3. Большев А. К., Яновский В. В. Подход к обнаружению аномального трафика в компьютерных сетях с использованием методов кластерного анализа // Изв. СПбГЭТУ «ЛЭТИ». 2006. № 3. С. 38–45.

4. Большев А. К., Лавров А. А. Метод идентификации версии системного программного обеспечения удаленного сетевого узла, основанный на комплексном анализе характеристик TCP/IP // Изв. СПбГЭТУ «ЛЭТИ». 2012. № 1. С. 45–51.

5. Лавров А. А., Яновский В. В. Идентификация операционной системы удаленного хоста методами анализа временных характеристик // Изв. СПбГЭТУ «ЛЭТИ». 2011. № 3. С. 34–39.

6. Лавров А. А., Лисс А. Р. Метод опорных векторов в задаче идентификации версии операционной системы удаленного узла // Изв. СПбГЭТУ «ЛЭТИ». 2013. № 10. С. 11–15.

A. K. Bolshev, A. A. Lavrov
*Saint-Petersburg state electro technical university «LETI»*

THE SOFTWARE FOR MONITORING DATA PROCESSING DIGITAL COMPUTING SYSTEM OF HYDROACOUSTIC STATION BASED ON NAGIOS

*The features of monitoring board digital computing systems and role of software for monitoring computing systems of hydroacoustic stations is examined. The real time software for monitoring data processing digital computing system of hydroacoustic station is proposed.*

**Hydroacoustic data processing digital computing systems, network monitoring, Nagios**

УДК: 20.53.19, 28.23.13

И. И. Холод
*Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)*

# Архитектура «облака» интеллектуального анализа данных на основе библиотеки алгоритмов с блочной структурой

*Описывается архитектура платформы, интегрирующая в себе технологии интеллектуального анализа данных и облачных вычислений. Предлагается построение «облака» интеллектуального анализа данных на базе библиотеки алгоритмов интеллектуального анализа данных, построенных на принципах функционально-блочной структуры.*

**Облако интеллектуального анализа данных, интеллектуальный анализ данных, облачные вычисления, интеллектуальный анализ распределенных данных**

За последние десятилетия в результате повсеместного использования информационных технологий у человечества накоплены огромные объемы данных. Эти данные содержат множество полезной информации, анализ которой позволяет переосмысливать пройденный путь и правильно выбирать дорогу в будущем. Неудивительно, что в последнее время бурно развивается технология интеллектуального анализа данных. Основным преимуществом данной технологии является извлечение из данных более сложных (по сравнению с математической статистикой) видов закономерностей, скрытых в данных, но при этом более понятных человеку. Наибольший эффект от методов интеллектуального анализа достигается при их применении к большим объемам данным. Это требует больших объемов памяти хранилищ и высокой производительности аппаратной составляющей, на которой хранятся и обрабатываются данные. Наиболее эффективным способом решения этой проблемы является распараллеливание алгоритмов интеллектуального анализа и выпол-

нение их на параллельных и распределенных вычислительных системах. Безусловно, недостатком данного решения является цена построения и создания таких систем. Усложняется это решение и тем, что с ростом объема анализируемых данных должны наращиваться как объемы хранилищ, так и производительные мощности вычислительных узлов. Таким образом, требованию масштабирования должны отвечать как сами алгоритмы, так и среда, в которой они будут выполняться.

Для решения данной проблемы в последнее время бурно развиваются технологии облачных вычислений. Облачные вычисления – это модель обеспечения повсеместного и удобного сетевого доступа по требованию к общему пулу конфигурируемых вычислительных ресурсов (например, вычислительным серверам, устройствам хранения данных, приложениям и сервисам – как вместе, так и по отдельности), которые могут быть оперативно предоставлены и освобождены с минимальными эксплуатационными затратами и/или обращениями к провайдеру. Основными свойствами такой модели являются [1]:

– самообслуживание по требованию;

– широкий сетевой доступ;

– оперативная эластичность;

– пул ресурсов;

– измеримость услуг.

