УДК 004.9 DOI: 10.18698/0236-3933-2018-2-98-123

МЕТОДЫ СИТУАЦИОННОГО АНАЛИЗА И ГРАФИЧЕСКОЙ ВИЗУАЛИЗАЦИИ ПОТОКОВ БОЛЬШИХ ДАННЫХ

А.В. Пролетарский pav@bmstu.ru
Д.В. Березкин berezkind@bmstu.ru
Ю.Е. Гапанюк gapyu@bmstu.ru
И.А. Козлов kozlovilya89@yandex.ru
А.Ю. Попов alexpopov@bmstu.ru
Р.С. Самарев samarev@acm.org
В.И. Терехов terekchow@bmstu.ru

МГТУ им. Н.Э. Баумана, Москва, Российская Федерация

Аннотация

Предложен подход к созданию информационной системы, осуществляющей обработку и глубокий анализ потоков разнородных данных. Проведен анализ существующих средств потоковой обработки данных. С помощью извлечения из потоков сообщений необходимой информации выполнены мониторинг и прогнозирование развития ситуаций для поддержки принятия управленческих решений. Подход основан на последовательном обнаружении событий в текстовом потоке, формировании ситуаций и построении сценариев их дальнейшего развития. Для представления причинно-следственных и иерархических связей между событиями и ситуациями предложены сложные графовые модели, в частности метаграфовая модель. Рассмотрены вопросы создания аппаратных средств для ускорения работы с такими моделями. Для достижения высокой эффективности использования полученных результатов анализа потока данных лицом, принимающим решения, применены методы когнитивной графики, в частности метод динамического анаморфирования. Предложена структура семиотической системы обработки больших потоков данных

Ключевые слова

Семиотическая система, сложные графовые модели, метаграф, обнаружение событий, сценарный анализ, когнитивная компьютерная графика, метод динамического анаморфирования

Поступила в редакцию 11.04.2017

Введение. Характерной тенденцией, влияющей на архитектуру современных информационных систем (ИС), является бурное развитие глобальной сети Интернет, где информация содержится, как правило, в неструктурированном виде. Кроме того, существует большое число документальных систем, в которых информация традиционно представлена в слабоструктурированном виде. С развитием сети Интернет, а также других средств связи появляется возможность эффективной удаленной работы с такими системами. Отмеченные особенности определяют требования к архитектуре современной ИС, которая должна быть

способной манипулировать информацией в текстовой форме, причем эти тексты могут иметь разные наборы знаков и объединять в себе разные модели данных и знаний. Эти требования приводят к необходимости создания динамических интеллектуальных ИС или систем семиотического типа.

В последние годы существенно изменился формат публикации информации в ИС. Сейчас это непрерывный поток информационных сообщений из разных источников, который необходимо обрабатывать как можно ближе к моменту публикации. В то же время изменилось понимание того, как должны строиться системы обработки таких данных. Если ранее применялись принципы пакетной обработки данных и такие инструменты, как Apache Hadoop, Apache Spark, то сейчас они безнадежно устарели [1]. На первый план выходят средства потоковой обработки данных. Однако в настоящее время нет устоявшихся принципов построения систем потоковой обработки больших данных, ориентированных на глубокий анализ содержимого потока структурированной и неструктурированной информации. Поэтому одной из задач является анализ применимости разных инструментальных средств работы с потоками больших данных (потоковых фреймворков) для создания современных ИС [2].

На основе потока разнородной информации требуется принимать различные управленческие решения. Для принятия наилучшего из возможных решений необходим экспертный анализ текстовых сообщений, получаемых с вебсайтов и из специализированных источников, что затруднительно ввиду огромного числа анализируемых документов. В настоящей работе предлагается решение этой проблемы путем создания семиотической системы, выполняющей автоматический мониторинг информации и обнаруживающей в текстовом потоке события, относящиеся к объектам и явлениям, интересующим пользователя. На основе обнаруженных событий отслеживается развитие ситуаций с течением времени и определяются возможные сценарии их дальнейшего развития в будущем, а также формируются предложения для лиц, принимающих решения (ЛПР) по действиям, необходимым для способствования или препятствования развитию ситуации по полученным сценариям.

Для проведения глубокого анализа событий и связей на основе потока информационных сообщений необходимо обеспечить хранение и обработку сложных графовых моделей данных. Поскольку мы имеем дело с большим объемом информации, необходимо использовать технологии анализа больших графов и обеспечить эффективную обработку таких данных путем правильного выбора архитектурных, алгоритмических и аппаратных решений. Эти вопросы будут рассмотрены далее.

Одним из способов быстрого анализа большого объема информации и принятия обоснованного варианта решения является визуальное представление ЛПР всей необходимой информации, учитывающей разные факторы и характеристики исследуемого процесса или явления. Эта информация должна носить в основном визуальный характер — в виде графического представления с тем, чтобы ее восприятие и обработка привели к резкому сокращению времени анализа и, соответ-

ственно, времени принятия обоснованного варианта решения. В настоящей работе исследуются методы когнитивной графики, которые необходимо применять для принятия решений по результатам анализа потоков больших данных.

Анализ современных инструментальных средств обработки потоков больших данных. Среди наиболее известных средств работы с потоками больших данных можно отметить следующие:

Apache Flink (https://flink.apache.org/);

Apache Kafka Streams (https://kafka.apache.org/);

Apache Samza (https://samza.apache.org/);

Heron (https://twitter.github.io/heron/);

Apache Spark Streaming (http://spark.apache.org/streaming/).

С использованием этих средств необходимо собрать, очистить, предварительно обработать и структурировать информацию для дальнейшего ситуационного анализа. Специфика решаемых задач определяет выбор потокового фреймворка (или процессора в данном контексте) и архитектуру системы обработки данных.

История потоковых процессоров начинается с конца 1980-х годов как попытка создания систем управления базами данных (СУБД) нового поколения, способных хранить данные как поток. За прошедшие годы произошло несколько этапов переосмысления принципов построения потоковых процессоров [3]. Можно отметить, что ни один из программных продуктов 1990-х начала 2000-х годов не дожил до нашего времени. Однако сохранилась идея обработки данных без потерь и с минимальной задержкой, которая реализована в очередном поколении потоковых процессоров.

Современный этап развития потоковых процессоров совпал с очередным всплеском интереса программистов к функциональному стилю программирования и соответствующими изменениями основных языков программирования. При этом четко просматриваются разные ветви развития потоковых процессоров, созданные за последние 10 лет. Среди них можно отметить Apache Storm и Apache Samza, которые требуют формирования схемы обработки как связи источника данных, операций трансформации данных и стока данных. Недостаток Apache Storm — необходимость учитывать множество деталей процесса обработки данных, что сильно затрудняет эффективное создание программных продуктов. Следующая ветвь — это потоковые процессоры, в которых процесс преобразования данных описывается декларативно в терминах функций языка программирования в функциональном стиле [2]. К этой ветви относятся потоковые процессоры Apache Spark, Apache Flink и Apache Kafka Streams. Основной проблемой выбора потокового процессора, однако, является не принцип программирования, а особенности его внутренней организации. Среди основных требований к потоковому процессору следует отнести гарантированную обработку данных, минимальное время задержки обработки, способность динамически балансировать нагрузку на вычислительном кластере. Проведенные оценки производительности [4] показывают, что продукты, ориентированные на пакетный принцип обработки, такие как Apache Spark, оказываются мало пригодными к реальному использованию в пото-

ковой обработке из-за большой задержки обработки данных и большой чувствительности к настройкам ограничений входного потока.

Еще одна ветвь потоковых процессоров — легковесные потоковые процессоры типа Apache Kafka Streaming и, в определенной степени, Java Reactor, которые хотя и описывают процесс обработки данных в функциональном стиле и имеют средства создания автономных приложений, но не позволяют выполнять переупорядочивание данных между узлами кластера, поскольку не предоставляют никаких средств для централизованного управления данными. Каждое такое приложение работает независимо от других. Распределение данных и их сбор могут быть реализованы с помощью очереди сообщений Apache Kafka.

Следует отметить разную степень надежности работы перечисленных программных продуктов, причем выявить это можно лишь при решении конкретных задач. Поэтому отдельной актуальной задачей в настоящее время является создание средства для тестирования разных потоковых процессоров.

Тем не менее наиболее распространенный способ построения систем потоковой обработки данных приведен в [1]. Это можно проиллюстрировать схемой (рис. 1).

