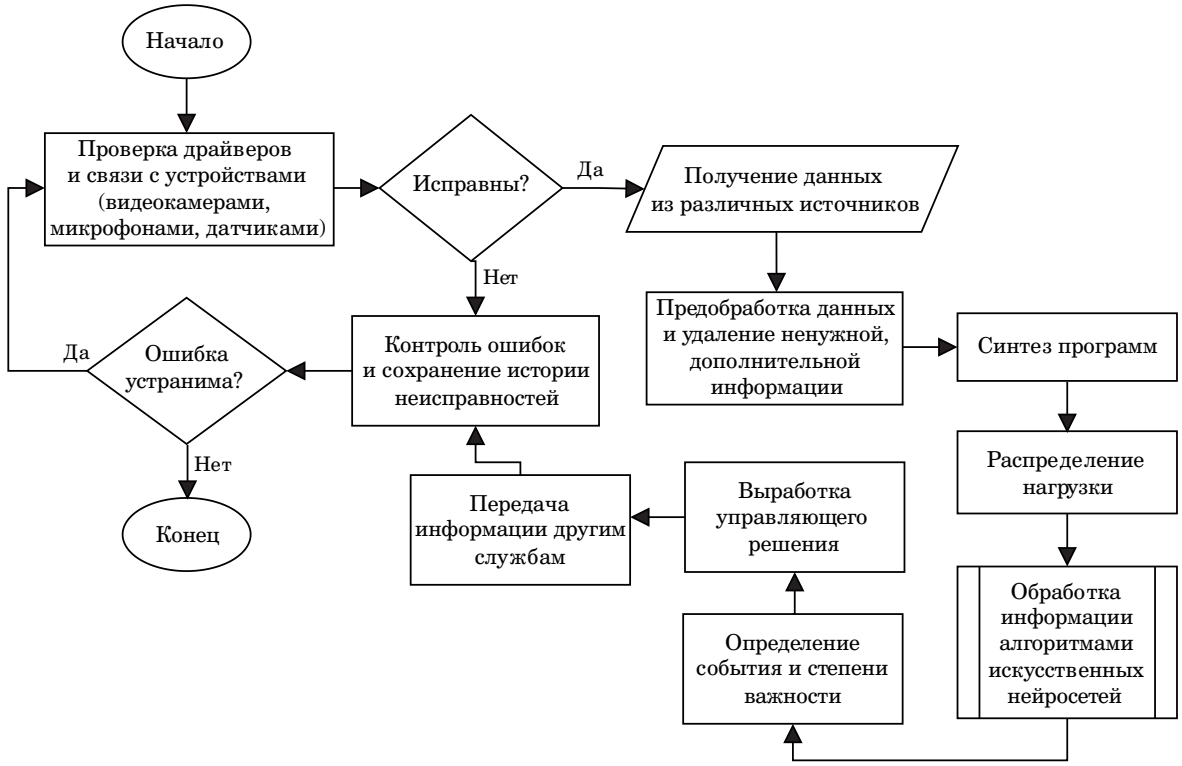
В соответствии с представленной на рис. 1 обобщенной моделью в облако поступают данные от многих устройств. Данные могут быть представлены в виде серии кадров, звуковых потоков, результатов измерений с датчиков. С учетом большого объема и разнородности данных необходимо их слияние и обобщение, однако при этом не должна снижаться точность определения событий при обработке данных в облаке. Возможности систем обработки данных позволяют независимо выполнять обработку различных типов данных с учетом их особенностей, а также проводить совместную обработку всех данных или нескольких типов данных [26–28]. Для обработки данных могут использоваться различные алгоритмы, предоставляемые существующими библиотеками машинного обучения, в том числе TensorFlow, OpenNN, NLTK, mlpack, Theano, Matplotlib, Armadillo. Как правило, процессы обработки данных реализуются в несколько этапов согласно стандарту CRISP-DM (CRoss Industry Standard Process for Data Mining, http://www.crisp-dm.org/), что предполагает последовательное и (или) параллельное выполнение многих отдельных алгоритмов. При этом в зависимости от результатов, полученных после выполнения каждого из этапов, может возникнуть необходимость в перестройке последующих этапов обработки. Построение и перестройка процессов обработки данных в динамике осуществляются за счет синтеза процессов обработки [29, 30].

В общем виде систему интеллектуального видеонаблюдения можно представить как объекты, соединенные с облачным сервером (см. рис. 1). Следует отметить, что для решения отдельных задач, в первую очередь задач, относящихся к предобработке, могут использоваться туманные вычислительные средства.

В основу алгоритма обработки данных систем видеонаблюдения положены методы синтеза [31–33] (рис. 2). Алгоритм в режиме выполнения



■ ***Рис. 1.*** Обобщенная модель систем обработки видеопоследовательностей
■ ***Fig. 1.*** Generalized logical model of video image processing systems



■ ***Рис. 2.*** Алгоритм обработки данных системы видеонаблюдения
■ ***Fig. 2.*** The algorithm for processing data of the video surveillance system

обработки данных предусматривает реализацию следующих шагов. На первом шаге осуществляется загрузка программного обеспечения в облако (Старт), проверка драйверов и устройств связи (роутеров, сетевых плат расширений и адаптеров; коммуникационные протоколы, тестовые ответы от устройств должны приходить за отведенное время) с видеокамерами, микрофонами, сенсорами различного предназначения. На следующем шаге осуществляются обмен управляющими сигналами и получение данных из различных источников. Далее анализируются характеристики обрабатываемых данных и условий обработки, оценивается возможность применения для их обработки процессов, построенных или на этапе обучения системы, или ранее при обработке данных. Если существующие процессы не применимы в текущих условиях, то проводится синтез новых процессов обработки или перестройка имеющихся. Например, в процессе синтеза могут исполняться следующие действия: определение алгоритма преобразования данных к единому формату в соответствии с описанием их структуры и состава; выбор алгоритма преобразования данных с учетом их характеристик; выбор архитектуры нейронной сети исходя из обрабатываемых данных и решаемой задачи. Для выполнения синтезированных процессов осуществляется распределение нагрузки между ВМ. На основе результатов обработки выдается управляющая последовательность. Так, при выявлении события, классифицированного как «пожар», в течение 10 с передается агрегированная информация (0,81; 0,83; 0,85; 0,89; 0,91; 0,93; 0,95; 0,97; 0,96; 0,95) с периодичностью один раз в секунду. Этот процесс представлен на схеме алгоритма как «Определение события и степени опасности». На следующем шаге вырабатывается управляющее решение. Далее информация передается ответственным службам, в рассматриваемом примере пожарным, скорой, полиции и др. На последнем этапе происходит логирование (сохранение информации о событии, принятом решении, дополнительной технической информации).

