В.Г. МАТВЕЙКИН, Б.С. ДМИТРИЕВСКИЙ, Н.Р. ЛЯПИН

ИНФОРМАЦИОННЫЕ СИСТЕМЫ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА

Москва
«Машиностроение»
2008

УДК 004.8
ББК 381
М336

Р е ц е н з е н т ы :

Доктор технических наук, профессор
Ярославского государственного технического университета
М.П. Цыганков

Доктор технических наук, профессор, ректор Тверского государственного технического университета
Б.В. Палюх

Матвейкин В.Г., Дмитриевский Б.С., Ляпин Н.Р.

М336 Информационные системы интеллектуального анализа. – М.: Машиностроение, 2008. – 92 с.; ил.

ISBN 978-5-94275-415-0

Разработана система поддержки интеллектуального анализа выполнения бизнес-процессов в электронном документообороте, которая предоставляет аналитику средства выявления глобальных зависимостей, средства обнаружения критических, загруженных инцидентных задач, средства прогноза завершения бизнес-процесса. Исследованы вопросы адаптации ассоциативных алгоритмов поиска частых подпоследовательностей для анализа выполнения бизнес-процессов, предложены новые, более продуктивные алгоритмы.

Для инженеров, научных работников и аспирантов.

УДК 004.8

ББК 3811

ISBN 978-5-94275-415-0

Научное издание

Матвейкин Валерий Григорьевич,
Дмитриевский Борис Сергеевич,
Ляпин Никита Романович

ИНФОРМАЦИОННЫЕ СИСТЕМЫ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА

Редактор Т.М. Глинкина
Инженер по компьютерному макетированию М.А. Филатова
Корректор О.М. Ярцева

Сдано в набор 18.08.2008 г. Подписано в печать 1.10.2008 г.
Формат 60×84/16. Бумага офсетная. Гарнитура Times New Roman.
Печать офсетная. Усл. печ. л. 5,35. Уч.-изд. л. 5,3.
Тираж 400 экз. Заказ 434

ООО "Издательство Машиностроение", 107076, Москва, Стромынский пер., 4

Подготовлено к печати и отпечатано в Издательско-полиграфическом центре
Тамбовского государственного технического университета
392000, Тамбов, Советская, 106, к. 14

По вопросам приобретения книги обращаться по телефону 8(4752)638108

ВВЕДЕНИЕ

По мере распространения информационных технологий увеличиваются объемы хранимой в базах данных информации, что приводит к развитию методов интеллектуального анализа данных – дисциплины, изучающей процесс нахождения новых, действительных и потенциально полезных знаний в базах данных. Интеллектуальный анализ (ИА) лежит на пересечении нескольких наук, главные из которых – это системы баз данных, статистика и искусственный интеллект.

Область интеллектуального анализа выросла из одного семинара в 1989 г. до десятков международных конференций в 2003 г. с тысячами исследователей во многих странах мира. Интеллектуальный анализ широко используется во многих областях с большим объемом данных. В науке – астрономии, биологии, биоинформатике, медицине, физике и других областях. В бизнесе – торговле, телекоммуникациях, банковском деле, промышленном производстве и т.д. Благодаря сети Интернет интеллектуальный анализ используется каждый день тысячи раз в секунду – каждый раз, когда кто-то использует поисковые системы для своих нужд.

Виды информации, с которыми работают исследователи, включают не только цифровые данные, но и все более текст, изображение, видео, звук и т.д. Новая и быстро растущая часть интеллектуального анализа – анализ связей между данными (link analysis) – имеет приложения в таких разных областях, как биоинформатика, цифровые библиотеки и т.п.

Математические и статистический подходы являются основой для ИА. Инструментарий ИА включает:

- Методы теории вероятностей и математической статистики.
- Регрессионный, дискриминантный анализ.
- Факторный, кластерный анализ.
- Карты восприятия, методы выявления логических закономерностей.
- Метод дерева решений.
- Методы теории нечетких множеств.
- Методы теории полезности.
- Метод анализа иерархий.
- Система сбалансированных показателей.
- Нейронные сети.

Под интеллектуальными (интеллект от лат. Intellectus – ум, рассудок, разум) методами подразумеваются такие способы решения задач, в основе которых лежат алгоритмы и действия, в большей или меньшей степени связанные с интеллектуальной деятельностью человека, его эволюцией, повседневным поведением.

С другой стороны, в современных организациях непрерывное развитие может быть увидено в перманентном изменении услуг и производимых товаров. Это требует все более действенного и эффективного организационного и производственного окружения. Одним из подтверждений подобных тенденций является распространение систем менеджмента качества на базе стандарта ИСО 9000 и таких методологий, как Continuous Process Improvement (CPI). Эти подходы требуют наличия инструментов моделирования бизнес-процессов, анализа их выполнения, средств контроля и документирования. При этом крайне важна возможность непрерывного улучшения и внесения изменений в их структуру. Значительным образом способствуют решению этих задач системы управления электронным документооборотом, такие, как «ДокМенеджер», «DocsVision», «Documentum» и др. Бизнес-процессы в них должны быть построены соответствующим образом, желательно с использованием научных методов. Основная проблема для сотрудников, ответственных за разработку бизнес-процессов, – осуществление действенного и эффективного дизайна и контроля бизнес-процессов.

Поскольку в современной организации значительное количество информации о бизнес-процессах записывается и хранится в электронном виде, подобные данные представляются полезными для получения актуальной картины, происходящей с бизнес-процессами. Решить эту задачу позволяют адаптированные методы ИА, чему уделяется особое внимание в этой монографии.

Разрабатывая систему поддержки бизнес-процесса, аналитику необходимо построить детальную модель, безошибочно описывающую реальный бизнес-процесс. Моделирование бизнес-процесса без использования системы интеллектуального анализа – сложная задача, требующая всестороннего знания процесса (общение с сотрудниками и менеджерами – участниками бизнес-процесса), что занимает много времени, и часто результаты получаются весьма субъективными. Именно поэтому в современной научной литературе возрастает интерес к решению этой проблемы в большей степени средствами ИА.

Глава 1. ИНТЕЛЛЕКТУАЛЬНЫЙ АНАЛИЗ ДАННЫХ

При использовании OLAP-систем [1] аналитику предоставляются средства проверки гипотез при анализе данных. При этом основной задачей аналитика является генерация гипотез. Он решает ее, основываясь на своих знаниях и опыте. Однако знания есть не только у человека, но и в накопленных данных, которые подвергаются анализу. Такие знания часто называют «скрытыми», так как они содержатся в гигабайтах и терабайтах информации, которые человек не в состоянии исследовать самостоятельно. В связи с этим существует высокая вероятность пропустить гипотезы, которые могут принести значительную выгоду [2].

Очевидно, что для обнаружения скрытых знаний необходимо применять специальные методы автоматического анализа, при помощи которых приходится практически добывать знания. За этим направлением прочно закрепился термин интеллектуального анализа данных. Классическое определение этого термина дал в 1996 г. один из основателей этого направления Пятецкий-Шапиро [1].