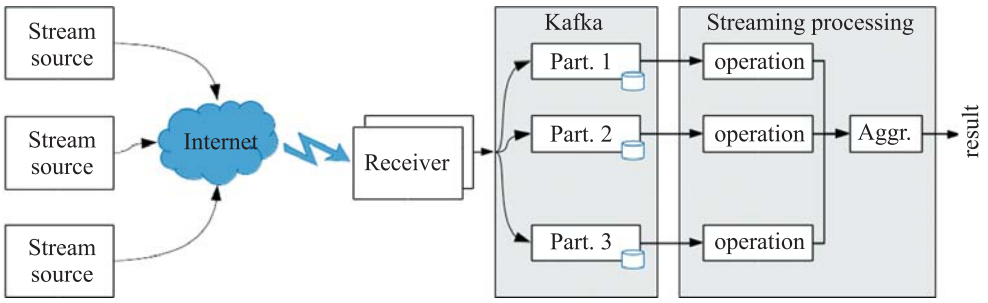


Рис. 1. Способ построения систем потоковой обработки данных

Приведенная схема универсальная и обеспечивает высокоскоростной сбор первичных данных и их агрегацию, позволяя проводить первичный анализ, снижающий объем данных и облегчающий применение алгоритмов машинного обучения на последующих этапах. Отметим, что в ряде случаев возможно использовать эти же средства для решения задач машинного обучения, однако в силу указанных причин необходимо предварительно оценить как можно более полный набор алгоритмов, которые будут необходимы, и провести реальный замер производительности этих средств перед окончательным принятием решения.

Таким образом, анализ современных потоковых процессоров показывает, что текущее состояние области не позволяет принять окончательное решение о выборе потокового процессора без выполнения прототипирования, поскольку поведение разных потоковых процессоров сильно различается в зависимости от конкретных решаемых задач.

Принципы хранения и обработки графовых моделей. Использование методов на основе обработки потоков больших данных предъявляет дополнитель-

ные требования к модели хранения данных. Классический сервис-ориентированный подход, при котором каждый сервис может использовать собственную базу данных (при этом базы данных могут даже использовать разные модели данных), оказывается недостаточным при построении системы ситуационного анализа. Это обусловлено тем, что система ситуационного анализа предполагает постепенное наращивание ситуации в процессе обработки данных и используемая модель данных должна поддерживать такое наращивание.

В наиболее полном исследовании по обработке больших данных [5] отмечается, что именно графовая модель наиболее подходит для обработки больших данных, и даны рекомендации по построению такой модели. Несмотря на то что книга [5] в основном ориентирована на пакетную обработку данных с использованием лямбда-архитектуры, рекомендация по использованию графовой модели данных не теряет своей актуальности и в случае использования потоковой обработки. При этом в случае потоковой обработки по сравнению с пакетной обработкой возникают дополнительные требования к скорости выполнения операций над графами (в силу увеличения обрабатываемого объема исходных данных), что приводит к идее аппаратного ускорения выполнения операций над графами.

В рамках графового подхода можно выделить три технологических направления.

1. *Использование стандартов семантического веба и хранилищ на их основе.* В этом случае в качестве модели данных используется RDF. В качестве языка запросов для RDF используется SPARQL. На основе RDF созданы стандарты описания онтологий, такие как RDFS и OWL. Для работы с RDF используются специализированные СУБД (RDF-хранилища), наиболее известными из которых является Apache Jena.

Технологии на основе семантического веба доведены до уровня промышленных и используются во многих ИС. Однако практика показала, что деление RDF-модели на триплеты оказалось слишком «мелким», в результате этого реляционной базе данных «среднего» размера может соответствовать хранилище, содержащее миллиарды триплетов. Это приводит к существенным вычислительным затратам, а также к сложности в содержательной интерпретации такой модели и ее сопровождению.

Другой серьезной проблемой является проблема описания N-арных отношений [6]. Модель RDF не позволяет простыми способами описывать N-арные отношения между вершинами графа, что усложняет описание комплексных ситуаций и затрудняет использование RDF в качестве основы для системы ситуационного анализа.

2. *Использование СУБД на основе разновидностей классической графовой модели.* Такие СУБД используют в качестве модели данных граф или мультиграф. В настоящее время для обозначения модели СУБД часто используют термин «property graph», т. е. граф, узлам которого соответствуют свойства, а направленным ребрам — отношения между ними.

В качестве наиболее известного примера такой СУБД можно назвать Neo4j. Она использует модель property graph (в качестве расширения поддерживаются гиперграфы) и снабжена мощным языком графовых запросов. В настоящее время графовые СУБД активно адаптируются для обработки больших данных, в частности, существует адаптер для использования Neo4j совместно с Apache Spark.

Однако плоские графы также не очень хорошо подходят для описания комплексных ситуаций. Поэтому появляется новая тенденция использования СУБД на основе сложных графовых моделей.

3. *Использование СУБД на основе сложных графовых моделей.* В качестве примера такой СУБД можно назвать HypergraphDB. Как следует из названия, HypergraphDB использует в качестве модели данных модель гиперграфа. Однако классическая гиперграфовая модель дает ограниченные преимущества по сравнению с плоской графовой моделью и также не позволяет в полной мере описывать комплексные ситуации. В соответствии с [7] для описания комплексных ситуаций в большей степени подходят более сложные графовые модели, такие как гиперсети и метаграфы. В работе [7] отмечается, что метаграфы и гиперсети являются лишь разными формальными описаниями одних и тех же процессов, которые происходят в «сетях с эмерджентностью», но при этом модель метаграфа является более простой при описании и более гибкой.

Поэтому именно метаграфовую модель [8, 9] предлагается использовать для описания комплексных ситуаций. Для обработки данных в метаграфовой модели предлагается использовать мультиагентный подход на основе продукционных правил.

Необходимо отметить, что в настоящее время существует большое количество реализованных программных решений для хранения и обработки плоских графов, в том числе больших. Аналогичные средства для сложных графовых моделей еще предстоит создать. Поэтому для описания ситуаций предполагается использовать гибридную графовую модель [10]. В случае простых ситуаций предполагается использовать плоские графы, а в случае комплексных ситуаций метаграфы.

С точки зрения больших данных существует два основных подхода к обработке графовых моделей:

– использование графовых СУБД и выполнение операций над графами в режиме постоянного хранения без изменения исходных графов; примерами таких систем являются Titan, Neo4j, Gradoop;

– итеративная обработка графов с их модификацией в процессе обработки; примерами таких систем являются Apache Flink Gelly, Apache Giraph.

Работа как с плоскими, так и со сложными графовыми моделями требует больших вычислительных затрат. Для решения этой проблемы предлагается использовать методы аппаратного ускорения выполнения операций над графами.

Перспективы аппаратного ускорения операций над графами. Современные средства вычислительной техники, построенные на универсальных микропроцессорах, имеют ряд принципиальных технических ограничений, снижаю-

щих скорость обработки графовых моделей [11]. Глубокая конвейеризация, применяемая в подавляющем большинстве микропроцессоров, опирается на механизм ранней аппаратной предвыборки данных и вызывает существенные задержки при обходе связанных вершин графов. Механизм виртуализации памяти предполагает преобразование логических адресов с помощью таблиц быстрого страничного преобразования (Translation Lookaside Buffer, TLB) микропроцессора [12]. Однако при выборке данных, расположенных в оперативной памяти на существенном адресном расстоянии, такой буфер оказывается неэффективным. Свой вклад в общие задержки вносят и динамические запоминающие устройства (ЗУ), применяемые для реализации современной оперативной памяти: выборка информации о вершинах графа вызывает постоянные открытия и закрытия страниц, что требует существенно большего количества тактов. Указанные недостатки приводят к тому, что скорость обработки графовых моделей оказывается недостаточной, несмотря на высокие тактовые частоты.

Попытки применить специальные аппаратные средства для ускорения обработки графов предпринимались неоднократно [13, 14]. Аппаратные ускорители операций над графами можно разделить на три типа.

Естественное представление вершин графа в виде сети аппаратных узлов (нодов) [13, 14]. Такой подход предполагает реализацию графа в виде аппаратной структуры взаимосвязанных аппаратных узлов, где каждый блок выполняет только несколько примитивных действий (например, вычисление длины пути, передача сигналов связанным нодам и пр.). Однако такой подход существенно ограничен возможностями аппаратной технологии и не позволяет обрабатывать произвольные графы большой размерности.

Ускорители обработки зависимых данных [15, 16]. Такие устройства сокращают временные затраты на доступ к зависимым данным. Например, известны ускорители для интеллектуального кэширования данных. Однако такие ускорители не увеличивают степень параллельности в системе, а лишь сокращают латентность доступа к памяти. Принципиальные проблемы конвейерной обработки с помощью подобного подхода не могут быть устранены.

Универсальные ускорители обработки множеств, структур данных и графов. Особенности ускорителей данного класса заключаются в аппаратной реализации всех механизмов обработки графов в едином устройстве. Графы и другие модели представления данных самостоятельно сохраняются и обрабатываются таким ускорителем без участия микропроцессора. С помощью специального аппаратного обеспечения достигается максимальная параллельность при обработке сложных моделей. Однако такие ускорители сложны в проектировании и требуют применения развитых технических решений (например, программируемой логической интегральной схемы (ПЛИС) большой размерности).

Примером реализации ускорителя третьего класса является вычислительная система со многими потоками команд и одним потоком данных (МКОД), разработанная в МГТУ им. Н.Э. Баумана [17]. В проекте выполнен полный цикл создания принципиально новой универсальной вычислительной системы, начиная от со-

здания принципов и моделей и заканчивая созданием опытного образца, специальных средств программирования, написанием тестов и испытаний. Новая система позволила сократить время обработки больших объемов данных за счет аппаратного ускорения действий над множествами и структурами данных. Разработано принципиально новое вычислительное устройство — процессор обработки структур (Structure processor unit, SPU) (СП), реализующий набор команд высокого уровня над множествами и структурами данных, его архитектура позволяет более эффективно решать задачи дискретной оптимизации, основанные на моделях множеств, графов и отношений.