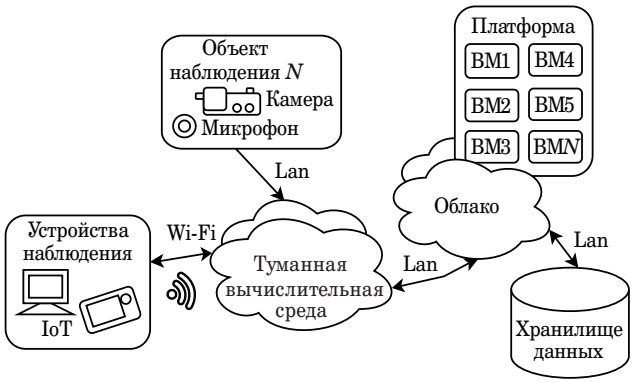
Также на шаг логирования может быть выполнен переход после старта работы алгоритма в случае, если при тестировании устройств были выявлены неисправности. Тогда при логировании сохраняется следующая информация: дата и время обнаружения неисправностей, типы и серийные номера устройств, расположение на объекте наблюдения и категория объекта, причина возможной неисправности и принятые меры для устранения (замена, ремонт, модернизация и пр.).

Для выполнения синтезированных процессов обработки на их основе выполняется синтез программ обработки, при котором определяется схема вызова алгоритмов обработки данных и ее представление в виде машинно-интерпретируемых команд. Более подробно вопросы синтеза программ на основе процессов обработки рассмотрены в работе [34].

Логическая модель обработки изображений в облачных системах

Предлагаемая распределенная модель для обработки изображений в облаке (сервере) предназначена для использования в системах с клиент-серверной архитектурой. Видеопоследовательности передаются в форме запакованных каталогов с кадрами, дополнительно, также в запакованном виде, передаются служебные данные. Методы, применяемые для передачи данных, например [35], позволяют использовать различные серверы на облачных платформах, но требуется наличие оркестратора (программы-менеджера по распределению данных в облаке) и хранилища информации на облачной платформе. Оркестратор необязательно должен находиться в облаке, он также может размещаться в туманной вычислительной среде (рис. 3).

Облачная платформа предоставляет возможность запуска из браузера серии ВМ (обычно не более 1000 на одном сервере и не более 12 750 на одном кластере). В качестве операционной системы, как правило, используется Linux (SLES, CentOS, RHL, ALT и пр.), но иногда серверные версии Windows (2016, 2022), которые нельзя отнести к популярным, но которые встречаются у партнеров Microsoft (AWS, Azure и пр.). Работа возможна на группе серверов, соединенных высокопроизводительными оптическими интерфейсами (ANSI/TIA-942-B-1 от TR-42.1), которые рассматриваются как единое целое. При опциональном добавлении видеокарт Nvidia (SoC) и свободных ядер ЦПУ и RAM снижается время, затрачиваемое на обработку данных. Использование серии контейнеров также снижает время на обработку одного кадра. Смартфоны могут использовать серверы подкачки (туманные среды) для мгновенной загрузки контента [36, 37] и снижения ошибок физического уровня при передаче данных — фликеров из-за электронесогласованности (помех физической природы).



■ ***Рис. 3.*** Логическая модель обработки изображений
■ ***Fig. 3.*** Logical model of image processing

До настоящего времени очень популярными остаются платформы на VK (Вконтакте Облако), Яндекс (Облако и Диски), МТС (Облако, Экосистема), Сбербанк (Облако и Платформа). Каждое облако предоставляет веб-сайт (платформу) для детального редактирования каждого параметра, начиная от процессора и оперативной памяти до скриптов машинного обучения и отдельных приложений в контейнере.

Операционная система ALT Linux 5.0 Server KDE версии 10.3 (https://www.basealt.ru/) на данный момент является самой популярной на облачных платформах в России, поскольку поддерживает стандарты безопасности ГОСТ-2012 в OpenSSL и имеет полную локализацию всех интерфейсов. Для непосредственного создания форм, меню и кодирования применяется интегрированная компонента в системе ALT Linux под названием Lazarus RAD IDE 3.4 DEB 64 bits (https://www.lazarus-ide.org/) с догруженными модулями SubEvrt-v.0.17.4a и SubSnpe-v.0.26.71s1, поддерживающими сборку под ALT Linux 5.0, Windows 11 и MacOS Sequoia 15. Предустановленный компонент libmariadb.lib в пакете Lazarus RAD IDE предоставляет объекты для работы с реляционными базами данных для логирования. Хранение основных данных в хранилище происходит по системной команде «cp» (https://losst.pro/komanda-cp-v-linux), интегрированной в ядро ALT Linux 5.0.

