O. O. Lukovenkova
Vitus Bering Kamchatka State University

A. B. Tristanov
Institute of Cosmophysical Research and Radio Wave Propagation FEB RAS

V. V. Geppener
Saint Petersburg Electrotechnical University «LETI»

MODELING TIME-FREQUENCYSTRUCTUREOF GEOACOUSTIC IMPULSESUSING METHODS OF INTELLIGENT ANALYSIS

The paper is devoted to modeling the time-frequency structure of a given class of geoacoustic impulses. A combination of time-frequency, sparse approximation, statistical and intelligent analysis methods is used for modelling. The signal time-frequency representation is constructed using the modified algorithm of adaptive matching pursuit. It is shown that the constructed representations are divided into classes by hierarchical clustering methods. Four classes of geoacoustic impulses have been obtained and analyzed. We suggest revealing internal patterns of the structure of a given impulse class using the algorithms of mining association rules and statistical analysis. Then we used the algorithms in order to model geoacoustic emission impulses. The developed algorithms for analysis and modeling of geoacoustic impulses are implemented using the MATLAB programming language. The designed system has been tested on typical geoacoustic impulses of characteristic shape. The correspondence of real and model impulses is shown, which proves the correctness of the internal structure patterns.

Geoacoustic emission, geoacoustic impulse, time-frequency structure, intelligent analysis of signals, sparse approximation

УДК 681.322

В. В. Цехановский, В. Д. Чертовской
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

Программная реализация иерархических систем управления

Рассмотрены классы реализации отдельного структурного элемента. Рассмотрены 9 классов сочетаний СУБД, алгоритмов, языков и сред реализации. Классы реализованы и апробированы на задаче статического линейного программирования с целью сравнительного выявления их возможностей. По сформулированным требованиям выбран класс InterBase в среде C++ Builder. Приведен прикладной пример реализации для системы управления приемом на работу. Выявлены недостатки класса InterBase в среде C++ Builder при его использовании в сетевом режиме. Выделен ряд технологий, обеспечивающих создание и сопровождение сервисов в распределенных вычислительных системах. При переходе к многоэлементной системе выявились дополнительные требования к методам реализации, на основе которых выполнен анализ сетевых архитектур. Показано предпочтение с позиций управления компонентной архитектуры, схожей с объектно-ориентированным вариантом. Предусмотрен сетевой режим в связке СУБД Denver – среда Apache, позволяющий реализовать процессы в наиболее распространенной трехуровневой системе управления.

Иерархическая система управления, классы реализации, отдельный элемент, система элементов

В последнее время перспективным является использование в корабле-, приборо- и машиностроении адаптивных автоматизированных систем управления производством [1], [2]. Основные технические решения для систем малой размерности апробированы в рамках пакета MatLab. Появилась необходимость решения невысокоразмерных оптимизационных задач реальных систем.

Постановка задачи. Решение проблемы следует искать на пути формирования программного продукта, сочетающего в себе базы данных, которые резко снижают ошибки ввода/вывода и при-

меняются на автоматизированном производстве, стандартные, простые апробированные программы оптимальных алгоритмов и удобного интерфейса пользователя.

Трудности при решении названной проблемы состоят в противоречивости требований, предъявляемых к программным продуктам. Основными требованиями являются: простой ввод (таблицы) и вывод (таблицы) данных в обычной табличной форме; большая размерность всех таблиц с простым экранным доступом; наличие многих сетевых пользователей в режиме «клиент-сервер» и связи со средой; предпочтительное применение только одного вида программного продукта; возможность оптимизации процессов планирования и управления посредством использования распространенного программного пакета; простота построения интерфейса; блочная иерархичная программная структура, удобная для оперативной ее модернизации.

Решение задачи. Перечисленным требованиям, как показал предварительный анализ, не удовлетворяет ни один из основных широко используемых программных монопродуктов. В связи с этим потребовалась интеграция нескольких программных средств.

Действительно, при значительном количестве данных требуется их упорядочение. При этом наиболее удобной формой являются базы данных с использованием таких СУБД, как Oracle, InterBase в среде Delphi или C++ Builder. Однако в этих СУБД отсутствуют программы оптимизационных алгоритмов, а их реализация на языке SQL сильно затруднена из-за недостаточных возможностей декларативного языка. Большое разнообразие алгоритмов, в том числе оптимизационных, содержится в пакете MatLab, однако в этом пакете проблематично построение баз данных.