Вычислительная система МКОД состоит из двух процессорных устройств: центрального процессора (Central processor unit, CPU) (ЦП) и СП. Устройства обмениваются командной информацией и данными через специальные очереди (рис. 2). В МКОД-системе ЦП — это универсальное процессорное устройство, хранящее в своем ОЗУ команды и данные (CPU Main Memory). Процессор обработки структур выполняет инициализацию системы, запускает задачу на исполнение и обрабатывает результаты работы всего алгоритма. При этом основной вычислительный алгоритм разделяется на два потока команд: команды обработки данных, исполняемые CPU, и команды обработки структур, исполняемые SPU.

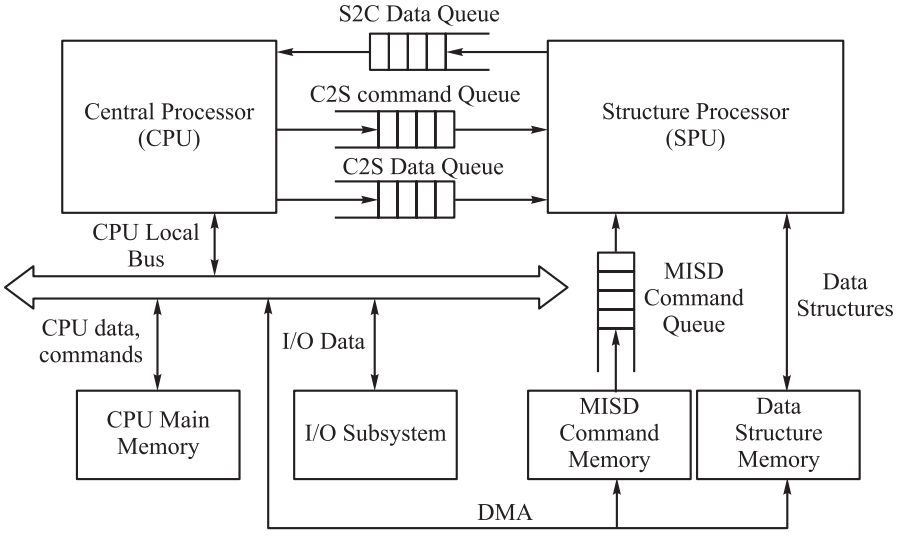


Рис. 2. Архитектура вычислительной системы с МКОД

Применительно к графовым моделям МКОД-система позволяет хранить множества вершин и ребер в локальной памяти структур SPU (Data Structure Memory), откуда их можно извлекать и обрабатывать параллельно и независимо с выполнением других задач в CPU. Вместе с тем большое количество ветвлений в алгоритмах и связь по данным двух частей алгоритма может негативно сказаться на параллельной работе СП и ЦП. Поэтому был исследован ряд алгоритмов оптимизации на сетях и графах, что позволило определить перспективные

варианты совершенствования представленного подхода и сократить зависимости между двумя вычислительными потоками.

Процессор обработки структур был спроектирован и успешно реализован на платформе FPGA фирмы Xilinx. В настоящее время SPU функционирует в составе вычислительной системы с CPU×86, к которой предоставляется удаленный доступ через сеть Интернет. По результатам проектирования выполнены тесты работоспособности SPU и исследована его производительность как на отдельных командах, так и на алгоритмах на графах. Результаты экспериментов для наглядности сведены в таблицу. Процессор SPU позволяет повысить до 164 раз эффективность операций над графами по сравнению с вычислительными системами на основе микроархитектур Microblaze, ARM11, Intel Core i. При этом аппаратная сложность SPU примерно в 450 раз меньше, чем у микропроцессора Intel Core i5, а затраченный объем электроэнергии для решения задач на МКОД-системе оказывается примерно в 31 раз меньше. Это указывает на необходимость дальнейших исследований способов применения разработанной архитектуры МКОД и SPU для анализа и обработки графовых моделей большой размерности.

Результаты экспериментов

Эксперимент	Ускорение в МКОД
Удаление вершины / ребра графа	до 164,4 раза
Добавление вершины / ребра в граф	до 42,7 раза
Поиск вершины / ребра	до 31,4 раза
Алгоритм поиска в глубину	до 12,9 раза
Алгоритм поиска в ширину	до 12,3 раза
Алгоритм Прима	до 10,3 раза
Алгоритм Дейкстры	до 7,6 раза
Алгоритм Крускала	до 7,8 раза

Одним из перспективных направлений внедрения SPU является реализация метаграфовых моделей и операций над ними. Принципы, заложенные в аппаратное обеспечение SPU, позволяют эффективно хранить элементы множеств, связанные N-арными отношениями. Предполагается, что использование разработанного аппаратного обеспечения SPU и специализированной локальной памяти для хранения метаграфовых моделей позволит сократить время их обработки.

Важным результатом применения SPU является реализация функций обработки высокого уровня с помощью аппаратного обеспечения. Система на основе SPU имеет возможность создания и вызова макрокоманд, реализованных в виде аппаратных модулей. Это позволяет выполнять функционально сложную обработку информации в наибольшей степени параллельно и независимо от основной вычислительной системы, благодаря чему достигается более высокая производительность. Программное обеспечение может вызывать такие макросредства через стандартные обращения ioctl к драйверу устройства, минуя низкий уровень представления моделей метаграфов в памяти, обхода метавершин и

прочих действий. Благодаря этому процесс разработки программного кода обработки метаграфовых моделей может быть существенно ускорен.

Методы ситуационного анализа и прогноза потоков больших данных. Обрабатываемые потоки разнородных данных имеют динамический характер и отражают развитие различных ситуаций реального мира с течением времени. Для принятия наилучших управленческих решений необходимо отслеживать развитие интересующих пользователя ситуаций, а также прогнозировать их возможное развитие в будущем.

Процесс мониторинга и прогнозирования развития ситуаций состоит из следующих этапов:

– обнаружение событий, отраженных в текстовых сообщениях;

– отслеживание развития ситуаций на основе обнаруженных событий;

– построение возможных сценариев дальнейшего развития ситуаций;

– формирование предложений для ЛПР.

Под «событием» понимается существенное изменение, произошедшее в реальном мире и отраженное в анализируемом потоке. Событие описывается многокритериальной моделью, компоненты которой отражают аспекты события, учитываемые экспертами при ручном анализе потока текстовых документов: состав слов документов и их заголовков, относящиеся к событию персоны, организации и географические наименования, временной интервал, релевантные событию темы и другие критерии [18].

В основе обнаружения событий лежит инкрементальная кластеризация. Каждое событие представлено кластером описывающих его документов. На основе сообщений из этого кластера формируются компоненты модели события. Для сопоставления документа и события выполняется попарное сравнение компонентов их моделей, для чего используются разные меры близости: косинусная мера, коэффициент Жаккара, расстояние Левенштейна. На основе такого многокритериального сравнения рассчитывается значение близости между документом и событием, для чего используется метод опорных векторов (Support Vector Machine, SVM).

Распределение документов по кластерам с помощью метода инкрементальной кластеризации состоит из следующих шагов.

1. Новый документ сравнивается с каждым ранее обнаруженным событием, в результате определяется расстояние между документом и событием.

2. Выбирается наиболее близкое к документу событие.

3. Если расстояние между сообщением и выбранным событием не превышает пороговое значение — сообщение относится к этому событию.

4. Если расстояние больше порогового значения — создается новое событие на основе этого документа.

На основе обнаруженных событий выполняется формирование ситуаций. Ситуации представляют собой цепочки взаимосвязанных событий, отражающие развитие разных процессов в реальном мире. При анализе потока тестовых сообщений из множества обнаруженных событий выделяются пары взаимосвязанных событий, потенциально принадлежащих одной ситуации. Для их выде-

ления выполняется попарное сравнение моделей событий с точки зрения всех аспектов. На основе формирования таких пар выполняется построение ситуационного графа, в котором узлы соответствуют событиям, а ребра — выделенным парам. Любой путь в этом графе является потенциальной ситуацией. Применение подхода, основанного на принципах обработки больших графов, позволяет использовать более сложные графовые модели событий и ситуаций, отражающие причинно-следственные и иерархические связи между событиями. Исследовательской задачей является построение таких моделей и применение их для ситуационного анализа и прогнозирования.