Контейнер или ВМ является одним элементом (узлом) в рассматриваемой модели. Для распределения нагрузки между многими ВМ авторами разработан алгоритм оркестрации (рис. 4). Вначале программа-оркестратор читает настройки в каталоге с программой, затем устанавливает подключения к ВМ, используя список настроек, которые были сохранены. На следующем шаге оркестратор закачивает пробный пакет по IP-адресу каждой ВМ с целью определить производительность ВМ и строит список машин от машин с самой высокой производительностью к машинам с самой низкой производительностью. На следующем шаге происходит динамическое построение маршрута передачи данных и потом динамическое распределение обработки между ВМ. Далее идет постоянная работа в цикле: загрузка пакета с кадрами и дополнительной информацией, выполнение скрипта на

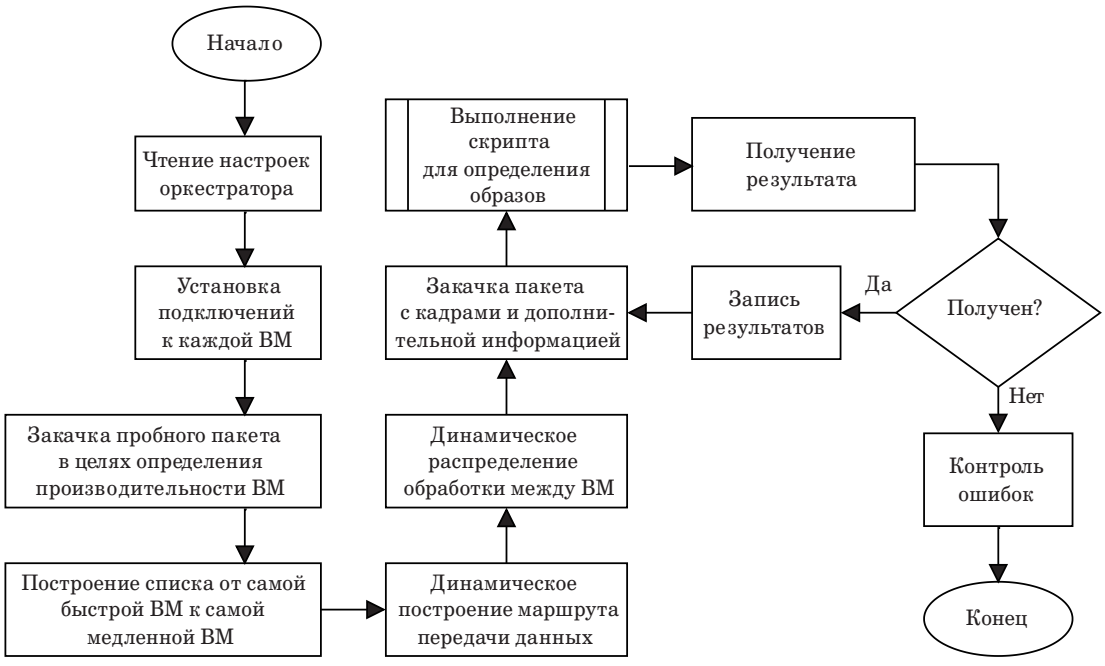
Python 3.13.0 и TensorFlow 2.18, получение результата (названия объектов, проценты). Если результат получен, тогда результат и характеристики объектов записываются в базу данных и выполняется загрузка следующего пакета на ВМ. Если результат не получен, тогда выполняется контроль ошибок и завершение работы программы («Конец» на рис. 4).

Авторы данного исследования предлагают использовать контейнеры, которые не зависят от платформы, а основываются на ВМ. Установленная программа на одной ВМ получает архив кадров (серию JPEG или PNG в зависимости от модели встроенного программного обеспечения в камере). Информация о местоположении камеры (GRPS-координаты, цех, автостоянка, проходная и др.), температуре, давлении, погодных условиях (дождь, снег, ветер и пр.), посторонних предметах (листья, животные и т.д.) передается как файл RSS2.0 и XML1.1 (+KML). Упаковщиком является программа под ALT Linux с названием 7-Zip v.24.08. Файл в архиве с названием info-subvideos.xml, помимо всей перечисленной информации, содержит служебную информацию о модели видеокамеры, фильтрах производителя, которые установлены по умолчанию (синева, зелень, оттенок серого и пр.), технических особенностях фокуса, уровне детализации, типе день/ночь и др.

Виртуальная машина (большой контейнер) с ALT Linux Server 5.0 не выполняет тестирование устройств (поскольку устройства являются виртуальными, и вторую проверку проходить не обязательно — этим занимается host-система) и сразу запускается из спящего режима, при этом она загружается как большой файл (объект) в стек оперативной памяти. Данный процесс занимает считаные секунды, ничем не уступая контейнерам (например, Docker), но превосходит контейнерные технологии благодаря более стабильной работе (по причине высокой изолированности от других программ). Программа для работы с данными в контейнере запускается мгновенно и ждет подключений от оркестратора из туманной вычислительной среды.

Активация программы из режима ожидания происходит в момент запроса на подключение от оркестратора. При получении архива seria-09-10-2024 xxx.7z происходит распаковка файлов из консоли Linux и обработка скриптом на Python. Имя архива содержит время, дату и случайное число, чтобы имя архива было уникальным.

Протокол HTTP(s) при передаче архивов шифрует информацию (SSL, Russian Trusted Sub CA of The Ministry of Digital Development and Communications) и защищает конфиденциальность данных на промежуточных устройствах (сетевых мониторах, роутерах, маршрутизаторах, снифферах и пр.). При использовании такого протокола передача информации не вызывает сложностей, поскольку есть модули для работы с HTTP(s) почти на всех языках программирования, а в некоторых случаях поддерживаются объекты из среды разработки, как в Lazarus IDE.