Интеллектуальный анализ данных – исследование и обнаружение «машиной» (алгоритмами, средствами искусственного интеллекта) в сырых данных скрытых знаний, которые ранее не были известны, нетривиальны, практически полезны, доступны для интерпретации человеком.

Рассмотрим свойства обнаруживаемых знаний, данные в определении, более подробно [2].

Знания должны быть новые, ранее неизвестные. Затраченные усилия на открытие знаний, которые уже известны пользователю, не окупаются. Поэтому ценность представляют именно новые, ранее неизвестные знания.

Знания должны быть нетривиальны. Результаты анализа должны отражать неочевидные, неожиданные закономерности в данных, составляющие так называемые скрытые знания. Результаты, которые могли бы быть получены более простыми способами (например, визуальным просмотром), не оправдывают привлечение мощных средств ИА.

Знания должны быть практически полезны. Найденные знания должны быть применимы, в том числе и на новых данных, с достаточно высокой степенью достоверности. Полезность заключается в том, чтобы эти знания могли принести определенную выгоду при их применении.

Знания должны быть доступны для понимания человеку. Найденные закономерности должны быть логически объяснимы, в противном случае существует вероятность, что они являются случайными. Кроме того, обнаруженные знания представляются в понятном для человека виде.

В ИА для представления полученных знаний служат модели. Виды моделей зависят от методов их создания. Наиболее распространенными являются: правила, деревья решений, кластеры и математические функции.

1.1. КЛАССИФИКАЦИЯ ЗАДАЧ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА

Методы ИА помогают решить многие задачи, с которыми сталкивается аналитик. Из них основными являются: классификация, регрессия, поиск ассоциативных правил и кластеризация [3]. Ниже приведено краткое описание основных задач анализа данных.

Задача классификации сводится к определению класса объекта по его характеристикам. Необходимо заметить, что в этой задаче множество классов, к которым может быть отнесен объект, заранее известно.

Задача регрессии, подобно задаче классификации, позволяет определить по известным характеристикам объекта значение некоторого его параметра. В отличие от задачи классификации значением параметра является не конечное множество классов, а множество действительных чисел.

При поиске ассоциативных правил целью является нахождение частых зависимостей (или ассоциаций) между объектами или событиями. Найденные зависимости представляются в виде правил и могут быть использованы как для лучшего понимания природы анализируемых данных, так и для предсказания появления событий.

Задача кластеризации заключается в поиске независимых групп (кластеров) и их характеристик во всем множестве анализируемых данных. Решение этой задачи помогает лучше понять данные. Кроме того, группировка однородных объектов позволяет сократить их число, а следовательно, и облегчить анализ.

Перечисленные задачи по назначению делятся на описательные и предсказательные.

Описательные задачи уделяют внимание улучшению понимания анализируемых данных. Ключевой момент в таких моделях – легкость и прозрачность результатов для восприятия человеком. Возможно, обнаруженные закономерности будут специфической чертой именно конкретных исследуемых данных и больше нигде не встретятся, но это все равно может быть полезно и потому должно быть известно. К такому виду задач относятся кластеризация и поиск ассоциативных правил.

Решение предсказательных задач разбивается на два этапа. На первом этапе на основании набора данных с известными результатами строится модель. На втором этапе она используется для предсказания результатов на основании новых наборов данных. При этом, естественно, требуется, чтобы построенные модели работали максимально точно. К данному виду задач относят задачи классификации и регрессии. Сюда можно отнести и задачу поиска ассоциативных правил, если результаты ее решения могут быть использованы для предсказания появления некоторых событий.

По способам решения задачи разделяют на обучение с учителем и обучение без учителя. Такое название произошло от термина «машинное обучение», часто используемого в англоязычной литературе и обозначающего все технологии ИА.

В случае обучения с учителем задача анализа данных решается в несколько этапов. Сначала с помощью какого-либо алгоритма ИА строится модель анализируемых данных – классификатор. Затем классификатор подвергается обучению. Другими словами, проверяется качество его работы и, если оно неудовлетворительно, происходит дополнительное обучение классификатора. Так продолжается до тех пор, пока не будет достигнут требуемых уровень качества или не станет ясно, что выбранный алгоритм не работает корректно с данными, либо же сами данные не имеют структуры, которую можно выявить. К этому типу задач относят задачи классификации и регрессии.

Обучение без учителя объединяет задачи, выявляющие описательные модели, например закономерности в покупках, совершаемых клиентами большого магазина. Очевидно, что если эти закономерности есть, то модель должна их представить и неуместно говорить об ее обучении. Достоинством таких задач является возможность их решения без каких-либо предварительных знаний об анализируемых данных. К ним относятся кластеризация и поиск ассоциативных правил.

Задачи классификации и регрессии. При анализе часто требуется определить, к какому из известных классов относятся исследуемые объекты, т.е. классифицировать их. В общем случае количество классов в задачах классификации может быть более двух. В интеллектуальном анализе задачу классификации рассматривают как задачу определения значения одного из параметров анализируемого объекта на основании значения других параметров [4]. Определяемый параметр часто называют зависимой переменной, а параметры, участвующие в его определении, – независимыми переменными.

Если значениями независимых и зависимой переменных являются действительные числа, то задача называется задачей регрессии. Задача классификации и регрессии решается в два этапа. На первом выделяется обучающая выборка. В нее входят объекты, для которых известны значения как независимых, так и зависимых переменных.

На основании обучающей выборки строится модель определения значения зависимой переменной. Ее часто называют функцией классификации или регрессии. Для получения максимально точной функции к обучающей выборке предъявляются следующие основные требования:

– количество объектов, входящих в выборку, должно быть достаточно большим. Чем больше объектов, тем построенная на ее основе функция классификации или регрессии будет точнее;

– в выборку должны входить объекты, представляющие все возможные классы в случае задачи классификации или всю область значений в случае задачи регрессии;

– для каждого класса в задаче классификации или каждого интервала области значений в задаче регрессии выборка должна содержать достаточное количество объектов.

На втором этапе построенную модель применяют к анализируемым объектам (к объектам с неопределенным значением зависимой переменной). Основные проблемы, с которыми сталкиваются при решении задач классификации и регрессии, – это неудовлетворительное качество исходных данных, в которых встречаются как ошибочные данные, так и пропущенные значения, различные типы атрибутов – числовые и категоричные, разная значимость атрибутов, а также так называемые overfitting и underfitting. Суть первой из них заключается в том, что классификационная функция при построении «слишком хорошо» адаптируется к данным, и встречающиеся в них ошибки и аномальные значения пытается интерпретировать как часть внутренней структуры данных. Очевидно, что такая модель будет некорректно работать в дальнейшем с другими данными, где характер ошибок будет несколько иной. Термином underfitting обозначают ситуацию, когда слишком велико количество ошибок при проверке классификатора на обучающем множестве. Это означает, что особых закономерностей в данных не было обнаружено и либо их нет вообще, либо необходимо выбрать иной метод обнаружения.