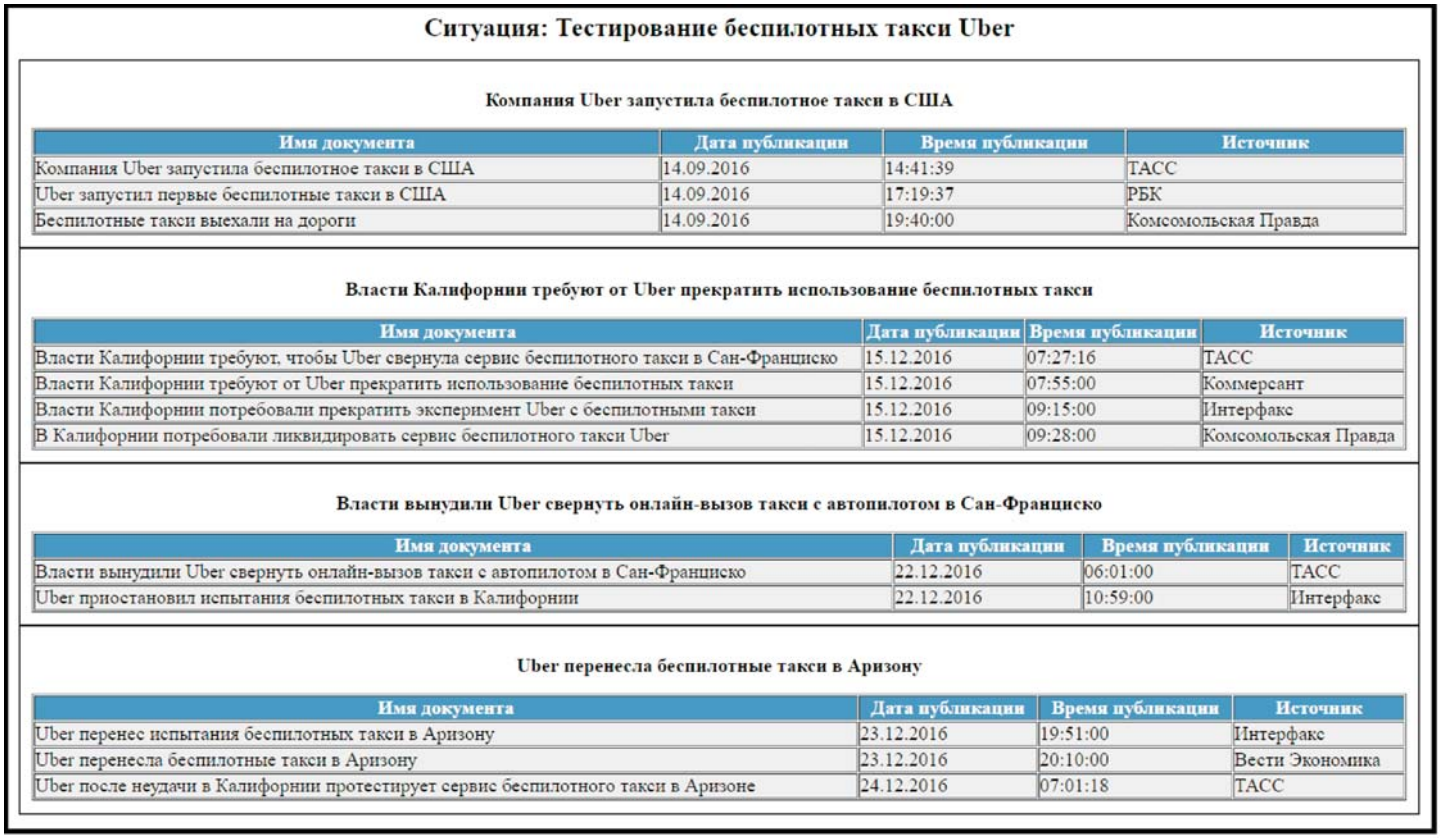
На рис. 3 показан пример формирования ситуации, связанной с тестированием беспилотных такси компанией Uber. Ситуация представляет собой цепочку из четырех взаимосвязанных событий.

Помимо отслеживания развития ситуации к настоящему моменту необходимо определять возможные сценарии ее дальнейшего развития в будущем. Формирование сценариев состоит в построении цепочек событий, являющихся потенциальными продолжениями текущей ситуации. Предлагаемый метод генерации сценариев основан на принципе исторической аналогии. Он предполагает наличие подготовленной экспертами базы эталонных ситуаций, описывающих развитие разных процессов в прошлом. Идея метода состоит в обнаружении для текущей ситуации аналогичных ей эталонных ситуаций [19]. При этом предполагается, что обнаруженные аналоги представляют собой возможные сценарии дальнейшего развития текущей ситуации.

Текущая ситуация сравнивается с каждой эталонной ситуацией на основе сопоставления отдельных событий. С помощью метода логистической регрессии оценивается вероятность того, что текущая ситуация аналогична эталонной. При этом эталонная ситуация разбивается на две части: начальную, которая соответствует текущей ситуации, и заключительную. Если вероятность аналогичности ситуаций превышает пороговое значение, заключительная часть эталонной ситуации признается возможным сценарием дальнейшего развития текущей.

Для обеспечения эффективного использования результатов прогнозирования в целях принятия решений из всех полученных сценариев необходимо выделять те, которые представляют наибольший интерес для ЛПР — оптимистический, пессимистический и наиболее вероятный. Наиболее вероятный сценарий формируется на основе эталонной ситуации, наиболее близкой к текущей. Для выделения наиболее оптимистичного и пессимистичного сценариев выполняется определение приоритетов сценариев с помощью метода анализа иерархий.

Сценарии сравниваются с точки зрения различных критериев, таких как длительность сценария, затраты на его реализацию, последствия реализации сценария. Приоритетность критериев относительно цели (определение наиболее оптимального сценария) вычисляется на основе попарных сравнений, выполняемых экспертами на этапе обучения. Приоритетность сценариев относительно каждого критерия вычисляется автоматически на основе характеристик сценариев. Таким образом, при анализе сценариев, сформированных для текущей ситуации, их прио-



Ситуация: Тестирование беспилотных такси Uber

Компания Uber запустила беспилотное такси в США

Имя документа	Дата публикации	Время публикации	Источник
Компания Uber запустила беспилотное такси в США	14.09.2016	14:41:39	ТАСС
Uber запустил первые беспилотные такси в США	14.09.2016	17:19:37	РБК
Беспилотные такси выехали на дороги	14.09.2016	19:40:00	Комсомольская Правда

Власти Калифорнии требуют от Uber прекратить использование беспилотных такси

Имя документа	Дата публикации	Время публикации	Источник
Власти Калифорнии требуют, чтобы Uber свернула сервис беспилотного такси в Сан-Франциско	15.12.2016	07:27:16	ТАСС
Власти Калифорнии требуют от Uber прекратить использование беспилотных такси	15.12.2016	07:55:00	Коммерсант
Власти Калифорнии потребовали прекратить эксперимент Uber с беспилотными такси	15.12.2016	09:15:00	Интерфакс
В Калифорнии потребовали ликвидировать сервис беспилотного такси Uber	15.12.2016	09:28:00	Комсомольская Правда

Власти вынудили Uber свернуть онлайн-вызов такси с автопилотом в Сан-Франциско

Имя документа	Дата публикации	Время публикации	Источник
Власти вынудили Uber свернуть онлайн-вызов такси с автопилотом в Сан-Франциско	22.12.2016	06:01:00	ТАСС
Uber приостановил испытания беспилотных такси в Калифорнии	22.12.2016	10:59:00	Интерфакс

Uber перенесла беспилотные такси в Аризону

Имя документа	Дата публикации	Время публикации	Источник
Uber перенес испытания беспилотных такси в Аризону	23.12.2016	19:51:00	Интерфакс
Uber перенесла беспилотные такси в Аризону	23.12.2016	20:10:00	Вести Экономика
Uber после неудачи в Калифорнии протестирует сервис беспилотного такси в Аризоне	24.12.2016	07:01:18	ТАСС

Рис. 3. Пример выявления событий и формирования ситуации

ритетность относительно цели вычисляется автоматически. Сценарий, имеющий наибольший приоритет, признается оптимистическим, сценарий с наименьшим приоритетом — пессимистическим.

При подготовке эталонных ситуаций эксперты снабжают каждое событие рекомендациями, определяющими какое лицо, какие действия и в какой срок должны быть выполнены при наступлении аналогичного события. Это позволяет формировать для каждого сценария предложения, которые могут быть использованы при принятии решений.

На рис. 4 приведены три возможных сценария дальнейшего развития вышеописанной ситуации. Каждый сценарий основан на ситуации, имевшей место в прошлом, и снабжен рекомендациями по действиям, которые необходимо предпринять с учетом возможности развития ситуации по этому сценарию.

Предложенный подход позволяет осуществлять мониторинг интересующих пользователей ситуаций, своевременно принимать меры по предотвращению кризисов и способствует развитию ситуаций по наиболее благоприятным сценариям. В частности, он позволяет обнаруживать события, связанные с деятельностью компаний-конкурентов, выявлять и отслеживать новые тенденции в науке, оценивать перспективность тех или иных разработок.

На основе предложенного подхода разработан ситуационный процессор, выполняющий автоматизированный мониторинг и прогнозирование развития ситуаций. Обучение процессора проводится экспертом на основе подготовленных эталонных событий и ситуаций. Обученный процессор выполняет автоматический анализ потока текстовых документов и предоставляет пользователю обнаруженные события, ситуации, вероятные сценарии их дальнейшего развития и предложения по действиям, которые необходимо выполнить.