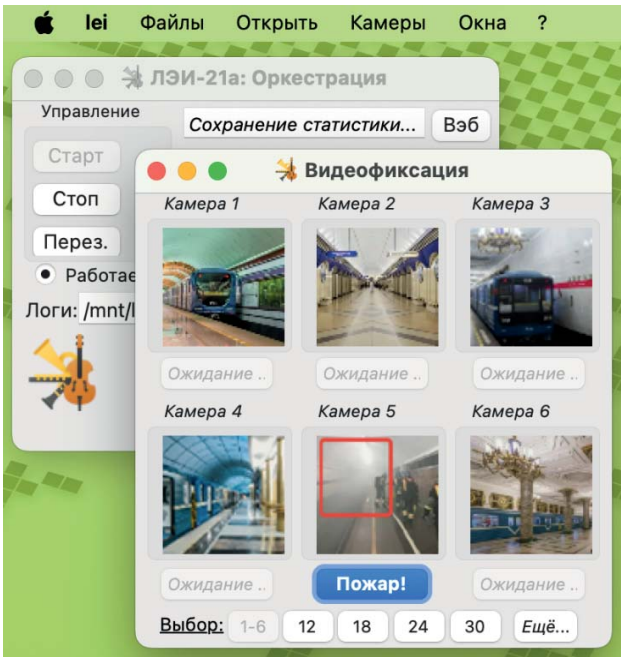
■ ***Рис. 4.*** Алгоритм работы программы-оркестратора для обработки изображений
■ ***Fig. 4.*** Flow chart of the image processing orchestrator program

Результаты обработки данных (кадры и XML-файлы) сохраняются в любом случае (успешно или не успешно прошла обработка). В базе данных MariaDB 11.6.1 сохраняется информация о событиях, например, найденных предметах (объектах). Полную информацию можно получить только с правами администратора на вебсайте, где в таблицах XHTML v.1.1SE указаны все события с детализацией во всплывающих фреймах. Пользователи облачных систем, например операторы по диагностике событий, используют приложение на смартфоне, которое содержит компонент «браузер». Подключение по HTTP(s) держится постоянно, от чего синхронизация событий между смартфоном и сервером возможна за сотую долю секунды, информация о событии в приложение на смартфоне поступает практически мгновенно. Нет двух этапов: согласования протоколов HTTP(s) и постройки маршрутов TCP/IP.

Пример применения алгоритма

В настоящий момент в Петербургском метрополитене (https://www.metro.spb.ru/) используется система интеллектуального видеонаблюдения (ИСВН) под названием Milestone XProtect Corporate 2016 (https://www.milestonesys.com/). В состав системы входят приложение XProtect Smart Client, видеокамеры Axis (https://www.axis.com/) и программное обеспечение P-Iris, которое непосредственно установлено на видеокамерах Axis. От каждой видеокамеры Axis идет два потока видеоинформации, один из которых сохраняется на сервере в течение 30 дней, а второй отправляется на видеообработку, после чего информация поступает в программу для оператора Smart Client под Windows 7. Выявленные системой события просматриваются оператором специального центра управления. Критически важные видеофрагменты дублируются на отдельный резервный сервер.

Авторы исследования разработали новое облачное приложение (рис. 5) для пакетной обработки изображений, представленных в виде серии кадров [37]. Установка и конфигурация приложения выполняются с использованием удаленного рабочего стола, предоставляющего доступ к ВМ и рабочему окружению KDE. Для этого используются программы Remote Manipulator System (RMS): Netop Remote Control (http://www.netop.ru/) и TeamViewer (https://www.teamviewer.com/) под управлением ALT Linux Server KDE. Выбор обусловлен программной совместимостью со всеми ведущими операционными системами (Windows, MacOS, FreeBSD, Linux, Cent OS, iOS, Android) и высокой производительностью при



■ ***Рис. 5.*** Внешний вид разработанной программы для обработки изображений в облачной среде

■ ***Fig. 5.*** The appearance of the program for processing images in the cloud

работе с различными версиями Linux из сред Windows версий 10/11.

При проведении экспериментальных исследований с использованием разработанного приложения задавались ограничения на мощность вычислительной среды в облаке (не более 15 700 flops), мощность вычислительной среды в тумане (не более 7900 flops), количество ВМ (не более девяти), производительность памяти (не более 6400), временной интервал, на котором должна быть выполнена обработка, 150–4500 с.

Для оценки эффективности предложенного решения, предполагающего синтез программ обработки изображений и распределенную обработку в облаке, выполнялось сравнение разработанной системы SubWatch с существующими системами, приведенными в табл. 1, и системой ИСВН, используемой в настоящее время для обработки видеоданных в метрополитене.

В качестве исходных данных при проведении экспериментального исследования использовались данные о проникновении лиц в служебные зоны Петербургского метрополитена. Камеры видеонаблюдения располагались в служебных зонах метрополитена. С камер было получено 326 735 пакетов, в каждом из которых была размещена серия кадров (12 к/с). Для обработки данных применялся алгоритм нейросетевой обработки TensorFlow 2.18 (https://www.tensorflow.org) из фреймворка Keras 3.0 (https://keras.io), на-

писанный на языке Python 3.13.0 (https://www.python.org) из среды разработки PyTorch 2.5 (https://pytorch.org).

Для работы с программой не требуется ручного внесения данных о камерах видеонаблюдения в файлы cfg-swatch.ini и setg-swatch.ini, так как программа самостоятельно получает необходимую информацию из файла JSON и XML, входящего в состав каждого 7zip-архива с кадрами.

Программное обеспечение было развернуто на нескольких ВМ, обладающих идентичными техническими характеристиками (Intel Broadwell с Nvidia Tesla v100, 2 vGPU, 8 ГБ ОЗУ, 2048 ГБ SSD). В качестве платформы использовался облачный сервис Yandex.Cloud с предустановленной операционной системой ALT Linux.

В ходе экспериментальных исследований измерялось время обработки изображений при использовании одной, трех и пяти ВМ. Полученные результаты показали, что при увеличении количества ВМ время обработки серии изображений уменьшается в среднем в 2,63 раза по сравнению с использованием одной ВМ. Проведен сравнительный анализ по шести критериям эффективности (табл. 2).

1. Сравнивалось время обработки изображений предлагаемой системой и существующими аналогами при обработке данных на пяти ВМ.

2. Вторым показателем эффективности является точность определения событий. За счет синтеза процессов удалось повысить точность определения событий на 11,25 %.