Задача поиска ассоциативных правил. Поиск ассоциативных правил является одним из самых популярных приложений ИА. Суть задачи заключается в определении часто встречающихся наборов объектов в большом множестве таких наборов. Данная задача является частным случаем задачи классификации [5]. Первоначально она решалась при анализе тенденций в поведении покупателей в супермаркетах. Анализу подвергались данные о совершаемых ими покупках, которые покупатели складывают в корзину. Это послужило причиной второго часто встречающегося названия – анализ рыночных корзин. При анализе этих данных интерес прежде всего представляет информация о том, какие товары предпочитают, в какие периоды времени и т.п. Такая информация позволяет более эффективно планировать закупку товаров, проведение рекламной кампании и т.д.

Задача поиска ассоциативных правил актуальна не только в сфере торговли. Например, в сфере обслуживания интерес представляет, какими услугами клиенты предпочитают пользоваться в совокупности. Для получения этой информации задача решается применительно к данным об услугах, которыми пользуется один клиент в течение определенного времени (месяца, года).

В медицине анализу могут подвергаться симптомы и болезни, наблюдаемые у пациентов. В этом случае знания о том, какие сочетания болезней и симптомов встречаются наиболее часто, помогают в будущем правильно ставить диагноз.

При анализе часто вызывает интерес последовательность производящих событий. При обнаружении закономерностей в таких последовательностях можно с некоторой долей вероятности предсказывать появление событий в будущем, что позволяет принимать более правильные решения. Такая задача является разновидностью задачи поиска ассоциативных правил и называется сиквенциальным анализом.

Основным отличием задачи сиквенциального анализа от поиска ассоциативных правил является установление отношения порядка между исследуемыми наборами. Данное отношение может быть определено разными способами. При анализе последовательности событий, происходящих во времени, объектами таких наборов являются события, а отношение порядка соответствует хронологии их появления.

Сиквенциальный анализ широко используется, например в телекоммуникационных компаниях, для анализа данных об авариях на различных узлах сети. Информация о последовательности совершения аварий может помочь в обнаружении неполадок и предупреждении новых аварий. Например, если известна последовательность сбоев:

$$\{e_5, e_2, e_7, e_{13}, e_6, e_1, ...\},$$

где e_i – сбой с кодом i, то на основании факта появления сбоя e_2 можно сделать вывод о скором появлении сбоя e_7. Зная это, можно предпринять профилактические меры, устраняющие причины возникновения сбоя. Если дополнительно обладать и знаниями о времени между сбоями, то можно предсказать не только факт его появления, но и время, что часто не менее важно.

Задача кластеризации. Задача кластеризации состоит в разделении исследуемого множества объектов на группы «похожих» объектов, называемых кластерами. Слово кластер английского происхождения (cluster), переводится как сгусток, пучок, группа. Родственные понятия, используемые в литературе, – класс, таксон, сгущение. Часто решение задачи разбиения множества элементов на кластеры называют кластерным анализом [5].

Кластеризация может применятся практически в любой области, где необходимо исследование экспериментальных или статистических данных.

Для научных исследований изучение результатов кластеризации, а именно вычисление причин, по которым объекты объединяются в группы, способно открыть новые перспективные направления.

Кластеризация отличается от классификации тем, что для проведения анализа не требуется иметь выделенную зависимую переменную. С этой точки зрения она относится к классу обучения без учителя. Задача решается на начальных этапах исследования, когда о данных мало что известно. Ее решение помогает лучше понять данные, и с этой точки зрения задача кластеризации является описательной задачей.

Для задачи кластеризации характерно отсутствие каких-либо различий как между переменными, так и между объектами. Напротив, ищутся группы наиболее близких, похожих объектов. Методы автоматического разбиения на кластеры редко используются сами по себе, просто для получения групп схожих объектов. После определения кластеров применяются другие методы ИА, для того чтобы попытаться установить, что означает такое разбиение, чем оно вызвано.

Кластерный анализ позволяет рассматривать достаточно большой объем информации и резко сокращать, сжимать большие массивы информации, делать их компактными и наглядными.

Отметим ряд особенностей, присущих задаче кластеризации.

Во-первых, решение сильно зависит от природы объектов данных (и их атрибутов). Так, с одной стороны, это могут быть однозначно определенные, четко количественно очерченные объекты, а с другой – объекты, имеющие вероятностное или нечеткое описание.

Во-вторых, решение значительно зависит также и от представления кластеров и предполагаемых отношений объектов данных и кластеров. Так, необходимо учитывать такие свойства, как возможность/невозможность принадлежности объектов

нескольким кластерам. Необходимо определение самого понятия принадлежности кластеру: однозначная (принадлежит / не принадлежит), вероятностная (вероятность принадлежности), нечеткая (степень принадлежности).

1.2. ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА

Интернет-технологии. В системах электронного бизнеса, где особую важность имеют вопросы привлечения и удержания клиентов, технологии ИА часто применяются для построении рекомендательных систем Интернет-магазинов и для решения проблемы персонализации посетителей Web-сайтов. Рекомендации товаров и услуг, построенные на основе закономерностей в покупках клиентов, обладают огромной убеждающей силой. Статистика показывает, что почти каждый посетитель магазина Amazon не упускает возможности посмотреть на то, что покупали другие посетители. Персонализация клиентов, другими словами, автоматическое распознавание принадлежности клиента к определенной целевой аудитории позволяет компании проводить более гибкую маркетинговую политику. Поскольку в электронной коммерции деньги и платежные системы также электронные, то важной задачей становится обеспечение безопасности при операциях с пластиковыми карточками. ИА позволяет обнаруживать случаи мошенничества.

Торговля. Для успешного продвижения товаров всегда важно знать, что и как продается, а также кто является потребителем. Исчерпывающий ответ на первый вопрос дают такие средства ИА, как анализ рыночных корзин и сиквенциальный анализ. Зная связи между покупками и временные закономерности, можно оптимальным образом регулировать предложение. С другой стороны, маркетинг имеет возможность непосредственно управлять спросом, но для этого необходимо знать как можно больше о потребителях – целевой аудитории маркетинга. ИА позволяет решать задачи выделения групп потребителей со схожими стереотипами поведения, т.е. сегментировать рынок. Для этого можно применять такие технологии ИА, как кластеризацию и классификацию.

Сиквенциальный анализ помогает торговым предприятиям принимать решения о создании товарных запасов. Он дает ответы на вопросы типа «Если сегодня покупатель приобрел видеокамеру, то через какое время он вероятнее всего купит новые батарейки и пленку?»

Промышленное производство. Промышленное производство создает идеальные условия для применения технологий ИА. Причина – в самой природе технологического процесса, который должен быть воспроизводимым и контролируемым. Все отклонения в течение процесса, влияющие на качество выходного результата, также находятся в заранее известных пределах. Таким образом, создается статистическая стабильность, первостепенную важность которой отмечают в работах по классификации.

Примером использования ИА в промышленности может быть прогнозирование качества изделия в зависимости от замеряемых параметров технологического процесса.

Медицина. В медицинских и биологических исследованиях, равно как и в практической медицине, спектр решаемых задач настолько широк, что возможно использование любых методологий ИА. Примером может служить построение диагностической системы или исследование эффективности хирургического вмешательства.