В настоящее время выделяют несколько основных моделей облачных вычислений [1]:

– программное обеспечение как услуга (SaaS – Software as a Service);

– платформа как услуга (PaaS – Platform as a Service);

– инфраструктура как услуга (IaaS – Infrastructure as a Service);

– рабочее место как услуга (WaaS – Workplace as a Service).

Учитывая возросшие требования к производительности алгоритмов интеллектуального анализа данных, логичным является интеграция двух технологий: интеллектуального анализа данных и облачных вычислений. Результатом такой интеграции будет создание «облака» интеллектуального анализа данных (Data Mining Cloud). Такое решение имеет ряд достоинств:

1. Вопросы реализации, обновления и сопровождения алгоритмов анализа скрыты от конечного пользователя.

2. Пользователь всегда использует самые последние версии алгоритмов.

3. Алгоритмы анализа (являющиеся ресурсоемкими) могут использовать все доступные вычислительные средства, имеющиеся в «облаке».

4. Возможно использование алгоритмов анализа как через веб-интерфейс, так и через программный интерфейс, созданный на основе веб-сервисов.

5. Алгоритмы могут применяться к данным, хранящимся как в «облаке», так и вне его.

6. Пользователь может не задумываться о масштабировании алгоритмов.

Данное решение имеет и недостатки:

1. Необходимость обеспечения высокого уровня безопасности к анализируемым данным.

2. Вероятность недоступности сервисов и/или прекращение существования провайдера.

В данном направлении активно ведутся работы в зарубежных институтах и IT-компаниях. Одним из первых в этой области начал работать Китайский мобильный институт. В 2007 г. в нем начались исследования и разработки в области облачных вычислений. В 2009 г. он официально анонсировал платформу для облачных вычислений BigCloud, включающую в себя инструменты для параллельного выполнения алгоритмов Data Mining **Big Cloud-Parallel Data Mining (BC-PDM)** [2].

BC-PDM представляет собой SaaS-платформу, построенную на базе Apache Hadoop. Пользователи могут загружать данные в хранилище (размещенное в облаке) из разных источников и применять к ним различные приложения по управлению данными, анализу данных и бизнес-приложения. В состав приложений анализа входят параллельные приложения, выполняющие ETL-обработку, анализ социальных сетей, анализ текстов (Text Mining), анализ данных (Data Mining), статистический анализ.

В 2009 г. компания Amazon расширила свои облачные сервисы Amazon EC2 и Amazon Simple Storage Service (Amazon S3) еще одним PaaS-сервисом – Amazon Elastic Mapreduce (EMR) [3]. Данный сервис также построен на платформе Apache Hadoop и предоставляет масштабируемую инфраструктуру для выполнения созданных пользователем (по определенным правилам) приложений. Сервис позволяет загрузить в Amazon S3 необходимое приложение и/или данные, которые в дальнейшем будут выполнены на job-узлах платформы Hadoop. В состав EMR входят примеры приложений, которые могут быть загружены в сер-

вис, в том числе и решающие задачи Data Mining. Таким образом, для выполнения анализа на EMR необходимо выполнить 4 следующих шага:

1. Создать приложение для анализа данных.

2. Загрузить данные и/или приложения на Amazon S3.

3. Запустить job-узел через консоль управления (AWS ManagementConsole) со ссылками на ранее загруженные в Amazon S3 данные и/или приложения.

4. Наблюдать через консоль управления за процессом анализа до его завершения.

Amazon EMR-первый из сервисов, который может быть классифицирован как SaaS и PaaS одновременно, в зависимости от его использования конечным пользователем.

В 2012 г. корпорация «Google» опубликовала свой новый облачный сервис **Google BigQuer** [4]. Данный сервис позволяет обрабатывать большие объемы данных, хранящихся в облаке. Если данные в облаке отсутствуют, то их предварительно необходимо загрузить туда. При загрузке данные трансформируются во внутренние форматы (сейчас поддерживаются 2 формата – CSV и JSON), в которых они потом используются для обработки. Место для загрузки данных ограничено (100 Гбайт бесплатно, свыше – платно).