Методы когнитивной графики для отображения потоков больших данных и поддержки принятия решений. Требования к точности и объему исходных данных существенно зависят как от этапа принятия решения, так и от управленческого звена, в котором оно вырабатывается. Чем выше уровень ЛПР, тем более обобщенная и усредненная информация необходима для принятия обоснованного варианта решения. При этом происходит существенный сдвиг от количественного к качественному обоснованию решения, также качественное решение значительно лучше и проще поддается визуализации, чем количественное, обеспечивая оперативность принятия решения при сохранении его достоверности.

Рассмотрим применение методов когнитивной компьютерной графики (ККГ) для отображения потоков больших данных с целью активировать мощные механизмы наглядно-образного мышления ЛПР при принятии решений в сложной обстановке или нахождении решения сложной проблемы. При этом под ККГ понимают совокупность реализованных на компьютере приемов и методов визуального представления условий задачи, которое позволяет ЛПР либо сразу увидеть ее решение, либо получить подсказку для его нахождения.

а) Оптимистический сценарий (получение разрешения на использование технологии)

Название события эталонной ситуации	Рекомендации	
Власти Британии выдали Amazon разрешение на испытания беспилотников для доставки товаров	*actor*	Руководство компании
	action	Инициировать получение специального разрешения
	period	1 месяц

б) Наиболее вероятный сценарий (запрет использования технологии до предоставления доказательств безопасности)

Название события эталонной ситуации	Рекомендации	
Шотландия ввела мораторий на технологию фрекинга	*actor*	Руководство компании
	action	Организовать подготовку обоснований безопасности технологии
	period	3 месяца

в) Пессимистический сценарий (прекращение использования технологии из-за проблем с безопасностью)

Название события эталонной ситуации	Рекомендации	
Samsung объявил о прекращении производства Galaxy Note 7	*actor*	Руководство компании
	action	Направить средства на развитие других технологий
	period	3 месяца

Рис. 4. Пример формирования оптимистического, наиболее вероятного и пессимистического сценариев развития ситуации

Для решения задачи графической визуализации потоков больших данных предлагается использовать метод динамического анаморфирования, позволяющий получать и анализировать сценарии развития ситуаций во времени [18, 19], которые строятся в метриках актуальных показателей и определяются методиками оценки ситуаций, с которыми работает ЛПР.

Сущность метода анаморфирования состоит в том, что потоки больших данных решаемой задачи представляются в виде исходных визуальных образов, построенных в евклидовой метрике. На основе выбранного показателя они преобразуются (анаморфируются) в другие визуальные образы, внутренняя структура которых изменяется так, что распределение выбранного показателя становится равномерным при сохранении топологического подобия с исходными визуальными образами [20–22]. Это позволяет проанализировать изменяющиеся в пространстве характеристики сразу нескольких явлений. Выполнить такой анализ проще, если одну или несколько изменяющихся характеристик полагать равномерно распределенными и на их фоне анализировать все остальные. Пример, иллюстрирующий сущность метода, приведен на рис. 5.

Общую постановку задачи анаморфирования можно представить следующим образом. Определим G как область на плоскости R^2 (площадная фигура, построенная на основе выбранного показателя), которая должна быть анаморфирована. При этом распределение выбранного показателя описывается функцией $p(z)$, определенной на части $G(z=(x,y)\in R^2)$.

Положим, что функция плотности $p(z)$ определена на всей R^2. Для этого можно определить $p(z)$ как постоянную вне области G (среднее значение $\overline{p}$ функции $p(z)$ по области G).

Анаморфозу можно задать отображением $h: R^2 \to R^2 (h:(x,y)\mapsto(u,v))$ или непрерывными взаимно-однозначными функциями $(U(x,y)=u$ и $V(x,y)=v$. Преобразование h изменяет площадь в окрестности точки $z=(x,y)$ с коэффициентом

$$J = J(U, V) = \frac{\partial U}{\partial x}\frac{\partial V}{\partial y} - \frac{\partial U}{\partial y}\frac{\partial V}{\partial x}. \qquad (1)$$

При этом условие $p(x,y)=\overline{p}=\text{const}$ может быть представлено в виде $J(U,V)=$ $=\dfrac{p(x,y)}{\overline{p}}$, а задача нахождения анаморфозы — это решение уравнения

$$\frac{\partial U}{\partial x}\frac{\partial V}{\partial y} - \frac{\partial U}{\partial y}\frac{\partial V}{\partial x} = \frac{p(x,y)}{\overline{p}}, \qquad (2)$$

для которого $U(x,y)$ и $V(x,y)$ определяют взаимно-однозначное преобразование [20].

После преобразования (2) каждая площадная фигура искажается в соответствии с заданной функцией, что ведет к изменению всей области G.

	1	2	3	4	5	6	7	8	9	10	11	...	49	50
1	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0		4,0	4,0
2	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0		4,0	4,0
3	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0		4,0	4,0
4	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0	4,0		4,0	4,0
5	4,0	4,0	3,0	3,0	3,0	3,0	3,0	3,0	3,0	3,0	3,0		4,0	4,0
6	4,0	3,0	3,0	3,0	3,0	3,0	3,0	3,0	3,0	3,0	3,0		3,0	4,0
7	4,0	3,0	2,0	2,0	2,0	2,0	2,0	2,0	2,0	2,0	2,0		3,0	3,0
8	4,0	3,0	2,0	1,5	1,5	1,5	1,5	1,5	1,5	1,5	1,5		2,0	2,0
9	4,0	3,0	2,0	1,5	1,5	1,5	1,5	1,5	1,5	1,5	1,5		1,5	1,5
10	4,0	3,0	2,0	1,5	1,5	1,3	1,3	1,3	1,3	1,3	1,3		1,5	1,5
11	4,0	3,0	2,0	1,5	1,5	1,3	1,3	1,3	1,3	1,3	1,3		1,3	1,3
...													1,0	1,0
49	3,0	3,0	2,0	1,5	1,5	1,3	1,3	1,0	0,5,7	0,5,7	0,5,7	0,5,7	0,5,7	1,0
50	3,0	3,0	2,0	1,5	1,5	1,3	1,3	1,0	0,5,7	0,5,5	0,5,5	0,5,5	0,5,7	1,0

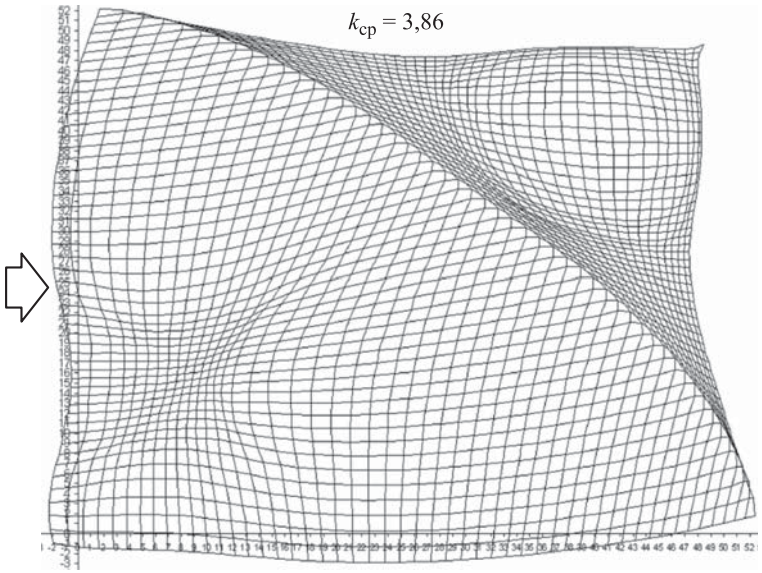


Рис. 5. Пример анаморфозы матрицы распределения показателя *k*

Трудности численного построения анаморфоз заключаются в необходимости контролировать сохранение взаимной однозначности преобразованных площадных фигур, а также в том, что существует бесконечно много преобразований, удовлетворяющих этому условию (2), так как условие выравнивания заданной плотности не определяет анаморфозу однозначно. Поэтому при построении можно использовать условие конформности преобразования (1), изменяющее все расстояния умножением на один и тот же коэффициент, не зависящий от направления и сохраняющий углы между прямыми линиями. Результатом устранения большинства из перечисленных трудностей было появление одного наиболее удачного алгоритма анаморфирования, приведенного в [20]. На основе этого алгоритма в [21] были разработаны две его модификации и способ работы этих алгоритмов с интегральным показателем анаморфирования, представляющим собой свертку N локальных показателей разной физической природы с учетом их важности и сбалансированности в решаемой задаче.

В первом варианте модифицированного алгоритма — на этапе подготовки данных используется специальным образом сконструированная логистическая функция, позволяющая избежать нарушения целостности получаемого изображения и проводить аналитический расчет анаморфозы с любой заранее заданной степенью точности ε. Во втором варианте модифицированного алгоритма наряду с логистической функцией в расчетной части алгоритма используется обученная на аналитических расчетных примерах искусственная нейронная сеть с одним скрытым слоем, позволяющая значительно сократить время его работы при допустимой потере точности вычислений [22].