Известно много экспертных систем для постановки медицинских диагнозов [6]. Они построены, главным образом, на основе правил, описывающих сочетания различных симптомов отдельных заболеваний. С помощью таких правил узнают не только, чем болен пациент, но и как нужно его лечить. Правила позволяют выбирать средства медикаментозного воздействия, определять показания/противопоказания, ориентироваться в лечебных процедурах, создавать условия наиболее эффективного лечения, предсказывать исходы назначенного курса лечения и т.п. Технологии ИА позволяют обнаруживать в медицинских данных шаблоны, составляющие основу указанных правил.

Одним из наиболее передовых направлений медицины является биоинформатика – область науки, разрабатывающая и применяющая вычислительные алгоритмы для анализа и систематизации генетической информации с целью выявления структуры и функции макромолекул, последующего использования этих знаний для объяснения различных биологических явлений и создания новых лекарственных препаратов. Объектом исследования биоинформатики являются огромные объемы информации о последовательности ДНК и первичной структуре белков, появившиеся в результате изучения структуры геномов микроорганизмов, млекопитающих и человека. Абстрагируясь от конкретного содержания этой информации, ее можно рассматривать как набор генетических текстов, состоящих из протяженных символьных последовательностей. Выявление структурных закономерностей в таких последовательностях входит в число задач, эффективно решаемых средствами ИА, например с помощью сиквенциального и ассоциативного анализа. Основная область практического применения биоинформатики – это разработка лекарств нового поколения.

Банковское дело. Классическим примером использования ИА на практике является решение проблемы о возможной некредитоспособности клиентов банка. Этот вопрос, тревожащий любого сотрудника кредитного отдела банка, можно решить и интуитивно. Если образ клиента в сознании банковского служащего соответствует его представлению о кредитоспособном клиенте, то кредит выдавать можно, иначе – отказать. По схожей схеме, но более продуктивно и полностью автоматически работают установленные во многих банках системы поддержки принятия решений со встроенной функциональностью ИА. Лишенные субъективной предвзятости, они опираются в своей работе только на историческую базу данных банка, где записывается детальная информация о каждом клиенте и в конечном итоге факт его кредитоспособности. Классификационные алгоритмы ИА обрабатывают эти данные, и полученные результаты используются далее для принятия решений.

Анализ кредитного риска заключается, прежде всего, в оценке кредитоспособности заемщика. Эта задача решается на основе анализа накопленной информации, т.е. кредитной истории «прошлых» клиентов. С помощью инструментов ИА (деревья решений, кластерный анализ, нейронные сети и др.) банк может получить профили добросовестных и неблагонадежных заемщиков. Кроме того, возможно классифицировать заемщика по группам риска, а значит, не только решить вопрос о возможности кредитования, но и установить лимит кредита, проценты по нему и срок возврата.

Мошенничество с кредитными карточками представляет собой серьезную проблему, т.е. убытки от него измеряются миллионами ежегодно, а рост количества мошеннических операций составляет по оценкам экспертов от 15 до 25 % ежегодно.

При решении этой задачи технология ИА предоставляет стереотипы подозрительных операций, созданные в результате анализа огромного количества транзакций – как законных, так и неправомерных. Исследуется не только отдельно взятая операция, но и совокупность последовательных во времени транзакций. Кроме того, алгоритмы и модели (например, нейронные сети), имеющиеся в составе продуктов ИА, способны тестироваться и самообучаться. При попытке совершения подозрительной операции средства ИА оперативно выдают предупреждение об этом, что позволяет банку предотвратить незаконные действия, а не устранять их последствия. Использование технологий ИА позволяется сократить число нарушений на 20 – 30 %.

Страховой бизнес. В страховании, так же как в банковском деле и маркетинге, возникает задача обработки больших объемов информации для определения типичных групп (профилей) клиентов. Эта информация используется для того, чтобы предлагать определенные услуги страхования с наименьшим для компании риском и, возможно, с пользой для клиента. Также с помощью технологий ИА решается такая часто встречающаяся в страховании задача, как определение случаев мошенничества.

Другие области применения. ИА может применяться практически везде, где возникает задача автоматического анализа данных. В качестве примера приведем такие популярные направления, как анализ и последующая фильтрация спама, а также разработка так называемых виртуальных собеседников. Последние сейчас являются не более чем экзотическим дополнением к интерфейсу некоторых сайтов, но предполагается, что в будущем они могут заменить собой call-центры компаний.

1.3. МОДЕЛИ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА

Предсказательные модели. Предсказательные модели строятся на основании набора данных с известными результатами. Они используются для предсказания результатов на основании других наборов данных. При этом, естественно, требуется, чтобы модель работала максимально точно, была статистически значима и оправданна и т.д.

К ним относятся следующие модели:

– классификации – описывают правила или набор правил, в соответствии с которыми можно отнести описание любого нового объекта к одному из классов. Такие правила строятся на основании информации о существующих объектах путем разбиения их на классы;

– последовательностей – описывают функции, позволяющие прогнозировать изменение непрерывных числовых параметров. Они строятся на основании данных об изменении некоторого параметра за прошедший период времени.

Описательные модели. Описательные модели уделяют внимание сути зависимостей в наборе данных, взаимному влиянию различных факторов, т.е. построению эмпирических моделей различных систем. Ключевой момент в таких моделях – легкость и прозрачность для восприятия человеком. Возможно, обнаруженные закономерности будут специфической чертой именно конкретных исследуемых данных и больше нигде не встретятся, но это все равно может быть полезно и потому должно быть известно.

К ним относятся следующие виды моделей:

– Регрессионные – описывают функциональные зависимости между зависимыми и независимыми показателями и переменными в понятной человеку форме. Необходимо заметить, что такие модели описывают функционально зависимость не только между непрерывными числовыми параметрами, но и между категориальными.

– Кластеризации – описывают группы (кластеры), на которые можно разделить объекты, данные о которых подвергаются анализу. Группируются объекты (наблюдения, события) на основе данных (свойств), описывающих сущность объектов. Объекты внутри кластера должны быть «похожими» друг на друга и отличаться от объектов, вошедших в другие кластеры. Чем больше похожи объекты внутри кластера и чем больше отличий между кластерами, тем точнее кластеризация.

– Исключений – описывают исключительные ситуации в записях (например, отдельных пациентов), которые резко отличаются чем-либо от основного множества записей (группы больных). Знание исключений может быть использовано двояким образом. Возможно, что эти записи представляют собой случайный сбой, например ошибки операторов, вводивших данные в компьютер. Характерный случай: если оператор, ошибаясь, ставит десятичную точку не в том месте, то такая ошибка сразу дает резкий «всплеск» на порядок. Подобную «шумовую», случайную составляющую имеет смысл отбросить, исключить из дальнейших исследований, поскольку большинство методов, которые будут рассмотрены в данной главе, очень чувствительны к наличию «выбросов» – резко отличающихся точек, редких, нетипичных случаев. С другой стороны, отдельные, исключительные записи могут представлять самостоятельный интерес для исследования, так как они могут указывать на некоторые редкие, но важные аномальные заболевания. Даже сама идентификация этих записей, не говоря об их последующем анализе и детальном рассмотрении, может оказаться очень полезной для понимания сущности изучаемых объектов или явлений.