Доступ к сервису может осуществляться с использованием веб-браузера, клиентского программного обеспечения, реализующего поддержку командной строки, или через API для распределенных систем, построенных в соответствии с архитектурным стилем REST [5]. При использовании веб-браузера и консольных приложений пользователь для обработки своих данных должен ввести SQL-подобный запрос. Он может включать все те же элементы, что и Select-запрос в SQL: FROM, JOIN, WHERE и др. Таким образом, пользователь имеет возможность сформировать довольно гибкий поисковый запрос по данным произвольного типа.

Необходимо отметить, что данный сервис не решает напрямую задач Data Mining: кластеризации, классификации и др., поэтому его нельзя рассматривать в чистом виде как Data Mining Cloud, скорее, он относиться к классу OLAP-систем.

Помимо существующих открытых облачных сервисов, которые могут быть использованы для решения задач Data Mining, существует ряд разработок, на базе которых может быть построен облачный интеллектуальный анализ данных.

Корпорация «Microsoft» предлагает сервер SQL Server Data Mining [6], который может быть использован для решения задач Data Mining в облаке. Данный сервер построен как WCF (Windows Communication Foundation)-приложение. Он предоставляет .NET API для написания сервисно-ориентированных приложений. Кроме того он позволяет расширять состав алгоритмов добавлением соответствующих плагинов.

На базе SQL Server Data Mining создан тестовый облачный SaaS-сервис, который предоставляет пользователям возможность через специальные приложения, запущенные из браузера, устанавливать данные на сервере/серверах, выбирать инструменты интеллектуального анализа данных, конфигурировать их и просматривать результаты. В качестве инструментов интеллектуального анализа данных сейчас предоставляется анализ ключевых факторов влияния, прогнозирование и расчет прогноза.

Сервис позволяет загружать данные, используя инструмент Load Data для загрузки данных в формате cvs.

Существуют средства для построения Data Mining-«облака» и на базе популярной открытой библиотеки алгоритмов Data Mining – Weka [7]. Ее расширение Weka4WS [8] реализует распределенный каркас для поддержки выполнения алгоритмов Data Mining в среде WSRFenabled Grids. Она интегрирует библиотеку Weka и WSRF-технологию для запуска удаленных Data Mining-алгоритмов и управления распределенными вычислениями в виде Workflow. Такая интеграция позволяет на некотором удаленном вычислительном узле развернуть WSRF-совместимый сервис для публикации всех Data Mining-алгоритмов, включенных в библиотеку Weka.

В настоящее время одним из самых популярных средств Data Mining (по версии KDnuggets [9]) является система RapidMiner, разработанная одноименной компанией. Данный продукт является Open Source, и версия с минимальными функциональными возможностями распространяется бесплатно. RapidMiner реализует клиент-серверную архитектуру. Rapid Miner Server может использоваться отдельно, предоставляя возможности интеллектуального анализа в виде веб-сервисов, тем самым реализуя модель облачных вычислений – SaaS [10].

RapidMiner реализует все необходимые операции для анализа: загрузку и преобразование данных (ETL), предобработку данных, визуализа-

цию данных, решение задач Data Mining. Он имеет открытую архитектуру, предоставляя возможность расширять себя новыми алгоритмами, в том числе и алгоритмами, реализованными из библиотек Weka и R.

Основными недостатками описанных систем являются:

1. Необходимость пользователю самому создавать аналитические приложения и/или SQL-подобные скрипты для их выполнения в «облаке».