3. Благодаря использованию оркестратора, обеспечивающего управление передачей данных из одной управляющей программы, сократилось время передачи данных в 0,91 раза. Время не тратится на согласование протоколов передачи данных.

4. Оперативность принятия решений выросла на 15,3 %, что свидетельствует о повышении скорости реакции системы на события.

5. Важным показателем является также уровень удовлетворенности системой пользователями. Для оценки применялся стандартный опросник лояльности из 100 вопросов. Обработка данных проводилась с помощью сервисов Google. Forms (https://docs.google.com/forms/) и Google. Tables. Полученный результат 33,7 % указывает на положительное восприятие системы пользователями (использовалась 100-балльная шкала).

6. Еще одним показателем, характеризующим эффективность системы, является точность принятия управленческих решений. Данные, предоставленные руководителями подразделений и служб, были агрегированы в Google.Tables. Результаты их обработки позволили установить повышение точности принятия решений более чем на 15 %.

■ ***Таблица 2.*** Результаты оценки эффективности предлагаемого решения

■ ***Table 2.*** Results of the evaluation of the effectiveness of the proposed solution

Система	Данные показателей эффективности					
	мс	%	мс	%	%	%
1. Ginzzu	4	0,79	–	–	–	–
2. Ivue	6	0,73	–	–	–	–
3. Camdrive	5	0,81	–	–	–	–
4. PS-link	2	0,94	–	–	–	–
5. HiWatch	3	0,87	–	–	–	–
6. Ezviz	3	0,89	–	–	–	–
7. Kvadro	2	0,91	–	–	–	–
8. Milestone XProtect	2	0,89	132	87	74	83
9. SubWatch	1,283	0,97	120	73	49	76
Больше, чем среднее	2,63	11,25	0,91	15,3	33,7	15

Заключение

В статье рассмотрены модели обработки изображений для облачных систем интеллектуального видеонаблюдения. Авторами предложен логически обоснованный алгоритм, позволяющий выполнять сложные синтезированные процессы обработки данных, предусматривающие применение алгоритмов машинного обучения, в частности нейросетей, на многих ВМ, развернутых на облачных платформах.

Предлагаемое решение позволяет строить системы интеллектуального видеонаблюдения, которые, в отличие от большинства существующих систем, используют облачную среду для обработки изображений и хранения данных. При обработке данных в облачной среде обеспечивается возможность учитывать характеристики данных и условия их обработки за счет синтеза процессов обработки и их перестройки, что становится возможным благодаря наличию в облаке достаточного объема вычислительных ресурсов. Представленный алгоритм динамического распределения обработки изображений в облачных системах интеллектуального видеонаблюдения обеспечивает сокращение времени передачи данных в облако и их обработки в облачной среде за счет динамического распределения процессов обработки между ВМ.

Разработано приложение для облачной среды, реализующее пакетную обработку серии кадров с дополнительной информацией. Приложение успешно протестировано на примере обнаруже-

ния незаконного проникновения посторонних лиц в служебные зоны метрополитена; продемонстрировано сокращение времени обработки данных (более чем в 2,5 раза) и повышение точности распознавания событий (на 11 %).

Созданное приложение также способствует повышению оперативности работы сотрудников, удовлетворенности системой пользователей и точности принятия управленческих решений.

Потенциальные области применения данного решения весьма широки: от мониторинга поведения пассажиров на платформах до обработки данных с камер, установленных на различных объектах инфраструктуры, таких как туннели железной дороги, шахты, эскалаторы и служебные помещения.

Финансовая поддержка

Работа выполнена при поддержке государственного бюджета, проект № FFZF-2022-0006.