Варианты решения высокоразмерной задачи. Рассмотрение проведем в 2 этапа: для отдельного структурного элемента; для системы элементов.

Перечисленным требованиям в наибольшей мере удовлетворяет СУБД InterBase в среде Delphi или C++ Builder и пакет MatLab.

Возможны следующие варианты реализации, которые назовем классами [2].

Класс 1. В качестве СУБД используется InterBase, а оптимизационный алгоритм, базирующийся на алгоритме Р. Габасова, находится в Delphi [1]. Алгоритм Р. Габасова может быть заменен алгоритмом simplex.

Класс 2. Передача данных из InterBase в MatLab с оптимизационным алгоритмом linprog.

Класс 3. Передача алгоритма (программы) из MatLab в Delphi, где осуществляется управление и вывод результата [2].

Класс 4. Использование внешних функций для ввода программ непосредственно в InterBase [3].

Заслуживает внимания работа СУБД InterBase в среде Borland Builder C++, на которой базируются классы 5–8. Среда Builder, являясь в определенной степени аналогом среды Delphi, имеет более широкие возможности из-за большей универсальности языка C++ в сравнении с языком Object Pascal.

Класс 5 является аналогом класса 1 и использует стандартную оптимизационную программу в рамках среды Builder C++.

Классы 6 и *7* строятся по аналогии с классами 2 и 3.

Класс 8 – фактически аналог класса 4.

Авторами были реализованы и апробированы на задаче статического линейного программирования классы 1–8 с целью сравнительного выявления их возможностей.

К минусам классов 1, 5 относятся затруднение дальнейшего сетевого использования полученного результата в режиме «клиент-сервер»; недостаточная проверка работы на разных задачах СЛП; реализация задач в Delphi, затрудняющая применение режима. Это особенно характерно для класса 1.

В классах 2 и 6 организация передачи данных из БД в MatLab и обратно сложна в силу того, что в разных программных продуктах типы данных существенно различны. К тому же в процессе передачи данных требуется серьезное вмешательство квалифицированного оператора, что может сильно затруднить работу обычного пользователя.

В классах 3 и 7 [4] использована «библиотека динамической компоновки или dynamic-link library – dll», называемая также и «динамически подключаемой библиотекой». Здесь определенное предпочтение следует отдать классу 7, хотя имеются некоторые затруднения с построением сетевого режима.

Классы 4 и 8 связаны с внешними функциями, которые требуется строить дополнительно. Предпочтителен класс 8, обладающий хорошими предпосылками для режима «клиент-сервер».

Проведенный анализ показывает, что не существует решения, удовлетворяющего всем перечисленным ранее требованиям.

Самым реальным из исследованных классов является в настоящее время вариант связки программ InterBase – Builder C++. Иллюстрируем вариант реализации на примере базы знаний «Прием на работу» по анкетным данным как помощь сотруднику отдела кадров. Схема базы знаний приведена на рис. 1.

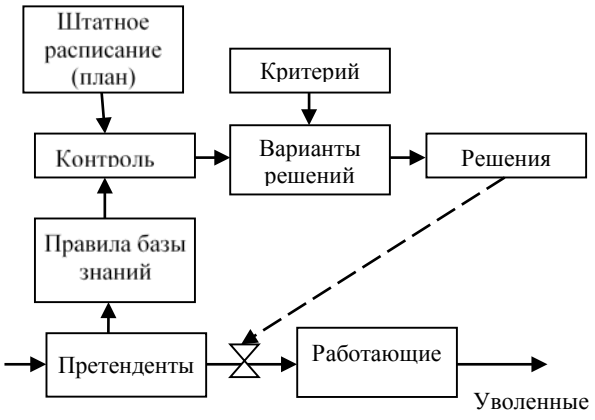

Рис. 1

Прием проводится в течение трех интервалов времени. Претенденты характеризуются данными, фрагмент которых показан в табл. 1 (Кадры). Прием осуществляется по правилам, представленным в табл. 2 (Правила). На каждом интервале времени компьютер предлагает решения-советы о принимаемых по трем должностям: научный сотрудник, инженер-конструктор и инженер по эксплуатации.