2. Необходимость хранения анализируемых данных внутри облака, что, в свою очередь, имеет ряд недостатков:

– требует дополнительных аппаратных средств для хранения данных;

– для обработки актуальных данных нужно или всегда хранить их во внутреннем хранилище, или решить задачу их синхронизации с источником информации;

– при загрузке информации производится преобразование во внутренние форматы, что может привести к искажению информации и/или вызвать ошибку при загрузке;

– обеспечение конфиденциальности хранящейся во внутреннем хранилище информации ложится на провайдера сервиса, что не всегда может удовлетворить владельца информации.

3. Использование непараллельных и/или строго распараллеленных алгоритмов, что не позволяет гибко перестраивать алгоритмы в зависимости от выбранных параметров среды выполнения, а также вида данных.

4. Привязка только к одной технологии выполнения распределенных вычислений (в основном Map Reduce и ее реализации Apache Hadoop), каждая из которых имеет свои недостатки и может эффективно применяться только при определенных условиях.

5. Решение конечных бизнес-задач, а не отдельных задач интеллектуального анализа данных, что ограничивает возможность комбинировать данные задачи и решать более широкий круг бизнес-задач.

6. Отсутствие в большинстве решений унифицированных программных интерфейсов, предоставляющих доступ как к сервисам «облака», так и к результатам анализа.

Исходя из изложенного, предлагается построить «облако» интеллектуального анализа, лишенное указанных недостатков и удовлетворяющее следующим требованиям:

1. Возможность работы как с данными внутри облака, так и с данными, хранящимися вне его.

2. Возможность выполнять все этапы анализа: настройку задачи анализа, подготовку и очистку данных, анализ данных, получение результатов анализа, оценку результатов анализа.

3. Возможность выполнять анализ как конечному пользователю (через веб-интерфейс), так и использовать себя в качестве сервиса (через API), сторонним системам.

4. Возможность пользователю гибко конфигурировать как среду выполнения анализа, так и средства, обеспечивающие выполнение распределенных вычислений.

5. Поддержка существующих стандартов и спецификаций в области Data Mining.

Для выполнения всех перечисленных требований предлагается архитектура «облака» интеллектуального анализа данных, включающая в себя следующие уровни (см. рисунок):

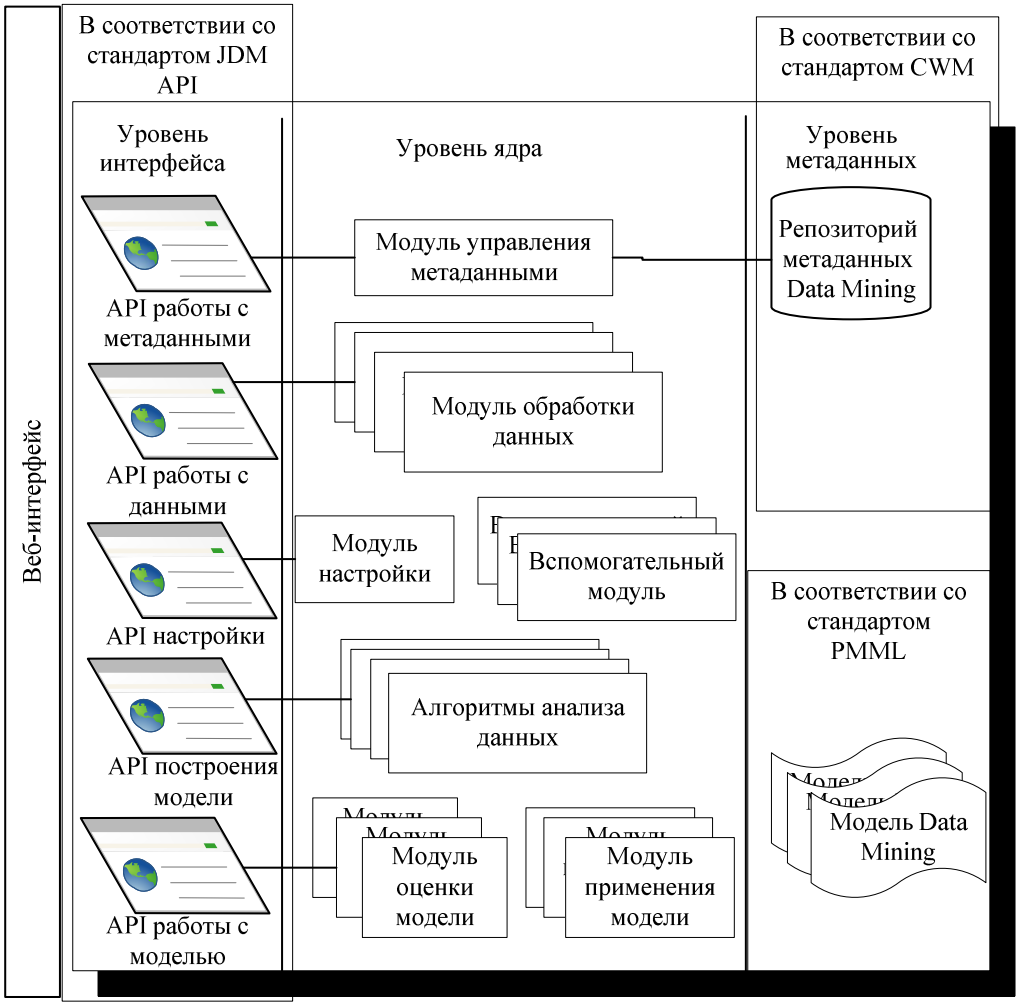
– уровень интерфейса – веб-интерфейс и API к «облаку», позволяющему пользователю выполнить настройку анализа, сам анализ и получать результаты анализа данных;