Суть работы динамической анаморфозы заключается в том, что актуальные при решении поставленной задачи локальные показатели, изменяющиеся во времени, можно рассматривать как временные ряды и прогнозировать их с использованием одного из методов искусственного интеллекта — символьной регрессии. Символьная регрессия является методом построения регрессионных моделей путем перебора разных произвольных суперпозиций функций из некоторого заданного набора, а ее реализация предполагает генерацию формулы, представленной в виде синтаксического дерева, с использованием методов генетического программирования (частного случая генетического алгоритма, работающего с синтаксическими деревьями переменной длины).

Задачу прогнозирования временного ряда локальных показателей можно представить следующим образом. На вход генетического алгоритма подается набор значений исходного временного ряда локальных показателей и набор базовых функций для построения синтаксического дерева. Базовые функции представляют собой логарифмическую, экспоненциальную, тригонометрические и степенные функции. С помощью генетического алгоритма из базовых функций формируется синтаксическое дерево, которое является описанием функции в аналитическом виде. Функцией приспособленности генетического алгоритма является ошибка аппроксимации исходного временного ряда локальных показателей и ряда, сгенерированного синтаксическим деревом, при этом генетический

алгоритм пытается уменьшить эту ошибку и добиться максимальной точности аппроксимации [23].

Преимуществом предложенного подхода является возможность корректировки синтаксического дерева в процессе работы. По мере появления новых данных вызываются дополнительные шаги генетического алгоритма, в результате чего модифицируется аналитический вид функции (синтаксическое дерево) таким образом, что функция аппроксимирует как старые, так и новые данные. По мере появления новых данных уточняется поведение функции и корректируется прогноз.

Получая в каждой временной точке t значения локальных показателей N разной физической природы можно свернуть их в интегральный показатель анаморфирования k_t и построить анаморфозу A_t на его основе.

Такой подход позволяет строить динамическую анаморфозу на основе прогнозных значений локальных показателей и, следовательно, построения сценариев действий ЛПР.

Очевидными достоинствами динамической анаморфозы в задачах ситуационного анализа и графической визуализации потоков больших данных являются:

- уменьшение размерности задачи на количество локальных показателей разной физической природы, свернутых в интегральный показатель анаморфирования;
- возможность визуального моделирования и принятия решения, учитывая изменяющиеся во времени локальный или интегральный показатели разной физической природы, что позволяет определить негативные или позитивные тенденции в изменении заданных показателей анализируемого процесса или явления и предложить качественные и количественные меры по их нейтрализации или усилению до установленных значений;
- выявление скрытых закономерностей поведения во времени разных параметров решаемой задачи, неявным образом зависящих от интегрального показателя анаморфирования;
- возможность построения сценариев действий ЛПР на основе визуального пространственно-временного анализа динамической анаморфозы, учитывающего процессы, связанные с быстрой эволюцией локального или интегрального показателей.

Структура предлагаемой системы обработки потоков больших данных. Обобщая приведенные методы, рассмотрим структуру предлагаемой информационной системы обработки потоков больших данных (рис. 6).

Агенты-извлекатели осуществляют извлечение информации с сайтов, из новостных лент, социальных сетей, изображений, видеопотока. Поскольку предполагается использование большого количества агентов-извлекателей, то архитектура системы изначально предполагает значительный объем трафика между агентами-извлекателями и звеном обработки данных. Поэтому необходимо использование фреймворков для обработки больших данных, ориентированных на потоковую обработку.

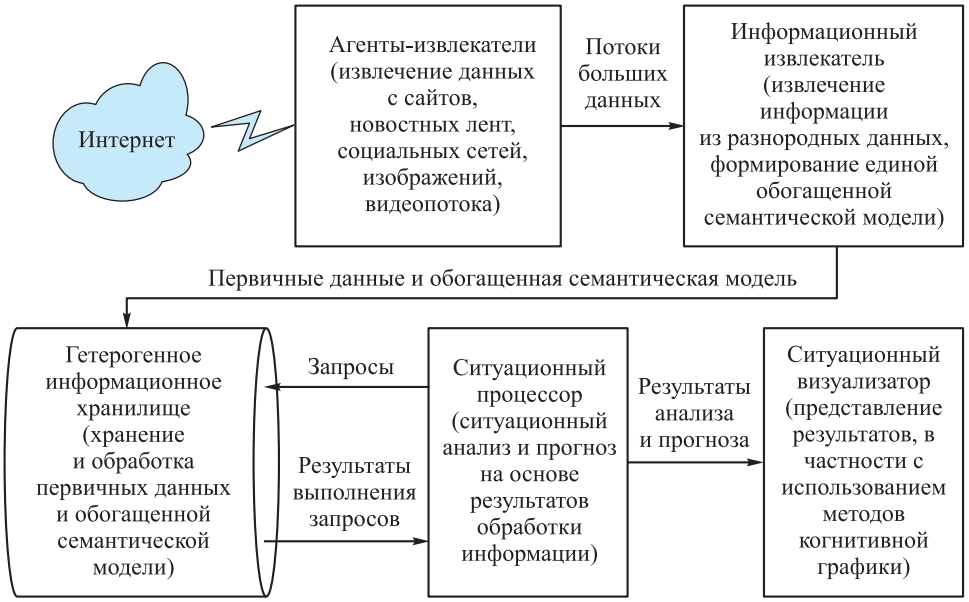


Рис. 6. Структура предлагаемой системы обработки потоков больших данных

Потоки больших данных передаются от агентов-извлекателей на информационный извлекатель, который извлекает информацию из разнородных данных и формирует единую обогащенную семантическую модель, в том числе с использованием алгоритмов машинного обучения.

Для данного звена предполагается выполнить исследование потоковых фреймворков, наиболее подходящих для решения задачи: Apache Flink, Apache Kafka Streaming, Apache Samza, Heron, Apache Spark Streaming.

Извлеченные данные помещаются в гетерогенное информационное хранилище, которое позволяет хранить и обрабатывать как необходимые для дальнейшего анализа первичные данные, так и полученную на их основе обогащенную семантическую модель. Гетерогенное информационное хранилище включает в себя СУБД с графовой или гибридной моделью данных, а также вспомогательную СУБД для хранения первичной информации.

Проведенный анализ показал, что в случае относительно простых ситуационных моделей можно использовать существующие графовые и гибридные СУБД: Titan, Neo4j, OrientDB, а также существующие системы обработки графов: Gradoop, Apache Flink Gelly, Apache Giraph, Google Pregel. Разработка детальной архитектуры системы на основе данных продуктов требует дополнительного исследования. Особенное внимание предполагается уделить таким сложным графовым моделям, как гиперграфы (для которых существует СУБД HypergraphDB) и метаграфы, для которых не разработано СУБД. Систем обработки сложных графов на данный момент также не разработано. Для обработки обогащенной семантической модели предполагается использование методов аппаратного ускорения для выполнения операций над графами.

Результаты выполнения запросов из гетерогенного информационного хранилища поступают на вход ситуационного процессора, который реализует разные алгоритмы ситуационного анализа и прогноза на основе графовых моделей, в том числе с использованием алгоритмов машинного обучения.

Результаты ситуационного анализа и прогноза поступают на ситуационный визуализатор, который позволяет представлять результаты анализа и прогноза в удобной для ЛПР форме. Важную роль в визуализации ситуаций играют методы когнитивной графики, в частности метод динамического анаморфирования.

Таким образом, рассмотренные методы позволили сформировать целостную структуру предлагаемой семиотической системы обработки потоков больших данных.

Заключение. Предложен подход к созданию ИС, осуществляющей обработку и глубокий анализ потоков разнородных данных. В связи с необходимостью обработки данных, загружаемых из большого числа источников информации, в реальном времени в качестве основы ИС предложено использовать потоковый процессор. В результате анализа современных потоковых процессоров выявлено, что текущее состояние области не позволяет принять окончательное решение о выборе потокового процессора без прототипирования, поскольку поведение разных потоковых процессоров сильно различается в зависимости от конкретных решаемых задач.

На основе собранных и агрегированных данных проведены мониторинг и прогнозирование развития ситуаций в целях поддержки принятия управленческих решений. Предложенный подход основан на последовательном выполнении обнаружения событий в текстовом потоке, формирования ситуаций и построения сценариев их дальнейшего развития. Использование такого метода машинного обучения, как метод опорных векторов, обеспечивает возможность гибкой настройки в соответствии с потребностями пользователей. Обученная ИС выделяет оптимистический и пессимистический сценарии с помощью метода анализа иерархий, определяет вероятность реализации сценариев, используя метод логистической регрессии, а также формирует предложения по действиям, необходимым для обеспечения развития ситуации по наиболее благоприятному сценарию.

Применение подхода, основанного на принципах обработки больших графов, позволяет использовать более сложные графовые модели событий и ситуаций, отражающие причинно-следственные и иерархические связи между событиями, в частности метаграфовую модель.

В МГТУ им. Н.Э. Баумана разработан принципиально новый процессор для обработки множеств и структур данных (процессор обработки структур), который успешно апробирован при решении задач на графах. Предполагается, что его применение для хранения и обработки метаграфовых моделей в составе аппаратного обеспечения семиотической системы обработки потоков больших данных позволит увеличить ее быстродействие, а также сократит время разработки и верификации системы.