Литература

1. **Sarkar S., Chatterjee S., Misra S.** Assessment of the suitability of fog computing in the context of internet of things. *IEEE Transactions on Cloud Computing*, 2018, no. 6(1), pp. 46–59. doi:10.1109/TCC.2015.2485206
2. **Subbotin A. N., Zhukova N. A., Man T.** Architecture of the intelligent video surveillance systems for fog environments based on embedded computers, *2021 10th Mediterranean Conf. on Embedded Computing (MECO)*, 2021, pp. 1–8. doi:10.1109/MECO52532.2021.9460270
3. **Subbotin A.** Data processing in foggy computing environments for machine learning. *2021 II Intern. Conf. on Neural Networks and Neurotechnologies (NeuroNT)*, 2021, pp. 51–53. doi:10.1109/NeuroNT53022.2021.9472203
4. **Subbotin A.** Applying machine learning in fog computing environments for panoramic teeth imaging. *2021 XXIV Intern. Conf. on Soft Computing and Measurements (SCM)*, 2021, pp. 237–239. doi:10.1109/SCM52931.2021.9507120
5. **Bhatia J., Patel T., Trivedi H., Majmudar V.** HTV dynamic load balancing algorithm for virtual machine instances in cloud. *International Symp. on Cloud and Services Computing*, 2012, pp. 15–20. doi:10.1109/ISCOS.2012.25
6. **Saecker M., Markl V.** Big data analytics on modern hardware architectures: A technology survey. *Business Intelligence*, 2013, no. 138, pp. 125–149. doi:10.1007/978-3-642-36318-4_6
7. **Yannuzzi M., Milito R., Serral-Graci R., Montero D., Nemirovsky M.** Key ingredients in an IoT recipe: fog computing, cloud computing, and more fog computing. *2014 IEEE 19th Intern. Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2014, pp. 325–329. doi:10.1109/CAMAD.2014.7033259
8. **Hameed A. R., Islam S., Ahmad I., Munir K.** Energy- and performance-aware load-balancing in vehicular fog computing. *Sustainable Computing: Informatics and Systems*, 2021, no. 30(100454). doi:10.1016/j.suscom.2020.100454
9. **Osipov V.** Synthesis of effective programs for managing information and computing resources. *Devices and Control Systems*, 1998, no. 12, pp. 24–27.
10. **Jaykrushna A., Patel P., Trivedi H., Bhatia J.** Linear regression assisted prediction-based load balancer for cloud computing. *2018 IEEE Punecon*, 2019, pp. 1–3. doi:10.1109/PUNECON.2018.8745409
11. **Bhatia J., Dave R., Bhayani H., Tanwar S., Nayyar A.** SDN-based real-time urban traffic analysis in VANET environment. *Computer Communications*, 2020, no. 149, pp. 162–175. doi:10.1016/j.comcom.2019.10.011
12. **Mao Y., You C., Zhang J., Huang K., Letaief K. B.** A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials*, 2017, no. 99(1-1), pp. 37–51. doi:10.1109/COMST.2017.2745201
13. **Liu Y., Fieldsend J. E., Min G.** A framework of fog computing: architecture, challenges, and optimization. *IEEE Access*, 2017, no. 5, pp. 25445–25454. doi:10.1109/ACCESS.2017.2766923
14. **Matrouk K., Alatoun K.** Scheduling algorithms in fog computing: A survey. *International Journal of Networked and Distributed Computing*, 2021, no. 9(1), pp. 59–74. doi:10.2991/ijndc.k.210111.001
15. **Rahm E., Do H. H.** Data cleaning: Problems and current approaches. *IEEE Data Eng.*, 2000, no. 23.
16. **Huang C., Lu R., Choo K.** Vehicular fog computing: architecture, use case, and security and forensic challenges. *IEEE Communications Magazine*, 2017, no.55(11),pp.105–111.doi:10.1109/MCOM.2017.1700322
17. *Multimedia Big Data Computing for IoT Applications: Concepts, Paradigms and Solutions*/S. Tanwar, S. Tyagi, N. Kumar (Eds.). 2019, vol. 163. 477 p. doi:10.1007/978-981-13-8759-3
18. **Gorlatova M., Inaltekin H., Chiang M.** Characterizing task completion latencies in multi-point multi-quality fog computing systems. *Computer Networks*, 2020, no. 181(107526). doi:10.1016/j.comnet.2020.107526
19. **Aburukba R. O., Landolsi T., Omer D.** A heuristic scheduling approach for fog-cloud computing environment with stationary IoT devices. *Journal of Network and Computer Applications*, 2021, no. 180(102994). doi:10.1016/j.jnca.2021.102994
20. **Kimovski D., Math R.** Cloud, fog or edge: Where to compute? *IEEE Internet Computing*, 2021, no. 25(4), pp. 30–36. doi:10.1109/MIC.2021.3050613
21. **Zhang C.** Design and application of fog computing and Internet of Things service platform for smart city.

Future Generation Computer Systems, 2020, no. 112, pp. 630–640. doi:10.1016/j.future.2020.06.016

22. **Etemadi M., Ghobaei-Arani M., Shahidinejad A.** Resource provisioning for IoT services in the fog computing environment: An autonomic approach. *Computer Communications*, 2020, no. 161, pp. 109–131. doi:10.1016/j.comcom.2020.07.028
23. **Zahmatkesh H., Al-Turjman F.** Fog computing for sustainable smart cities in the IoT era: Caching techniques and enabling technologies — an overview. *Sustainable Cities and Society*, 2020, no. 59(102139). doi:10.1016/j.scs.2020.102139
24. **Bellendorf J., Mann Z. A.** Classification of optimization problems in fog computing. *Future Generation Computer Systems*, 2020, no. 107, pp. 158–176. doi:10.1016/j.future.2020.01.036
25. **Chen X., Zhou Y., Yang L., Lv L.** Hybrid fog/cloud computing resource allocation: Joint consideration of limited communication resources and user credibility. *Computer Communications*, 2021, no. 169, pp. 48–58. doi:10.1016/j.comcom.2021.01.026
26. **Kaur M., Aron R.** Energy-aware load balancing in fog cloud computing. *Materials today: Proceedings*, *2020. Intern. Conf. on Modern Trends in Manufacturing Technologies and Equipment 2020, ICMTMTE 2020*. Elsevier Ltd, 2021. doi:10.1016/j.matpr.2020.11.121
27. **Wang F., Wang J., Yang W.** Efficient incremental authentication for the updated data in fog computing. *Future Generation Computer Systems*, 2021, no. 114, pp. 130–137. doi:10.1016/j.future.2020.07.039
28. **Ogundoyin S. O., Kamil I. A.** A Fuzzy-AHP based prioritization of trust criteria in fog computing services. *Applied Soft Computing*, 2020, no. 97(106789). doi:10.1016/j.asoc.2020.106789
29. **Osipov V. U.** *Information Security: Synthesis of Control Programs*. Petrodvorets, VMIRE Publ., 2001. 64 p.
30. **Tianxing M., Osipov V., Vodyaho A., Lebedev S., Zhukova N.** Distributed technical object model synthesis based on monitoring data. *International Journal of Knowledge and Systems Science*, 2019, no. 10, pp. 27–43. doi:10.4018/IJKSS.2019070103
31. **Karagiannis V., Schulte S.** Distributed algorithms based on proximity for self-organizing fog computing systems. *Pervasive and Mobile Computing*, 2021, no. 71(101316). doi:10.1016/j.pmcj.2020.101316
32. **Dou W., Tang W., Liu B., Xu X., Ni Q.** Blockchain-based mobility-aware offloading mechanism for fog computing services. *Computer Communications*, 2020, no. 164, pp. 261–273. doi:10.1016/j.comcom.2020.10.007
33. **Zeeshan A., Shehzad A. C., Khalid M., Sahil G., Zhihan L., Yousaf B. Z.** A clogging resistant secure authentication scheme for fog computing services. *Computer Networks*, 2021, no. 185(107731). doi:10.1016/j.comnet.2020.107731
34. **Bhatia J., Modi Y., Tanwar S., Bhavsar M.** Software defined vehicular networks: A comprehensive review. *International Journal of Communication Systems (IJCS)*, 2019, no. 32(12). doi:10.1002/dac.4005
35. **Dastjerdi A. V., Gupta H., Calheiros R. N., Ghosh S. K., Buyya R.** Fog computing: Principles, architectures, and applications. *Cornell University*, 2016. doi:10.48550/arXiv.1601.02752
36. **Бутырский Е. Ю., Водяхо А. И., Жукова Н. А., Субботин А. Н.** Облачные системы интеллектуального видеонаблюдения. Логические модели и модель сбора данных. *Информация и космос*, 2022, № 4, с. 74–81. EDN: YEMPWW
37. **Бутырский Е. Ю., Водяхо А. И., Жукова Н. А., Субботин А. Н.** Облачные системы интеллектуального видеонаблюдения. Программное обеспечение. *Информация и космос*, 2023, № 1, с. 45–55. EDN: EXBSEI