Работа с базой знаний ведется в следующем порядке:

1. Запустить exe-файл или программу Builder C++. При открытии формы Builder C++ текущим становится первый интервал времени (цикл). В меню доступным остается элемент меню (ЭМ) Обработать/Начать, тогда как все другие элементы недоступны. Устанавливаются количество циклов (по умолчанию 3) и текущий первый цикл.

2. После нажатия ЭМ Обработать/Начать данные из нерабочих таблиц вводятся в одноименные рабочие таблицы, т. е. компьютерная система после проведенного ранее сеанса работы возвращается в исходное положение.

3. После этого закрывается доступ к ЭМ Обработать/Начать и становится доступным ЭМ Показать/Штатное расписание, с помощью которого вызывается соответствующая таблица. После закрытия этой таблицы «открываются» ЭМ Показать/Кадры, Показать/Работающие, Показать/Претенденты, выдающие на экран монитора для последовательного просмотра пользователем соответствующие таблицы.

При просмотре таблицы Претенденты на экране делается доступной кнопка «Далее». Если ее не нажимать, то можно вернуться к новому просмотру всех перечисленных таблиц.

4. Нажатие кнопки «Далее» прекращает доступ к перечисленным ранее ЭМ. Открывается доступ к ЭМ Обработать/Запуск правил. При нажатии ЭМ правил вырабатывают для ЛПР решения-советы о принятии претендентов на работу. Статус «претенденты» заменяется на «принимаемые». Одновременно по каждой должности подсчитывается количество принимаемых. ЭМ Обработать/Запуск правил становится недоступным.

5. Список принимаемых в виде таблицы можно получить, нажав ЭМ Показать/Принимаемые, при этом будут отражены и должности «отказать».

6. После просмотра таблицы закрывается доступ к ЭМ Показать/Принимаемые и доступным становится ЭМ Обработать/Количественный результат. На экране появляется таблица Штатное

Таблица 1

Время	Фамилия И. О.	Ученая степень	Открытия	Средний балл	Стаж	Должность
–	Водопьянов Н. В.	Да	Да	4.8	7	Научный сотрудник
1	Каримов Э. К.	Да	Да	3.3	5	Научный сотрудник
1	Крамской Б. В.	Да	Да	4.2	8	Научный сотрудник
3	Борин В. Н.	Да	Да	4.6	5	Претендент

Таблица 2

Номер правила	Ученая степень	Открытия	Средний балл	Стаж	Должность
1	Нет				Отказать
2	Да	Да			Научный сотрудник
3	Да	Нет	≥ 3.5		Инженер-конструктор
4	Да	Нет	< 3.5	≥ 2	Инженер по эксплуатации
5	Да	Нет	< 3.5	< 2	Отказать

расписание, в которой указано количество вакансий и принимаемых. Открывается доступ к ЭМ Обработать/Объяснения.

7. Если у пользователя какие-либо результаты работы компьютера вызывают сомнения, то следует нажать ЭМ Обработать/Объяснения. Получается словесное описание правила, с помощью которого для какой-либо записи получен результат. Появляется кнопка «Далее».

8 Нажатие кнопки «Далее» может иметь 2 исхода.

9. Если принимаемых больше, чем вакансий, то переход к следующему этапу работы программного продукта невозможен. При этом выдается дополнительное сообщение, предлагающее пользователю изменить либо правила, либо результат. Становятся доступными ЭМ Обработка/Корректировка правил, Обработка/Корректировка результата.

10. После изменения параметров правил (Обработка/Корректировка правил) происходит возврат к ЭМ Обработка/Запуск правил и пп. 4–9 могут повториться.

11. После изменения результатов (Обработка/Корректировка результата) название соответствующей должности заменяется на «отказать» и доступными остаются ЭМ Обработка/Количественный результат, Обработка/Изменение результатов.

12. Если число принимаемых не превышает количества вакансий, то, в зависимости от поставленной цели, возможно обращение к ЭМ Обработка/Изменение результатов. Затем при нажатии кнопки «Далее» закрывается доступ к ЭМ пп. 6–9 и становится доступным ЭМ Показать/Принятые. Записи с должностями «отказать» уничтожаются, так как считается, что вторично лица, получившие отказ, не попадают в число претендентов. Статус «принимаемые» меняется на «принятые». Одновременно корректируется штатное расписание. На экране появляется список принятых.