– Итоговые – выявление ограничений на данные анализируемого массива. Например, при изучении выборки данных по пациентам не старше 30 лет, перенесшим инфаркт миокарда, обнаруживается, что все пациенты, описанные в этой выборке, либо курят более 5 пачек сигарет в день, либо имеют вес не ниже 95 кг. Подобные ограничения важны для понимания данных массива; по сути дела – это новое знание, извлеченное в результате анализа. Таким образом, Data Summarization – это нахождение каких-либо фактов, которые верны для всех или почти всех записей в изучаемой выборке данных, но которые достаточно редко встречались бы во всем мыслимом многообразии записей такого же формата и, например, характеризовались бы теми же распределениями значений полей. Если взять для сравнения информацию по всем пациентам, то процент либо сильно курящих, либо чрезмерно тучных людей будет весьма невелик. Можно сказать, что решается как бы неявная задача классификации, хотя фактически задан только один класс, представленный имеющимися данными.

– Ассоциации – выявление закономерностей между связанными событиями. Примером такой закономерности служит правило, указывающее, что из события X следует событие Y. Такие правила называются ассоциативными.

Для построения рассмотренных моделей используются различные методы и алгоритмы ИА. Ввиду того что технология ИА развивалась и развивается на стыке таких дисциплин, как статистика, теория информации, машинное обучение, теория

баз данных, вполне закономерно, что большинство алгоритмов и методов ИА были разработаны на основе различных технологий и концепций. Рассмотрим технологии, наиболее часто реализуемые методами ИА.

1.4. МЕТОДЫ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА

Базовые методы. К базовым методам ИА принято относить, прежде всего, алгоритмы, основанные на переборе [7, 8]. Простой перебор всех исследуемых объектов требует $O(2^N)$ операций, где N – количество объектов. Следовательно, с увеличением количества данных объем вычислений растет экспоненциально, что при большом объеме делает решение любой задачи таким методом практически невозможным.

Для сокращения вычислительной сложности в таких алгоритмах, как правило, используют разного вида эвристики, приводящие к сокращению перебора. Оптимизация подобных алгоритмов сводится к приведению зависимости количества операций от количества исследуемых данных к функции линейного вида. В то же время зависимость от количества атрибутов, как правило, остается экспоненциальной. При условии, что их немного (в подавляющем большинстве случаев их значительно меньше, чем данных), такая зависимость является приемлемой.

Основным достоинством данных алгоритмов является их простота, как с точки зрения понимания, так и реализации. К недостаткам можно отнести отсутствие формальной теории, на основании которой строятся такие алгоритмы, а следовательно, сложности, связанные с их исследованием и развитием.

К базовым методам ИА можно отнести также и подходы, использующие элементы теории статистики. В связи с тем, что ИА является развитием статистики, таких методов достаточно много. Основная их идея сводится к корреляционному, регрессионному и другим видам статистического анализа. Основным недостатком является усреднение значений, что приводит к потере информативности данных.

Основным способом исследования задач анализа данных является их отображение на формализованный язык и последующий анализ полученной модели. Неопределенность по объему отсутствующей информации у системного аналитика можно разделить на три большие группы:

- неизвестность;
- неполнота (недостаточность, неадекватность);
- недостоверность.

Недостоверность бывает физической (источником ее является внешняя среда) и лингвистической (возникает в результате словесного обобщения и обусловливается необходимостью описания бесконечного числа ситуаций ограниченным числом слов за ограниченное время).

Выделяют два вида физической неопределенности:

- неточность (неточность измерений значений определенной величины, выполняемых физическими приборами);
- случайность (или наличие во внешней среде нескольких возможностей, каждая из которых случайным образом может стать действительностью; предполагается знание соответствующего закона распределения вероятностей).

Выделяют два вида лингвистической неопределенности:

- неопределенность значений слов (многозначность, расплывчатость, неясность, нечеткость). Она возникает в случае, если отображаемые одним и тем же словом объекты задачи управления различны;
- неоднозначность смысла фраз (выделяют синтаксическую и семантическую).

Для обработки физических неопределенностей успешно используются методы теории вероятностей и классическая теория множеств. Однако с развитием систем, использующих методы теории искусственного интеллекта, в которых требуется обрабатывать понятия и отношения естественного языка, возникла необходимость расширения множества формальных методов с целью учета лингвистической неопределенности задач.

Генетические алгоритмы. Генетические алгоритмы (ГА) относятся к числу универсальных методов оптимизации, позволяющих решать задачи различных типов (комбинаторные, общие задачи с ограничениями и без ограничений) и различной степени сложности. При этом ГА характеризуются возможностью как однокритериального, так и многокритериального поиска в большом пространстве, ландшафт которого является негладким [8].

В последние годы резко возросло число работ, прежде всего зарубежных ученых, посвященных развитию теории ГА и вопросам их практического использования. Результаты данных исследований показывают в частности, что ГА могут получить более широкое распространение при интеграции с другими методами и технологиями. Появились работы, в которых доказывается эффективность интеграции ГА и методов теории нечеткости, а также нейронных вычислений и систем.

Нейронные сети. Нейронные сети – это класс моделей, основанных на биологической аналогии с мозгом человека и предназначенных после прохождения этапа так называемого обучения на имеющихся данных для решения разнообразных задач анализа данных. При применении этих методов прежде всего встает вопрос выбора конкретной архитектуры сети (числа «слоев» и количества «нейронов» в каждом из них).

1.5. ПРОЦЕСС ОБНАРУЖЕНИЯ ЗНАНИЙ

Основные этапы анализа. Для обнаружения знаний в данных недостаточно просто применить методы ИА, хотя, безусловно, этот этап является основным в процессе анализа. Весь процесс состоит из нескольких этапов. Рассмотрим основные из них, чтобы продемонстрировать, что без специальной подготовки аналитика методы ИА сами по себе не решают существующих проблем.

Итак, весь процесс можно разбить на следующие этапы [9]:

- Понимание и формулировка задачи анализа.
- Подготовка данных для автоматизированного анализа.
- Применение методов ИА и построение моделей.

– Проверка построенных моделей.
– Интерпретация моделей человеком.

На первом этапе выполняется осмысление поставленной задачи и уточнение целей, которые должны быть достигнуты методами ИА. Важно правильно сформулировать цели и выбрать необходимые для их достижения методы, так как от этого зависит дальнейшая эффективность всего процесса.

Второй этап состоит в приведении данных к форме, пригодной для применения конкретных методов ИА. Данный процесс ниже будет описан более подробно, здесь заметим только, что вид преобразований, совершаемых над данными, во многом зависит от используемых методов, выбранных на предыдущем этапе.

Третий этап – это собственно применение методов ИА. Сценарии этого применения могут быть самыми различными и включать сложную комбинацию разных методов, особенно если используемые методы позволяют проанализировать данные с разных точек зрения.

Следующий этап – проверка построенных моделей. Очень простой и часто используемый способ заключается в том, что все имеющиеся данные, которые необходимо анализировать, разбиваются на две группы. Как правило, одна из них большего размера, другая – меньшего. На большей группе, применяя те или иные методы ИА, получают модели, а на меньшей – проверяют их. По разнице в точности между тестовой и обучающей группами можно судить об адекватности построенной модели.