UDC 004.032.26
doi:10.31799/1684-8853-2024-6-15-26
EDN: HNIROD

Dynamic distribution algorithm for image processing in cloud-based intelligent video surveillance systems

N. A. Zhukova[a], Dr. Sc., Tech., Professor, orcid.org/0000-0001-5877-4461, nazhukova@mail.ru
A. N. Subbotin[b], Post-Graduate Student, orcid.org/0000-0002-4823-6288
[a]St. Petersburg Federal Research Center of the RAS, 39, 14th Line, 199178, Saint-Petersburg, Russian Federation
[b]Saint-Petersburg Electrotechnical University «LETI», 5, Prof. Popov St., 197376, Saint-Petersburg, Russian Federation

Introduction: The presence of powerful servers in the cloud environment allows cloud video surveillance systems to perform complex image processing using machine learning and neural network methods, in addition, it becomes possible to build such processes dynamically. **Purpose:** To develop an algorithm for distributed image processing for an intelligent video surveillance system in a cloud environment. **Results:** We propose a new algorithm for data processing in video surveillance systems, providing a synthesis of complex processes and the widespread use of machine learning and neural network methods. For distributed image processing in the cloud in accordance with the synthesized processes, we use a mathematical model based on scheduling theory. We develop a logical model that determines the methods for transmitting images to the cloud, as well as the organization of virtual machines used in the cloud. Taken together, the obtained results provide new opportunities for the dynamic synthesis of processes in data processing and for the dynamic distribution of their execution across different types of virtual machines located on different physical servers. **Practical relevance:** The practical application of the obtained results in solving problems of intelligent video surveillance in the subway makes it possible to increase the speed of data processing (reduce time) by more than 2.5 times and increase the accuracy of event detection by more than 11%.

Keywords — distributed image processing, process synthesis, cloud environment, logical model.

For citation: Zhukova N. A., Subbotin A. N. Dynamic distribution algorithm for image processing in cloud-based intelligent video surveillance systems. *Informatsionno-upravliaiushchie sistemy* [Information and Control Systems], 2024, no. 6, pp. 15–26 (In Russian). doi:10.31799/1684-8853-2024-6-15-26, EDN: HNIROD

Financial support
This work was supported by the state budget, project No. FFZF-2022-0006.