13. Если список не вызывает сомнений, он нажатием кнопки «Обработка/Принять всех» закрывается. При этом статус «принятые» сменится на «работающие». На этом заканчивается очередной цикл работы в данном сеансе. Делается доступным ЭМ Обработка/Продолжить.

14. Нажатие его может вызвать 2 результата.

15. Если этот цикл не последний, то во всех таблицах интервал времени (цикл) автоматически меняется на следующий и появляется доступ к ЭМ Показать/Уволенные.

16. При его нажатии первоначально вызывается таблица Кадры (работающие), в которой вручную (по указанию преподавателя) для уволившихся специалистов устанавливается статус «уволен» Список-таблицу уволенных (до 20 % от состава) можно просмотреть. После закрытия списка в таблицу Кадры добавляется новый список претендентов и осуществляется переход к ЭМ Показать/Штатное расписание. Затем, начиная с п. 3, цикл повторяется.

17. Если номер текущего цикла равен количеству циклов, то работа программы заканчивается и пользователь извещается об этом. Демонстрируются результирующие таблицы Кадры и Штатное расписание и осуществляется выход из базы данных.

Отметим, что в приведенном примере пользователь один. Можно себе представить, что на крупном предприятии прием на разные должности осуществляют разные лица. Тогда требуется реализовать сетевой вариант, что в рамках СУБД InterBase затруднительно. К тому же накопленный авторами опыт показал, что эта СУБД имеет серьезные ограничения по размерности.

В связи с этим особый интерес представляет *класс 9*, в котором СУБД Denver (в рамках среды Apache) сочетается со стандартным алгоритмом .Net Solver Foundation, а работа осуществляется в среде разработки Visual Studio с применением языка C#. Класс позволяет к тому же использовать стандартный оптимизационный алгоритм на языке Java.

Этот класс интересен еще тем, что позволяет перейти к сетевому варианту [4].

При переходе к сетевому локальному варианту системы дополнительные порой противоречивые требования предъявляются разработчиками, пользователями, администраторами. Если подвести общий итог, то выдвигаются следующие требования. Система должна:

– быть достаточно простой по структуре;

– обладать широкими функциональными возможностями и высокой производительностью;

– быть надежной и удобной в функционировании при наличии хорошей эксплуатационной документации;

– быть достаточно дешевой в разработке и эксплуатации;

– оставаться современной в течение возможно более длительного промежутка времени, в том числе за счет простоты модернизации.

Эти требования достигаются посредством блочного построения системы с установлением связей между структурными элементами.

Таблица 3

Способ	Суть
Web	Вводится понятие перемещаемого ресурса. Ресурсом может быть что угодно. Web поддерживает несколько очень простых технологий
Модель «клиент-сервер»	Парадигма клиент-серверной архитектуры: ряд клиентов и серверов совместно с промежуточным программным обеспечением и средой взаимодействия составляют единую систему для распределенных вычислений и представления данных
Объектные системы	Объект – расширение известного механизма передачи управления и данных внутри программы одной машины на передачу управления и данных через сеть на другую машину. Удаленный объект представляет собой данные, определяющие его состояние. Состояние можно менять вызовом методов. Процесс разработки приложения с использованием технологии CORBA состоит из следующих этапов: 1. Определение интерфейса на IDL. 2. Обработка IDL для создания кода заглушки и скелетона. 3. Создание кода реализации объекта (сервер). 4. Построение кода использования данного объекта (клиент)
Агентные технологии	Агент – автономный процесс, способный реагировать на среду исполнения и вызывать изменения в ней, возможно, во взаимодействии с пользователями или другими агентами
Компонентные системы	Программная система – набор многократно используемых программных компонентов с четко определенным открытым интерфейсом, изменения в систему которых вносятся созданием новых компонентов или изменением старых
Сервис-ориентированная архитектура	Сервис-ориентированная архитектура (service-oriented architecture, SOA) – подход к созданию ИС, основанный на использовании сервисов или служб (service) с единым механизмом взаимодействия на основе концепции свободных связей и поддержки формальных интерфейсов
Web-сервисы	Web-сервис определяется как сервис (услуга), который предоставляется через WWW с использованием языка XML и протокола HTTP
Технологии одноранговых сетей	Разделение вычислительных ресурсов и сервисов производится напрямую между участниками сети, без использования центрального сервера. Каждый узел может быть как клиентом, так и сервером, предоставлять или использовать ресурсы сети
Технологии Грид	Грид – система, которая координирует распределенные ресурсы с помощью стандартных, открытых, универсальных протоколов и интерфейсов. Концепция грид-вычислений похожа на концепцию электросети. Можно запустить на вычисление любую задачу с любого компьютера, ресурсы для вычисления должны быть автоматически предоставлены на удаленных высокопроизводительных серверах
Облачные вычисления	«Облачные вычисления» – не только предложения услуг через Интернет, но и аппаратные средства и программные системы (SaaS) в центрах обработки данных. Облако – аппаратное и программное обеспечение центра обработки данных