– уровень ядра – реализация алгоритмов анализа данных, а также вспомогательных методов, позволяющих повысить эффективность анализа;

– уровень метаданных – репозиторий, в котором сохраняются метаданные, порождаемые и необходимые в процессе выполнения анализа.

Для реализации модели облачных вычислений SaaS-сервис должен обеспечивать на уровне интерфейса пользовательский и программный интерфейсы. Чтобы конечный пользователь мог работать с «облаком», оно должно реализовывать пользовательский (GUI) интерфейс. Учитывая, что к «облаку» будут обращаться через Интернет, имеет смысл реализовывать его на основе веб-технологий в виде тонкого/умного клиента. За основу может быть выбрана, например, технология GWT.

Для интеграции с прикладными системами должен поддерживаться программный интерфейс (API), который уместно реализовать в виде веб-сервисов с опубликованным описанием интерфейса на языке WSDL. За основу может быть взят стандарт JDMAPI [11], описывающий основные операции интеллектуального анализа данных в виде Java-интерфейсов. Это позволит использовать облако в системах с сервисно-ориентированной архитектурой.




В обоих случаях должен обеспечиваться следующий общий сценарий работы:

1. Выбор источника данных и методов их предварительной обработки и преобразования (из состава ETL-инструментов).

2. Выбор среды и средств распределенного выполнения (количество вычислительных процессоров, их характеристики, средства распределенных вычислений и др.).

3. Выбор и настройка алгоритма интеллектуального анализа.

4. Выполнение интеллектуального анализа и построение модели знаний.

5. Просмотр и оценка построенной модели знаний.

6. Применение построенной модели знаний к новым данным.

Уровень ядра должен быть реализован на базе библиотеки алгоритмов интеллектуального анализа данных для параллельного и распределенного выполнения – DXelopes [12]. Данная библиотека реализует алгоритмы в функционально-блочной структуре, позволяя перестраивать ее, в том числе и для параллельного выполнения. Данная особенность позволит легко перестраивать параллельные алгоритмы в зависимости от выбранных пользователем условий выполнения. Кроме самих алгоритмов интеллектуального анализа данных должны быть реализованы сопутствующие методы: преобразование данных, настройка процесса анализа, оценка построенных моделей, применение построенных моделей к данным и другие вспомогательные модули.