Последний этап – интерпретация полученных моделей человеком в целях их использования для принятия решений, добавление получившихся правил и зависимостей в базы знаний и т.д. Этот этап часто подразумевает использование методов, находящихся на стыке технологий ИА и технологии экспертных систем. Оттого, насколько эффективным он будет, в значительной степени зависит успех решения поставленной задачи.

1.6. ЗАДАЧА ПОИСКА АССОЦИАТИВНЫХ ПРАВИЛ

Формальная постановка задачи. Одной из наиболее распространенных задач анализа данных является определение часто встречающихся наборов объектов в большом множестве наборов. Опишем эту задачу в обобщенном виде [10]. Для этого обозначим объекты, составляющие исследуемые наборы, следующим множеством:

$$I = \{i_1, i_2, ..., i_j, ..., i_n\},$$

где i_j – объекты, входящие в анализируемые наборы; n – общее количество объектов.

Наборы объектов из множества I, хранящиеся в БД и подвергаемые анализу, называются транзакциями. Опишем транзакцию как подмножество множества I:

$$T = \{i_j \mid i_j \in I\}.$$

Такие транзакции в магазине соответствуют наборам товаров, покупаемых потребителем и сохраняемых в БД в виде товарного чека или накладной. В них перечисляются приобретаемые покупателем товары, их цена, количество и др.

Набор транзакций, информация о которых доступна для анализа, обозначим следующим множеством:

$$D = \{T_1, T_2, ..., T_r, ..., T_m\},$$

где m – количество доступных для анализа транзакций.

Множество транзакций, в которые входит объект i_j, обозначим следующим образом:

$$D_{i_j} = \{T_r \mid i_j \in T_r; j = 1..n; r = 1..m\} \subseteq D.$$

Некоторый произвольный набор объектов обозначим следующим образом:

$$F = \{i_j \mid i_j \in I; j = 1..n\}.$$

Набор, состоящий из k объектов, называется k-элементным набором (в данном примере это 2-элементный набор).

Множество транзакций, в которые входит набор F, обозначим следующим образом:

$$D_F = \{T_r \mid F \subseteq T_r; r = 1..m\} \subseteq D.$$

Отношение количества транзакций, в которое входит набор F, к общему количеству транзакций называется поддержкой набора F и обозначается $Supp(F)$:

$$Supp(F) = \frac{|D_F|}{|D|}.$$

При поиске аналитик может указать минимальное значение поддержки интересующих его наборов $Supp_{\min}$. Набор называется частым, если значение его поддержки больше минимального значения поддержки, заданного пользователем:

$$Supp(F) > Supp_{\min}.$$

Таким образом, при поиске ассоциативных правил требуется найти множество всех частых наборов:

$$L = \{F \mid Supp(F) > Supp_{\min}\}.$$

Сиквенциальный анализ. При анализе часто вызывает интерес последовательность происходящих событий. При обнаружении закономерностей в таких последовательностях можно с некоторой долей вероятности предсказывать появление событий в будущем, что позволяет принимать более правильные решения.

Последовательностью называется упорядоченное множество объектов. Для этого на множестве должно быть задано отношение порядка.

Тогда последовательность объектов можно описать в следующем виде:

$$S = \{..., i_p, ..., i_q, ...\}, \text{ где } q < p .$$

Различают два вида последовательностей: с циклами и без циклов. В первом случае допускается вхождение в последовательность одного и того же объекта на разных позициях:

$$S = \{..., i_p, ..., i_q, ...\}, \text{ где } q < p, \text{ а } i_q = i_p .$$

Говорят, что транзакция T содержит последовательность S, если $S \subseteq T$ и объекты, входящие в S входят и в множество T с сохранением отношения порядка. При этом допускается, что в множестве T между объектами из последовательности S могут находиться другие объекты.

Поддержкой последовательности S называется отношение количества транзакций, в которое входит последовательность S, к общему количеству транзакций. Последовательность является частой, если ее поддержка превышает минимальную поддержку, заданную пользователем:

$$Supp(S) > Supp_{\min} .$$

Задачей сиквенциального анализа является поиск всех частых последовательностей:

$$L = \{S \mid Supp(S) > Supp_{\min}\} .$$

Основным отличием задачи сиквенциального анализа от поиска ассоциативных правил является установление отношения порядка между объектами множества I. Данное отношение может быть определено разными способами. При анализе последовательности событий, происходящих во времени, объектами множества I являются события, а отношение порядка соответствует хронологии их появления.

Сиквенциальный анализ актуален и для телекоммуникационных компаний. Основная проблема, для решения которой он используется, – это анализ данных об авариях на различных узлах телекоммуникационной сети. Информация о последовательности совершения аварий может помочь в обнаружении неполадок и предупреждении новых аварий.

Разновидности задачи поиска ассоциативных правил. Во многих прикладных областях объекты множества I естественным образом объединяются в группы, которые в свою очередь также могут объединяться в более общие группы, и т.д. Наличие иерархии изменяет представление о том, когда объект i присутствует в транзакции T. Очевидно, что поддержка не отдельного объекта, а группы, в которую он входит, больше:

$$Supp(I_q^g) \geq Supp(i_j), \text{ где } i_j \in I_q^g .$$

Это связано с тем, что при анализе групп подсчитываются не только транзакции, в которые входит отдельный объект, но и транзакции, содержащие все объекты анализируемой группы.

Использование иерархии позволяет определить связи, входящие в более высокие уровни иерархии, поскольку поддержка набора может увеличиваться, если подсчитывается вхождение группы, а не ее объекта. Кроме поиска наборов, часто встречающихся в транзакциях, состоящих из объектов $F = \{i \mid i \in I\}$ или групп одного уровня иерархии $F = \{I^g \mid I^g \in I^{g+1}\}$, можно рассматривать также смешанные наборы объектов и групп $F = \{i, I^g \mid i \in I^g, I^{g+1}\}$. Это позволяет расширить анализ и получить дополнительные знания.

При иерархическом построении объектов можно варьировать характер поиска, изменяя анализируемый уровень. Очевидно, что чем больше объектов в множестве I, тем больше объектов в транзакциях T и частых наборах. Это в свою очередь увеличивает время поиска и усложняет анализ результатов. Уменьшить или увеличить количество данных можно с помощью иерархического представления анализируемых объектов. Перемещаясь вверх по иерархии, обобщаем данные и уменьшаем из количество, и наоборот.

Недостатком обобщения объектов является меньшая полезность полученных знаний, т.е. в этом случае они относятся к группам товаров, что не всегда приемлемо. Для достижения компромисса между анализом групп и анализом отдельных объектов часто поступают следующим образом: сначала анализируют группы, а затем в зависимости от полученных результатов исследуют объекты заинтересовавших аналитика групп. В любом случае можно утверждать, что наличие иерархии в объектах и ее использование в задаче поиска ассоциативных правил позволяют выполнять более гибкий анализ и получать дополнительные знания.