При использовании сети применяются сервисы, которые на основе запроса на предоставление данных возвращают ответ. Под сервисом, как известно, понимается сетевая сущность, предоставляющая определенные функциональные возможности.

Имеется ряд технологий, обеспечивающих создание и сопровождение сервисов в распределенных вычислительных системах. Основными из них являются:

– клиент-сервер;
– объектные системы;
– агентные технологии;
– компонентные системы;
– сервис-ориентированная архитектура;
– web-сервисы;
– технологии одноранговых сетей;
– технологии Грид;
– облачные вычисления.

Суть технологий отражена в табл. 3 (Технологии распределенной структуры).

Сравнительные характеристики технологий распределенной структуры приведены в табл. 4.

Из табл. 3 видно, что здесь отражены системы как с одноранговой архитектурой, так и архитектурой «клиент-сервер».

Четыре последние технологии достаточно сложны для рассматриваемых локальных систем и более подходят для крупномасштабных систем. Агентные технологии ограничены в применении в связи с потенциальным «шпионским» характером. Web-сервисная и сервис-ориентированная архитектуры более подходят для глобальных систем.

Определенное предпочтение следует отдать технологии «клиент-сервер» в объектно-ориентированном варианте. В то же время такая реализация часто связана с недостаточной размерностью СУБД. Объектные и компонентные системы требуют формирования интерфейсов, что может оказаться непростой задачей. Предпочтительной с

Таблица 4

Способ	Достоинства	Недостатки
Web	Простота структуры	Сложность обеспечения безопасности данных. Применение в основном для поисковых систем
Модель «клиент-сервер»	Простота структуры. Возможность применения в системах управления	Трудность определения безопасности. Применение для систем относительно небольшой размерности. При росте количества клиентов повышаются требования к мощности сервера и пропускной способности канала. Устойчивая работа всех клиентов зависит от загруженности и функционирования одного сервера
Объектные системы	Простота разработки распределенных приложений в сравнении с клиент-серверным подходом. Возможность разработки приложений для гетерогенных вычислительных сред с помощью виртуальных машин и независимого описания интерфейсов взаимодействующих компонентов	Реализация удаленных вызовов гораздо сложнее реализации вызовов локальных процедур. Проблема передачи данных с адресного пространства с одной машины на другую. Неоднородность языков программирования и операционных сред: структуры данных и структуры вызова процедур. Строгая ограниченность данной технологии платформой Java. Ограниченность масштабируемости
Агентные технологии	Возможность доступа к разным точкам источника данных	Ряд серьезных проблем с безопасностью агентных платформ. Трудности определяет принцип работы технологии: любой администратор вычислительной сети не позволит агентам путешествовать по компьютерам пользователей
Компонентные системы	Пригодность для разработки крупных систем. Простая быстрая процедура инсталляции. Увеличение доли повторного применения кода. Снижение стоимости программного обеспечения. Унификация обработки объектов различной природы. Процесс создания программного обеспечения меньше зависит от человека	Высокие требования при разработке компонентов. Ограниченность технологии платформой Java
Сервис-ориентированная архитектура	Пригодность для разработки средних по размерам систем. Увеличение доли повторного применения кода. Снижение стоимости программного обеспечения. Унификация обработки сервисов различной природы	Недостаточная апробация программного обеспечения. Нет возможности работы с состояниями
Web-сервисы	Пригодность для разработки средних по размерам систем. Увеличение доли повторного применения кода. Снижение стоимости программного обеспечения. Унификация обработки сервисов различной природы. Обеспечение безопасности данных	Достаточно сложная архитектура
Технологии одноранговых сетей	Упрощается поддержка масштабируемости при значительном росте количества узлов в вычислительной сети. Повышается отказоустойчивость сети. Возможность объединения ресурсов отдельных участников сети при возможности предоставления услуг другим участникам. Отсутствие зависимости от централизованных сервисов и ресурсов. Пригодность для систем с большим объемом данных и обменом информацией между пользователями	1. Рост требований к производительности каждого компьютера. 2. Низкая степень защищенности машин из-за открытого доступа. 3. Трудность учета гетерогенности аппаратного и программного обеспечения потенциальных участников сети. 4. Сложность поиска доступных ресурсов без использования централизованной точки управления. 5. Затруднения в вопросах безопасности предоставления ресурсов в основном индивидуальным пользователям