Открытая архитектура библиотеки DXelopes позволит добавлять как новые алгоритмы, так и расширять другие модули компонентов (для работы с разными источниками данных, для расширения возможностей параллельного выполнения и т. д.), реализованных в соответствии со структурой библиотеки. Это позволит в рамках «облака» реализовать модель PaaS.

На уровне метаданных должно обеспечиваться сохранение данных, порождаемых на разных этапах анализа. При этом необходимо учитывать, что сервисная архитектура не предполагает сохранения состояния процесса анализа (порожда-

емых в рамках процесса метаданных) между вызовами методов сервиса. Другими словами, при вызове метода он будет выполняться независимо от того, какие функции вызывались до него и что будет вызвано после него. Его работа будет зависеть только от параметров, переданных ему в качестве аргументов. Например, при вызове методов настройки алгоритма сами настройки будут возвращены пользователю, однако алгоритм, находящийся внутри «облака», останется неизменным. Для изменения алгоритма и выполнения его в соответствии с настройками их нужно сохранить и передать в метод, запускающий данный алгоритм. При этом сервис, запускающий алгоритм, должен выполнить настройку алгоритма до его запуска.

Сохранение метаданных, порождаемых в процессе анализа, может выполняться как вне «облака» (на стороне пользователя), так и внутри «облака». В первом случае создаваемые метаданные будут передаваться между сервисом и клиентом, что значительно увеличит объем передаваемых данных. Однако при этом пользователь имеет возможность изменять эти метаданные.

Во втором случае метаданные будут сохраняться внутри «облака» в отдельном репозитории. При этом они должны идентифицироваться уникальным образом. Идентификаторы созданных и сохраненных метаданных будут возвращаться методами, породившими их. В дальнейшем эти идентификаторы будут передаваться в методы, использующие эти метаданные, которые в свою очередь по этим идентификаторам будут извлекать их из репозитория. При таком подходе пользователи не имеют непосредственного доступа к метаданным. Для решения этой проблемы в сервис должны быть добавлены методы, позволяющие получить метаданные по идентификатору и сохранить их непосредственно в репозитории. Для работы с репозиторием в сервисе на уровне ядра должен быть реализован модуль управления метаданными, а на уровне интерфейса – соответствующий ему API.

Сам репозиторий будет реализован в соответствии со стандартом CWM [13], определяющим структуру метаданных Data Mining и связи между ними. Модели, создаваемые в процессе анализа, будут строиться и сохранятся в соответствии со стандартом PMML [14], определяющим формат сохранения моделей Data Mining на основе XML.

Реализация описанной архитектуры позволит создать сервис в соответствии с моделью SaaS при использовании уже имеющихся алгоритмов в «облаке» и в соответствии с моделью PaaS в случае добавления в облако собственных алгоритмов. Реализация указанных стандартов позволит унифицировать работу с «облаком» и предоставляемыми им сервисами. Кроме того, «облако», построенное в соответствии с описанными принципами, позволит устранить указанные ранее недостатки.

Исследования, результаты которых описаны в данной статье, выполнены в СПбГЭТУ при финансовой поддержке Министерства образования и науки Российской Федерации в рамках договора № 02.G25.31.0058 от 12.02.2013 г.

## СПИСОК ЛИТЕРАТУРЫ

1. Mell P., Grance T. The NIST Definition of Cloud Computing. Recommendations of the National Institute of Standards and Technology / Computer Security Division Information Technology Laboratory National Institute of Standards and Technology. Gaithersburg, 2011.

2. BC-PDM: Data mining, social network analysis and text mining system based on cloud computing / L. Yu, J. Zheng, W. C. Shen, B. Wu, B. Wang, L. Qian, B. R. Zhang // Proc. of the 18th ACM SIGKDD Intern. conf. on Knowledge discovery and data mining, New York, 2012. P. 1496–1499.

3. Amazon Elastic MapReduce. Developer Guide. URL: http://s3.amazonaws.com/awsdocs/ElasticMapReduce/latest/emr-dg.pdf.

4. Google Developers. Google BigQuery. URL: https://developers.google.com/bigquery/what-is-bigquery?hl=ru.

5. Fielding R. Architectural Styles and the Design of Network-based Software Architectures: dis. / University of California, 2000. P. 180.