Для достижения высокой эффективности использования полученных результатов анализа потока данных ЛПР применяется метод динамического анаморфирования, позволяющий отображать и анализировать сценарии развития ситуаций во времени. Наглядность отображения достигается путем преобразования исходных визуальных образов в соответствии со значениями показателей, актуальных при решении поставленной задачи. Для формирования сценариев дальнейшего развития ситуации выполняется построение анаморфозы на основе прогнозных значений показателей, полученных путем использования метода символьной регрессии.

В результате обобщения приведенных методов ситуационного анализа и графической визуализации потоков больших данных предложена структура семиотической системы.

ЛИТЕРАТУРА

1. *Kreps J.* Putting Apache Kafka to use: a practical guide to building a stream data platform (part 1). URL: https://www.confluent.io/blog/stream-data-platform-1 (дата обращения: 10.02.2017).

2. *Самарев Р.С.* Обзор состояния области потоковой обработки данных // Труды ИСП РАН. 2017. Т. 29. № 1. С. 231–260. DOI: 10.15514/ISPRAS-2017-29(1)-13
URL: http://www.ispras.ru/proceedings/isp_29_2017_1/isp_29_2017_1_231 (дата обращения: 20.02.2017).

3. *Andrade H.C.M., Gedik B., Turaga D.S.* Fundamentals of stream processing: application design, systems, and analytics. Cambridge University Press, 2014. 558 p.

4. *Chintapalli S., Dagit D., Evans B., Farivar R., et al.* Benchmarking streaming computation engines: storm, flink and spark streaming // IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). 2016. P. 1789–1792. DOI: 10.1109/IPDPSW.2016.138
URL: http://ieeexplore.ieee.org/document/7530084

5. *Марц Н., Уоррен Дж.* Большие данные. Принципы и практика построения масштабируемых систем обработки данных в реальном времени. М.: Вильямс, 2016. 368 с.

6. *Defining* N-ary relations on the semantic web // W3C Working Group Note 12 April 2006.
URL: http://www.w3.org/TR/swbp-n-aryRelations (дата обращения: 10.02.2017).

7. *Черненький В.М., Терехов В.И., Гапанюк Ю.Е.* Представление сложных сетей на основе метаграфов // Нейроинформатика — 2016. XVIII Всероссийская научно-техническая конференция. Сборник научных трудов. Ч. 1. М.: МИФИ, 2016. С. 225–235.

8. *Самохвалов Э.Н., Ревунков Г.И., Гапанюк Ю.Е.* Использование метаграфов для описания семантики и прагматики информационных систем // Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение. 2015. № 1. С. 83–99. DOI: 10.18698/0236-3933-2015-1-83-99

9. *Гапанюк Ю.Е., Ревунков Г.И., Федоренко Ю.С.* Предикатное описание метаграфовой модели данных // Информационно-измерительные и управляющие системы. 2016. № 12. С. 122–131.

10. *Березкин Д.В., Ревунков Г.И., Гапанюк Ю.Е.* Подходы к построению гибридных хранилищ данных для информационных систем // Гибридные и синергетические интеллектуальные системы. Материалы III Всероссийской поспеловской конференции с международным участием. Калининград: Изд-во БФУ им. И. Канта, 2016. С. 141–148.

11. *Macko P., Margo D., Seltzer M.* Performance introspection of graph databases // Proc. 6th Int. Systems and Storage Conf. (SYSTOR '13). ACM, 2013. 10 p. DOI: 10.1145/2485732.2485750 URL: https://dl.acm.org/citation.cfm?doid=2485732.2485750

12. *Efficient* virtual memory for big memory servers / A. Basu, J. Gandhi, J. Chang, M.D. Hill, M.M. Swift // SIGARCH Comput. Archit. News. 2013. Vol. 41. No. 3. P. 237–248. DOI: 10.1145/2508148.2485943 URL: https://dl.acm.org/citation.cfm?id=2485943

13. *Delorimier M., Kapre N., Mehta N., Dehon A.* Spatial hardware implementation for sparse graph algorithms in GraphStep // ACM Trans. Auton. Adapt. Syst. 2011. Vol. 6. No. 3. Art. 17.

14. *Accelerating* breadth first search on GPU-BOX / T. Mitsuishi, Sh. Nomura, J. Suzuki, Yu. Hayashi, M. Kan, H. Amano // SIGARCH Comput. Archit. News. 2014. Vol. 42. No. 4. P. 81–86. DOI: 10.1145/2693714.2693729 URL: https://dl.acm.org/citation.cfm?doid=2693714.2693729

15. *FPGA-based* hardware acceleration for boolean satisfiability / K. Gulati, S. Paul, S.P. Khatri, S. Patil, A. Jas // ACM Trans. Des. Autom. Elect. Syst. 2009. Vol. 14. No. 2. DOI: 10.1145/1497561.1497576 URL: https://dl.acm.org/citation.cfm?id=1497576

16. *SQRL:* hardware accelerator for collecting software data structures / Sn. Kumar, A. Shriraman, V. Srinivasan, D. Lin, J. Phillips // Proc. 23rd Int. Conf. on Parallel Architectures and Compilation (PACT'14). ACM, 2014. P. 475–476. DOI: 10.1145/2628071.2628118 URL: https://dl.acm.org/citation.cfm?id=2628118

17. *Popov A.* An introduction to the MISD technology // Proc. 50th Hawaii Int. Conf. on System Sciences (HICSS50). 2017. P. 1003–1012.

18. *Андреев А.М., Березкин Д.В., Козлов И.А.* Подход к автоматизированному мониторингу тем на основе обнаружения событий в потоке текстовых документов // Информационно-измерительные и управляющие системы. 2017. Т. 15. №. 3. С. 49–60.

19. *Андреев А.М., Березкин Д.В., Козлов И.А.* Подход к автоматизированному мониторингу и прогнозированию развития инновационных образовательных технологий // Наука и образование: научное издание. 2016. № 7. С. 196–208. URL: http://technomag.edu.ru/jour/article/view/999

20. *Гусейн-заде С.М., Тикунов В.С.* Анаморфозы: что это такое? М.: Эдиториал УРСС, 1999. 168 с.

21. *Терехов В.И.* Применение когнитивной компьютерной графики в системах поддержки принятия решения должностных лиц органов военного управления. М.: Общевойсковая академия ВС РФ, 2012. 150 с.

22. *Терехов В.И.* Применение когнитивной компьютерной графики для визуализации актуальной информации лицам, принимающим решение // Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение. 2012. Спец. вып. С. 109–119.

23. *Koza J.* Genetic programming: on the programming of computers by means of natural selection. MIT Press, 1992. 840 p.

Пролетарский Андрей Викторович — д-р техн. наук, профессор, декан факультета «Информатика и системы управления», заведующий кафедрой «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Березкин Дмитрий Валерьевич — канд. техн. наук, доцент кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Гапанюк Юрий Евгеньевич — канд. техн. наук, доцент кафедры «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Козлов Илья Андреевич — младший научный сотрудник кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Попов Алексей Юрьевич — канд. техн. наук, доцент кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Самарев Роман Станиславович — канд. техн. наук, доцент кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Терехов Валерий Игоревич — канд. техн. наук, доцент кафедры «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана (Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5, стр. 1).

Просьба ссылаться на эту статью следующим образом:

Пролетарский А.В., Березкин Д.В., Гапанюк Ю.Е., Козлов И.А., Попов А.Ю., Самарев Р.С., Терехов В.И. Методы ситуационного анализа и графической визуализации потоков больших данных // Вестник МГТУ им. Н.Э. Баумана. Сер. Приборостроение. 2018. № 2. С. 98–123. DOI: 10.18698/0236-3933-2018-2-98-123

METHODS OF SITUATION ANALYSIS AND GRAPHICAL VISUALISATION OF BIG DATA STREAMS

A.V. Proletarskiy pav@bmstu.ru
D.V. Berezkin berezkind@bmstu.ru
Yu.E. Gapanyuk gapyu@bmstu.ru
I.A. Kozlov kozlovilya89@yandex.ru
A.Yu. Popov alexpopov@bmstu.ru
R.S. Samarev samarev@acm.org
V.I. Terekhov terekchow@bmstu.ru

Bauman Moscow State Technical University, Moscow, Russian Federation

Abstract

The article presents an approach to developing a data processing system dealing with manipulation and deep analytics of heterogeneous data streams. We analysed existing instruments for stream data processing. In order to support decision making, we extract the necessary information from message streams to monitor and predict evolution of situations. We base our approach on sequential event detection in a text stream, forming situations and constructing scenarios of their subsequent evolution. We suggest using complex graph models, the metagraph model

Keywords

Semiotic system, complex graph models, metagraph, event detection, scenario analysis, cognitive computer graphics, dynamic anamorphosis technique

in particular, in order to represent cause-and-effect and hierarchical connections between events and situations. We consider the issues of developing hardware for acceleration of operations dealing with these models. We employ techniques from so-called "cognitive graphics", in particular dynamic anamorphosis, in order to ensure that the decision maker uses the results of this data stream analysis in a highly efficient manner. We propose a semiotic system structure for processing big data streams

Received 11.04.2017

REFERENCES

[1] Kreps J. Putting Apache Kafka to use: a practical guide to building a stream data platform (part 1). Available at: https://www.confluent.io/blog/stream-data-platform-1 (accessed: 10.02.2017).