Окончание табл. 4

Способ	Достоинства	Недостатки
Технологии Грид	Позволила объединить крупные комплексы обработки и хранения данных, используя следующие свойства: 1. Гетерогенность. 2. Масштабируемость. 3. Приспособляемость	Сложная инфраструктура, которую целесообразно применять для сверхбольших систем
Облачные вычисления	Сокращение в 3–5 раз стоимости бизнес-приложений и более чем в 5 раз стоимости приложений для конечных пользователей. Оплата по мере использования ресурса. Масштабируемость и гибкость. Высокий уровень загрузки оборудования и снижение затрат на его эксплуатацию	Сложности организации репликации данных между сервисами. Сложность администрирования в процедуре функционирования. Сложность архитектуры. Высокие риски в защите информации

позиций управления представляется компонентная архитектура, более похожая на объектно-ориентированный вариант.

Рассмотрены классы компьютерной реализации отдельных структурных элементов многоуровневой системы управления. В технологии «клиент-сервер» перечисленным ранее требованиям в наибольшей мере удовлетворил класс СУБД InterBase в среде C++. Приведена реализация этого класса. При переходе к многоэлементной системе потребуется наличие сетевого режима, что в рамках СУБД InterBase связано с большими трудностями. К тому же выяснилось, что эта СУБД ограничена по размерности и более подходит для систем средней размерности.

В связи с этим целесообразно использовать связку СУБД Denver – среда Apache, свободную от указанных недостатков. Отмечено, что связка обладает хорошими сетевыми свойствами.

СПИСОК ЛИТЕРАТУРЫ

1. Чертовской В. Д. Интеллектуализация автоматизированного управления производством. СПб.: Изд-во С.-Петерб. ун-та, 2007. 164 с.

2. Косяков М. С. Введение в распределенные вычисления. СПб.: Изд-во НИУ ИТМО, 2014. 155 с.

3. Подкур М. Л., Подкур П. Н., Смоленцев Н. К. Программирование в среде Borland C++ Builder с математическими библиотеками MatLab C/C++. М.: ДМК Пресс, 2006. 496 с.

4. Радченко Г. И. Распределенные вычислительные системы. Челябинск: Фотохудожник, 2012. 184 с.

V. V. Tsehanovsky, V. D. Chertovskoy
Saint Petersburg Electrotechnical University «LETI»

PROGRAM REALIZATION OF HIERARCHICAL CONTROL SYSTEM

Classes of realization of a separate structural element are considered. Nine classes of combinations of the DBMS, algorithms, languages and the environments of implementation are considered. Classes are implemented and approved on the task of static linear programming for the purpose of comparative detection of their opportunities. On the formulated requirements the class InterBase is selected from the environment C ++ by Builder. The application-oriented example of realization for control system of employment is given. Class InterBase shortcomings of the environment C ++ are revealed by Builder in case of its use in the network mode. A row of the technologies providing creation and attending of services in the distributed computing systems is selected. Upon transition to multielement system additional requirements to implementation methods on the basis of which the analysis of network architectures is made came to light. The preference from line items of control of the component architecture similar to object-oriented option is shown. The network mode is provided in a linking of Denver DBMS - the environment Apach, to realize processes in the most widespread three-level management system.

Hierarchy control system, classes realization, separate element, system of elements