Reference

1. Sarkar S., Chatterjee S., Misra S. Assessment of the suitability of fog computing in the context of internet of things. *IEEE Transactions on Cloud Computing*, 2018, no. 6(1), pp. 46–59. doi:10.1109/TCC.2015.2485206
2. Subbotin A. N., Zhukova N. A., Man T. Architecture of the intelligent video surveillance systems for fog environments based on embedded computers, *2021 10th Mediterranean Conf. on Embedded Computing (MECO)*, 2021, pp. 1–8. doi:10.1109/MECO52532.2021.9460270
3. Subbotin A. Data processing in foggy computing environments for machine learning. *2021 II Intern. Conf. on Neural Networks and Neurotechnologies (NeuroNT)*, 2021, pp. 51–53. doi:10.1109/NeuroNT53022.2021.9472203
4. Subbotin A. Applying machine learning in fog computing environments for panoramic teeth imaging. *2021 XXIV Intern. Conf. on Soft Computing and Measurements (SCM)*, 2021, pp. 237–239. doi:10.1109/SCM52931.2021.9507120
5. Bhatia J., Patel T., Trivedi H., Majmudar V. HTV dynamic load balancing algorithm for virtual machine instances in cloud. *International Symp. on Cloud and Services Computing*, 2012, pp. 15–20. doi:10.1109/ISCOS.2012.25
6. Saecker M., Markl V. Big data analytics on modern hardware architectures: A technology survey. *Business Intelligence*, 2013, no. 138, pp. 125–149. doi:10.1007/978-3-642-36318-4_6
7. Yannuzzi M., Milito R., Serral-Graci R., Montero D., Nemirovsky M. Key ingredients in an IoT recipe: fog computing, cloud computing, and more fog computing. *2014 IEEE 19th Intern. Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2014, pp. 325–329. doi:10.1109/CAMAD.2014.7033259
8. Hameed A. R., Islam S., Ahmad I., Munir K. Energy- and performance-aware load-balancing in vehicular fog computing. *Sustainable Computing: Informatics and Systems*, 2021, no. 30(100454). doi:10.1016/j.suscom.2020.100454
9. Osipov V. Synthesis of effective programs for managing information and computing resources. *Devices and Control Systems*, 1998, no. 12, pp. 24–27.
10. Jaykrushna A., Patel P., Trivedi H., Bhatia J. Linear regression assisted prediction-based load balancer for cloud computing. *2018 IEEE Punecon*, 2019, pp. 1–3. doi:10.1109/PUNECON.2018.8745409
11. Bhatia J., Dave R., Bhayani H., Tanwar S., Nayyar A. SDN-based real-time urban traffic analysis in VANET environment. *Computer Communications*, 2020, no. 149, pp. 162–175. doi:10.1016/j.comcom.2019.10.011
12. Mao Y., You C., Zhang J., Huang K., Letaief K. B. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials*, 2017, no. 99(1-1), pp. 37–51. doi:10.1109/COMST.2017.2745201
13. Liu Y., Fieldsend J. E., Min G. A framework of fog computing: architecture, challenges, and optimization. *IEEE Access*, 2017, no. 5, pp. 25445–25454. doi:10.1109/ACCESS.2017.2766923
14. Matrouk K., Alatoun K. Scheduling algorithms in fog computing: A survey. *International Journal of Networked and Distributed Computing*, 2021, no. 9(1), pp. 59–74. doi:10.2991/ijndc.k.210111.001
15. Rahm E., Do H. H. Data cleaning: Problems and current approaches. *IEEE Data Eng.*, 2000, no. 23.
16. Huang C., Lu R., Choo K. Vehicular fog computing: architecture, use case, and security and forensic challenges. *IEEE Communications Magazine*, 2017, no. 55(11), pp. 105–111. doi:10.1109/MCOM.2017.1700322
17. *Multimedia Big Data Computing for IoT Applications: Concepts, Paradigms and Solutions.* Tanwar S., Tyagi S., Kumar N. (Eds.). 2019, vol. 163. 477 p. doi:10.1007/978-981-13-8759-3
18. Gorlatova M., Inaltekin H., Chiang M. Characterizing task completion latencies in multi-point multi-quality fog computing systems. *Computer Networks*, 2020, no. 181(107526). doi:10.1016/j.comnet.2020.107526
19. Aburukba R. O., Landolsi T., Omer D. A heuristic scheduling approach for fog-cloud computing environment with stationary IoT devices. *Journal of Network and Computer Applications*, 2021, no. 180(102994). doi:10.1016/j.jnca.2021.102994
20. Kimovski D., Math R. Cloud, fog or edge: Where to compute? *IEEE Internet Computing*, 2021, no. 25(4), pp. 30–36. doi:10.1109/MIC.2021.3050613
21. Zhang C. Design and application of fog computing and Internet of Things service platform for smart city. *Future Generation Computer Systems*, 2020, no. 112, pp. 630–640. doi:10.1016/j.future.2020.06.016
22. Etemadi M., Ghobaei-Arani M., Shahidinejad A. Resource provisioning for IoT services in the fog computing environment: An autonomic approach. *Computer Communications*, 2020, no. 161, pp. 109–131. doi:10.1016/j.comcom.2020.07.028
23. Zahmatkesh H., Al-Turjman F. Fog computing for sustainable smart cities in the IoT era: Caching techniques and enabling technologies – an overview. *Sustainable Cities and Society*, 2020, no. 59(102139). doi:10.1016/j.scs.2020.102139
24. Bellendorf J., Mann Z. A. Classification of optimization problems in fog computing. *Future Generation Computer Systems*, 2020, no. 107, pp. 158–176. doi:10.1016/j.future.2020.01.036
25. Chen X., Zhou Y., Yang L., Lv L. Hybrid fog/cloud computing resource allocation: Joint consideration of limited communication resources and user credibility. *Computer Communications*, 2021, no. 169, pp. 48–58. doi:10.1016/j.comcom.2021.01.026
26. Kaur M., Aron R. Energy-aware load balancing in fog cloud computing. *Materials today: Proceedings*, *2020. Intern. Conf. on Modern Trends in Manufacturing Technologies and Equipment 2020, ICMTMTE 2020*. Elsevier Ltd, 2021. doi:10.1016/j.matpr.2020.11.121
27. Wang F., Wang J., Yang W. Efficient incremental authentication for the updated data in fog computing. *Future Generation Computer Systems*, 2021, no. 114, pp. 130–137. doi:10.1016/j.future.2020.07.039
28. Ogundoyin S. O., Kamil I. A. A Fuzzy-AHP based prioritization of trust criteria in fog computing services. *Applied Soft Computing*, 2020, no. 97(106789). doi:10.1016/j.asoc.2020.106789
29. Osipov V. U. *Information Security: Synthesis of Control Programs*. Petrodvorets, VMIRE Publ., 2001. 64 p.
30. Tianxing M., Osipov V., Vodyaho A., Lebedev S., Zhukova N. Distributed technical object model synthesis based on monitoring data. *International Journal of Knowledge and Systems Science*, 2019, no. 10, pp. 27–43. doi:10.4018/IJKSS.2019070103
31. Karagiannis V., Schulte S. Distributed algorithms based on proximity for self-organizing fog computing systems. *Pervasive and Mobile Computing*, 2021, no. 71(101316). doi:10.1016/j.pmcj.2020.101316
32. Dou W., Tang W., Liu B., Xu X., Ni Q. Blockchain-based mobility-aware offloading mechanism for fog computing services. *Computer Communications*, 2020, no. 164, pp. 261–273. doi:10.1016/j.comcom.2020.10.007
33. Zeeshan A., Shehzad A. C., Khalid M., Sahil G., Zhihan L., Yousaf B. Z. A clogging resistant secure authentication scheme for fog computing services. *Computer Networks*, 2021, no. 185(107731). doi:10.1016/j.comnet.2020.107731
34. Bhatia J., Modi Y., Tanwar S., Bhavsar M. Software defined vehicular networks: A comprehensive review. *International Journal of Communication Systems (IJCS)*, 2019, no. 32(12). doi:10.1002/dac.4005
35. Dastjerdi A. V., Gupta H., Calheiros R. N., Ghosh S. K., Buyya R. Fog computing: Principles, architectures, and applications. *Cornell University*, 2016. doi:10.48550/arXiv.1601.02752
36. Butyrsky E. U., Vodyakho A. I., Zhukova N. Z., Subbotin A. N. Cloud systems of intelligent video surveillance. Logical models and data collection model. *Information and Space*, 2022, no. 4, pp. 74–81 (In Russian). EDN: YEMPWW
37. Butyrsky E. U., Vodyakho A. I., Zhukova N. Z., Subbotin A. N. Cloud systems and intelligent video surveillance. Software. *Information and Space*, 2023, no. 1, pp. 45–55 (In Russian). EDN: EXBSEI