6. SQL Server Data Mining. Home. URL: http://www.sqlserverdatamining.com/ssdm/.

7. Weka: Practical machine learning tools and techniques with Java implementations. (Working paper 99/11) / I. H. Witten, E. Frank, L. Trigg, M. Hall, G. Holme & S. J. Cunningham; Hamilton, New Zealand: University of Waikato, Department of Computer Science, 1999.

8. Talia D., Trunfio P., Verta O. The Weka4WS framework for distributed data mining in service-Oriented Grids // Concurrency and Computation: Practice and Experience. 2008. Vol. 20. P.1933–1951.

9. «KDnuggets Annual Software Poll:RapidMiner and R vie for first place» KDnuggets, June 2013. URL: http://www.kdnuggets.com/2013/06/kdnuggets-annual-software-poll-rapidminer-r-vie-for-first-place.html.

10. David Norris, «RapidMiner – a potential game changer» IT-Director.com, November 22, 2013. URL: http://www.it-director.com/content.php?cid=14555.

11. JSR-000073 Data Mining API. (Maintenance Release). URL: https://jcp.org/aboutJava/communityprocess/mrel/jsr073/index.html.

12. Холод И. И., Накидкин А. В. Параллельное выполнение алгоритмов интеллектуального анализа данных на основе асинхронного обмена сообщениями // Изв. СПбГЭТУ «ЛЭТИ». 2012 . № 9. С. 47–53.

13. Common Warehouse Metamodel (CWM) Specification. URL: http://www.omg.org/spec/CWM/1.1/.

14. PMML Specification.: Data Mining Group. URL: http://www.dmg.org/PMML-4_0.

I. I. Kholod
*Saint-Petersburg state electro technical university «LETI»*

DATA MINING CLOUD ARCHITECTURE BASED ON LIBRARY OF ALGORITHMS WITH FUNCTIONAL-BLOCK STRUCTURE

*This article describes the architecture of the platform that integrates data mining and cloud computing technologies. Proposed to build a data mining cloud on the basis of library data mining algorithms based on the principles of functional-block structure.*

**Data Mining Cloud, Data Mining, cloud computing, distributed data mining**

УДК 621.3.049.77.001.2

С. Э. Миронов, А. К. Фролкин
*Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)*

# Средства автоматизации проектирования топологии ячеек СБИС

*Описаны средства проектирования оптимизированной топологии ячеек интегральных схем. Разработанные программные средства генерируют текстовое технологически инвариантное описание эскизов топологии, которое с помощью средств сжатия топологии настраивается на требуемые пользователю проектные нормы.*

**Топологии ячеек, СБИС, размещение элементов, Эйлеровы пути, трассировка, оптимизация топологии, технологически инвариантное проектирование топологии**

**Автоматизация проектирования топологии ячеек СБИС.** В микроэлектронике все большее значение приобретает фактор обеспеченности средствами САПР. При этом область их применения постоянно расширяется, и сейчас помимо таких традиционных задач, как функционально-логическое и электрическое моделирование или сборка топологии макроблоков СБИС библиотек из ячеек, автоматизируются даже такие «неформальные» этапы проектирования, как оптимизация топологии.

Процесс автоматизации проектирования топологии ячеек можно разделить на две большие задачи: задачу оптимизации взаимного расположения транзисторов ячейки и задачу трассировки внутренних связей ячейки. Описанию результатов решения этих задач и посвящена статья.

Для упрощения задача автоматизации проектирования оптимизированной топологии решалась в технологически инвариантной концепции. Она предполагает разработку не детально прорисованных чертежей топологии с учетом жестких конструкторско-технологических требований, а топологических эскизов, которые с помощью систем сжатия топологии могут оперативно настраиваться на требуемые пользователю проектные нормы.