В рассмотренной задаче поиска ассоциативных правил наличие объекта в транзакции определялось только его присутствием в ней ($i_j \in T$) или отсутствием $i_j \notin T$. Часто объекты имеют дополнительные атрибуты, как правило, численные. Например, товары в транзакции имеют атрибуты: цена и количество. При этом наличие объекта в наборе может определяться не просто фактом его присутствия, а выполнением условия по отношению к определенному атрибуту. Например, при анализе транзакций, совершаемых покупателем, может интересовать не просто наличие покупаемого товара, а товара, покупаемого по некоторой цене.

Для расширения возможностей анализа с помощью поиска ассоциативных правил в исследуемые наборы можно добавлять дополнительные объекты. В общем случае они могут иметь природу, отличную от основных объектов. Например, для определения товаров, имеющих больший спрос в зависимости от месторасположения магазина, в транзакции можно добавить объект, характеризующий район.

Представление результатов. Решение задачи поиска ассоциативных правил, как и любой задачи, сводится к обработке исходных данных и получению результатов. Обработка исходных данных выполняется по некоторому алгоритму ИА. Результаты, получаемые при решении этой задачи, принято представлять в виде ассоциативных правил. В связи с этим при их поиске выделяют два основных этапа:

– Нахождение частых наборов объектов.
– Генерация ассоциативных правил, найденных частых наборов объектов.

Ассоциативные правила имеют следующий вид:

Если (условие) то (результат),

где условие – обычно не логическое выражение (как в классификационных правилах), а набор объектов из множества I, с которыми связаны (ассоциированы) объекты, включенные в результат данного правила.

Как уже отмечалось, в ассоциативных правилах условие и результат являются объектами множества I:

Если X то Y,

где $X \in I,\ Y \in I,\ X \cup Y = \varphi$.

Ассоциативное правило можно представить как импликацию над множеством: $X \Rightarrow Y$, где $X \in I,\ Y \in I,\ X \cup Y = \varphi$.

Основным достоинством ассоциативных правил является их легкое восприятие человеком и простая интерпретация языками программирования. Однако они не всегда полезны. Выделяют три вида правил.

– Полезные правила содержат действительную информацию, которая ранее была неизвестна, но имеет логичное объяснение. Такие правила могут быть использованы для принятия решений, приносящих выгоду.

– Тривиальные правила содержат действительную и легко объяснимую информацию, которая уже известна. Такие правила, хотя и объяснимы, но не могут принести какой-либо пользы, так как отражают или известные законы в исследуемой области, или результаты прошлой деятельности. Иногда такие правила могут использоваться для проверки выполнения решений, принятых на основании предыдущего анализа.

– Непонятные правила содержат информацию, которая не может быть объяснена. Такие правила могут быть получены или на основе аномальных значений, или глубоко скрытых знаний. Напрямую такие правила нельзя использовать для принятия решений, так как их необъяснимость может привести к непредсказуемым результатам. Для лучшего понимания требуется дополнительный анализ.

Ассоциативные правила строятся на основе частых наборов. Так, правила, построенные на основании набора F (т.е. $X \cup Y = F$), являются всеми возможными комбинациями объектов, входящих в него.

Таким образом, количество ассоциативных правил может быть очень большим и тяжело воспринимаемым для человека. К тому же, не все из построенных правил несут в себе полезную информацию. Для оценки их полезности вводятся следующие величины.

Поддержка 0 показывает, какой процент транзакций поддерживает данное правило. Так как правило строится на основании набора, то, значит, правило $X \Rightarrow Y$ имеет поддержку, равную поддержке набора F, который составляют X и Y:

$$Supp_{X \Rightarrow Y} = Supp_F = \frac{|D_{F=X \cup Y}|}{|D|}.$$

Очевидно, что правила, построенные на основании одного и того же набора, имеют одинаковую поддержку.

Достоверность показывает вероятность того, что из наличия в транзакции набора X следует наличие в ней набора Y. Достоверностью правила $X \Rightarrow Y$ является отношение числа транзакций, содержащих наборы X и Y, к числу транзакций, содержащих набор X:

$$Conf_{X \Rightarrow Y} = \frac{|D_{F=X \cup Y}|}{|D_X|} = \frac{Supp_{X \cup Y}}{Supp_X}.$$

Очевидно, что чем больше достоверность, тем правило лучше, причем у правил, построенных на основании одного и того же набора, достоверность будет разная.

К сожалению, достоверность не позволяет оценить полезность правила. Если процент наличия в транзакциях набора Y при условии наличия в них набора X меньше, чем процент безусловного наличия набора Y:

$$Conf_{X \Rightarrow Y} = \frac{Supp_{X \cup Y}}{Supp_X} < Supp_y,$$

это значит, что вероятность случайно угадать наличие в транзакции набора Y больше, чем предсказать это с помощью правила $X \Rightarrow Y$. Для исправления такой ситуации вводится мера – улучшение.

Улучшение показывает, полезнее ли правило случайного угадывания. Улучшение правила – это отношение числа транзакций, содержащих наборы X и Y, к произведению количества транзакций, содержащих набор X, и количества транзакций, содержащих набор Y:

$$impr_{X \Rightarrow Y} = \frac{|D_{F=X \cup Y}|}{|D_X||D_Y|} = \frac{Supp_{X \cup Y}}{Supp_X * Supp_Y}.$$

Если улучшение больше единицы, то это значит, что с помощью правила предсказать наличие набора Y вероятнее, чем случайное угадывание, если меньше единицы, то наоборот.

В последнем случае можно использовать отрицающее правило, т.е. правило, которое предсказывает отсутствие набора Y: $X \Rightarrow \neg Y$.

У такого правила улучшение будет больше единицы, так как $Supp_{\neg Y} = 1 - Supp_Y$.

Таким образом, можно получить правило, которое предсказывает результат лучше, чем случайным образом. Правда, на практике такие правила мало применимы.

Данные оценки используются при генерации правил. Аналитик при поиске ассоциативных правил задает минимальные значения перечисленных величин. В результате те правила, которые не удовлетворяют этим условиям, отбрасываются и не

включаются в решение задачи. С этой точки зрения нельзя объединять разные правила, хотя и имеющие общую смысловую нагрузку. Например, следующие правила:

$$X = \{i_1, i_2\} \Rightarrow Y = \{i_3\},$$
$$X = \{i_1, i_2\} \Rightarrow Y = \{i_4\}$$

нельзя объединить в одно $X = \{i_1, i_2\} \Rightarrow Y = \{i_3, i_4\}$,

так как достоверности их будут разные, следовательно, некоторые из них могут быть исключены, а некоторые – нет.

Если объекты имеют дополнительные атрибуты, которые влияют на состав объектов в транзакциях, а следовательно, и в наборах, то они должны учитываться в генерируемых правилах. В этом случае условная часть правил будет содержать не только проверку наличия объекта в транзакции, но и более сложные операции сравнения: больше, меньше, включает и др. Результирующая часть правил также может содержать утверждения относительно значений атрибутов.