[2] Samarev R.S. Review of streaming processing field. *Trudy ISP RAN* [Proceedings of ISP RAS], 2017, vol. 29, no. 1, pp. 231–260 (in Russ.). DOI: 10.15514/ISPRAS-2017-29(1)-13
Available at: http://www.ispras.ru/proceedings/isp_29_2017_1/isp_29_2017_1_231

[3] Andrade H.C.M., Gedik B., Turaga D.S. Fundamentals of stream processing: application design, systems, and analytics. Cambridge University Press, 2014. 558 p.

[4] Chintapalli S., Dagit D., Evans B., Farivar R., Graves T., Holderbaugh M., Liu Z., Nusbaum K., Patil K., Peng B.J., Poulosky P. Benchmarking streaming computation engines: storm, flink and spark streaming. *IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2016, pp. 1789–1792. DOI: 10.1109/IPDPSW.2016.138
Available at: http://ieeexplore.ieee.org/document/7530084

[5] Marz N., Warren J. Data: principles and best practices of scalable realtime data systems. Manning Publications, 2015. 328 p.

[6] Defining N-ary relations on the semantic web. W3C Working Group Note 12 April 2006. Available at: http://www.w3.org/TR/swbp-n-aryRelations (accessed: 10.02.2017).

[7] Chernen'kiy V.M., Terekhov V.I., Gapanyuk Yu.E. [Representation of complex networks based on metagraphs]. *Neyroinformatika — 2016. XVIII Vserossiyskaya nauchno-tekhnicheskaya konferentsiya. Sbornik nauchnykh trudov. Ch. 1* [Neuroinformatics — 2016. Proc. XVIII Russ. Sci.-Tech. Conf. P. 1]. Moscow, MEPhI Publ., 2016. Pp. 225–235 (in Russ.).

[8] Samokhvalov E.N., Revunkov G.I., Gapanyuk Yu.E. Metagraphs for information systems semantics and pragmatics definition. *Vestn. Mosk. Gos. Tekh. Univ. im. N.E. Baumana, Priborostr.* [Herald of the Bauman Moscow State Tech. Univ., Instrum. Eng.], 2015, no. 1, pp. 83–99 (in Russ.). DOI: 10.18698/0236-3933-2015-1-83-99

[9] Gapanyuk Yu.E., Revunkov G.I., Fedorenko Yu.S. Predicate representation of metagraph data model. *Informatsionno-izmeritel'nye i upravlyayushchie sistemy* [Information-measuring and Control Systems], 2016, no. 12, pp. 122–131 (in Russ.).

[10] Berezkin D.V., Revunkov G.I., Gapanyuk Yu.E. [Approaches to hybrid databank construction for informational system]. *Gibridnye i sinergeticheskie intellektual'nye sistemy. Materialy III Vserossiyskoy Pospelovskoy konferentsii s mezhdunarodnym uchastiem* [Hybrid and synergetic intelligence systems. Proc. III Russ. Pospelov Conf. with International Participation]. Kaliningrad, I. Kant BFU Publ., 2016. Pp. 141–148 (in Russ.).

[11] Macko P., Margo D., Seltzer M. Performance introspection of graph databases. *Proc. 6th Int. Systems and Storage Conf. (SYSTOR '13)*. ACM, 2013. 10 p. DOI: 10.1145/2485732.2485750
Available at: https://dl.acm.org/citation.cfm?doid=2485732.2485750

[12] Basu A., Gandhi J., Chang J., Hill M.D., Swift M.M. Efficient virtual memory for big memory servers. *SIGARCH Comput. Archit. News*, 2013, vol. 41, no. 3, pp. 237–248.
DOI: 10.1145/2508148.2485943 Available at: https://dl.acm.org/citation.cfm?id=2485943

[13] Delorimier M., Kapre N., Mehta N., Dehon A. Spatial hardware implementation for sparse graph algorithms in GraphStep. *ACM Trans. Auton. Adapt. Syst*., 2011, vol. 6, no. 3, art. 17.

[14] Mitsuishi T., Nomura Sh., Suzuki J., Hayashi Yu., Kan M., Amano H. Accelerating breadth first search on GPU-BOX. *SIGARCH Comput. Archit. News*, 2014, vol. 42, no. 4, pp. 81–86.
DOI: 10.1145/2693714.2693729
Available at: https://dl.acm.org/citation.cfm?doid=2693714.2693729

[15] Gulati K., Paul S., Khatri S.P., Patil S., Jas A. FPGA-based hardware acceleration for boolean satisfiability. *ACM Trans. Des. Autom. Elect. Syst*., 2009, vol. 14, no. 2.
DOI: 10.1145/1497561.1497576 Available at: https://dl.acm.org/citation.cfm?id=1497576

[16] Kumar Sn., Shriraman A., Srinivasan V., Lin D., Phillips J. SQRL: hardware accelerator for collecting software data structures. *Proc. 23rd Int. Conf. on Parallel Architectures and Compilation (PACT'14)*, ACM, 2014, pp. 475–476. DOI: 10.1145/2628071.2628118
Available at: https://dl.acm.org/citation.cfm?id=2628118

[17] Popov A. An introduction to the MISD technology. *Proc. 50th Hawaii Int. Conf. on System Sciences (HICSS50)*, 2017, pp. 1003–1012.

[18] Andreev A.M., Berezkin D.V., Kozlov I.A. Automated topic monitoring based on event detection in text stream. *Informatsionno-izmeritel'nye i upravlyayushchie sistemy* [Information-measuring and Control Systems], 2017, vol. 15, no. 3, pp. 49–60 (in Russ.).

[19] Andreev A.M., Berezkin D.V., Kozlov I.A. Podkhod k avtomatizirovannomu monitoringu i prognozirovaniyu razvitiya innovatsionnykh obrazovatel'nykh tekhnologiy. *Nauka i obrazovanie: nauchnoe izdanie* [Science and Education: Scientific Publication], 2016, no. 7, pp. 196–208 (in Russ.). Available at: http://technomag.edu.ru/jour/article/view/999

[20] Guseyn-zade S.M., Tikunov V.S. Anamorfozy: chto eto takoe? [Anamorphosis: what is this?] Moscow, Editorial URSS Publ., 1999. 168 p.

[21] Terekhov V.I. Primenenie kognitivnoy komp'yuternoy grafiki v sistemakh podderzhki prinyatiya resheniya dolzhnostnykh lits organov voennogo upravleniya [Application of cognitive computer graphics in decision support systems of military administration official bodies]. Moscow, Obshchevoyskovaya akademiya VS RF Publ., 2012. 150 p.

[22] Terekhov V.I. Application of cognitive computer graphics for urgent data visualization by decision-making persons. *Vestn. Mosk. Gos. Tekh. Univ. im. N.E. Baumana, Priborostr.* [Herald of the Bauman Moscow State Tech. Univ., Instrum. Eng.], 2012, spec. iss, pp. 109–119 (in Russ.).

[23] Koza J. Genetic programming: on the programming of computers by means of natural selection. MIT Press, 1992. 840 p.

Proletarskiy A.V. — Dr. Sc. (Eng.), Professor, Dean of Faculty of Information Technology and Control Systems, Head of Department of Computer Systems and Networks, Bauman Moscow State Technical University (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Berezkin D.V. — Cand. Sc. (Eng.), Assoc. Professor, Department of Computer Systems and Networks, Bauman Moscow State Technical University (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Gapanyuk Yu.E. — Cand. Sc. (Eng.), Assoc. Professor, Department of Information Processing and Control Systems, Bauman Moscow State Technical University (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Kozlov I.A. — Junior Research Fellow, Department of Computer Systems and Networks, Bauman Moscow State Technical University (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Popov A.Yu. — Cand. Sc. (Eng.), Assoc. Professor, Department of Computer Systems and Networks, Bauman Moscow State Technical University (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Samarev R.S. — Cand. Sc. (Eng.), Assoc. Professor, Department of Computer Systems and Networks, Bauman Moscow State Technical University (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Terekhov V.I. — Cand. Sc. (Eng.), Assoc. Professor, Department of Information Processing and Control Systems, Bauman Moscow State Technical University (2-ya Baumanskaya ul. 5, str. 1, Moscow, 105005 Russian Federation).

Please cite this article in English as:

Proletarskiy A.V., Berezkin D.V., Gapanyuk Yu.E., Kozlov I.A., Popov A.Yu., Samarev R.S., Terekhov V.I. Methods of Situation Analysis and Graphical Visualisation of Big Data Streams. *Vestn. Mosk. Gos. Tekh. Univ. im. N.E. Baumana, Priborostr*. [Herald of the Bauman Moscow State Tech. Univ., Instrum. Eng.], 2018, no. 2, pp. 98–123 (in Russ.).
DOI: 10.18698/0236-3933-2018-2-98-123