Алгоритмы. *Алгоритм Apriori.* Выявление частых наборов объек- тов – операция, требующая большого количества вычислений, а следовательно, и времени. Алгоритм Apriori описан в 1994 г. [10]. Он использует одно из свойств поддержки, гласящее: поддержка любого набора объектов не может превышать минимальной поддержки любого из его подмножеств:

$$Supp_F \le Supp_E \text{ при } E \subset F .$$

Алгоритм Apriori определяет часто встречающиеся наборы за несколько этапов. На i-м этапе определяются все часто встречающиеся i-элементные наборы. Каждый этап состоит из двух шагов: формирование кандидатов и подсчет поддержки кандидатов.

Рассмотрим i-й этап. На шаге формирования кандидатов алгоритм создает множество кандидатов из i-элементных наборов, чья поддержка пока не вычисляется. На шаге подсчета кандидатов алгоритм сканирует множество транзакций, вычисляя поддержку наборов-кандидатов. После сканирования отбрасываются кандидаты, поддержка которых меньше определенного пользователем минимума, и сохраняются только часто встречающиеся *i*-элементные наборы. Во время 1-го этапа выбранное множество наборов-кандидатов содержит все 1-элементные частые наборы. Алгоритм вычисляет их поддержку во время шага подсчета кандидатов.

Описанный алгоритм можно записать в виде следующего псевдокода:

L_1 = {часто встречающиеся 1-элементые наборы}
Для ($k = 2$; L_{k-1} <> Ø; k++)
  C_k = Apriorigen(Fk-1)
  Для всех $t \in D$ выполнить
    $C_t = subset(C_k, t)$
    Для всех $c \in C_t$ выполнить
      $c.count++$
    Конец для всех
  Конец для всех
  $L_k = \{c \in C_k \mid c.count \ge Supp_{min}\}$
Конец для
Результат = $\bigcup_k L_k$

Опишем обозначения, используемые в алгоритме:

- L_k – множество k-элементных частых наборов, чья поддержка не меньше заданной пользователем. Каждый член множества имеет набор упорядоченных ($i_j < i_p$, если $j < p$) элементов F и значение поддержки набора $Supp_F < Supp_{min}$: $L = \{(F_1, Supp_1), (F_2, Supp_2), ..., (F_q, Supp_q)\}$, где $F = \{i_1, i_2, ..., i_k\}$;
- C_k – множество кандидатов k-элементных наборов потенциально частых. Каждый член множества имеет набор упорядоченных ($i_j < i_p$, если $j < p$) элементов F и значение поддержки набора $Supp$.

Опишем алгоритм по шагам.

Шаг 1. Присвоить $k = 1$ и выполнить отбор всех 1-элементных наборов, у которых поддержка больше минимально заданной пользователем $Supp_{min}$.

Шаг 2. $k = k + 1$.

Шаг 3. Если не удается создавать k-элементные наборы, то завершить алгоритм, иначе выполнить следующий шаг.

Шаг 4. Создать множество k-элементных наборов кандидатов в частые наборы. Для этого необходимо объединить в k-элементные кандидаты ($k-1$)-элементные частые наборы. Каждый кандидат $c \in C_k$ будет формироваться путем добавления к ($k-1$)-элементному частому набору p элемента из другого ($k-1$)-элементного частого набора q. Причем добавляется последний элемент набора q, который по порядку выше, чем последний элемент набора p ($p.item_{k-1} < q.item_{k-1}$). При этом первые все $k-2$ элемента обоих наборов одинаковы ($p.item_1 = q.item_1$, $p.item_2 = q.item_2$, ..., $p.item_{k-2} = p.item_{k-2}$).

Шаг 5. Для каждой транзакции T из множества D выбрать кандидатов C_t из множества C_k, присутствующих в транзакции T. Для каждого набора из построенного множества C_k удалить набор, если хотя бы одно из его ($k-1$) подмножеств не является часто встречающимся, т.е. отсутствует во множестве L_{k-1}. Это можно записать в виде следующего псевдокода.

Для всех $c \in C_k$ выполнить
  Для всех ($k-1$) – поднаборов s из c выполнить

Если ($s \notin L_{k-1}$) то

Удалить c из C_k

Шаг 6. Для каждого кандидата из множества C_k увеличить значение поддержки на единицу.

Шаг 7. Выбрать только кандидатов L_k из множества C_k, у которых значение поддержки больше заданной пользователем $Supp_{\min}$. Вернуться к шагу 2.

Результатом работы алгоритма является объединение всех множеств L_k для всех k.

Разновидности алгоритма Apriori. Алгоритм AprioriTid является разновидностью алгоритма Apriori. Отличительной чертой данного алгоритма является подсчет значения поддержки кандидатов не при сканировании множества D, а с помощью множества $\overline{C}_k$, являющегося множеством кандидатов (k-элементных наборов) потенциально частых, в соответствие которым ставятся идентификатор TID транзакций, в которых они содержатся.

Каждый член множества $\overline{C}_k$ является парой $< TID, \{F_k\} >$, где ка-ждый F_k является потенциально частым k-элементным набором, представленным в транзакции с идентификатором TID. Множество $\overline{C}_1 = D$ соответствует множеству транзакций, хотя каждый объект в транзакции соответствует однообъектному набору в множестве $\overline{C}_1$, содержащем этот объект. Для $k > 1$ множество $\overline{C}_k$ генерируется в соответствии с алгоритмом, описанным ниже. Член множества $\overline{C}_k$, соответствующий транзак- ции T, является парой следующего вида:

$$< T.TID, \{c \in C_k \mid c \in T\} > .$$

Подмножество наборов в $\overline{C}_k$ с одинаковыми TID (т.е. содержатся в одной и той же транзакции) называется записью. Если транзакция не содержит ни одного k-элементного кандидата, то $\overline{C}_k$ не будет иметь записи для этой транзакции, т.е. количество записей в $\overline{C}_k$ может быть меньше, чем в D, особенно для больших значений k. Кроме того, для больших значений k каждая запись может быть больше, чем соответствующая ей транзакция, так как в транзакции будет содержаться мало кандидатов. Однако для малых значений k каждая запись может быть больше, чем соответствующая транзакция, так как $\overline{C}_k$ включает всех кандидатов k-элементных наборов, содержащихся в транзакции.

Другой разновидностью алгоритма Apriori является алгоритм MSAP (Mining Sequential Alarm Patterns), специально разработанный для выполнения сиквенкциального анализа сбоев телекоммуникационной сети.

Он использует следующее свойство поддержки последовательностей: для любой последовательности L_k ее поддержка будет меньше, чем поддержка последовательностей из множества L_{k-1}.

Алгоритм MSAP для поиска событий, следующих друг за другом, использует понятие «срочного окна». Это позволяет выявлять не просто одинаковые последовательности событий, а следующие друг за другом. В остальном данный алгоритм работает по тому же принципу, что и Apriori.

1.7. КЛАСТЕРИЗАЦИЯ

Постановка задачи кластеризации. Первые публикации по кластерному анализу появились в конце 30-х прошлого столетия, но активное развитие этих методов и их широкое использование началось в конце 60-х – начале 70-х годов [11]. В дальнейшем это направление анализа интенсивно развивалось. Появились новые методы, модификации уже известных алгоритмов, существенно расширилась область применения кластерного анализа. Если первоначально эти методы использовались в психологии, археологии, биологии, то сейчас они стали активно применяться в социологии, экономике, статистике, в исторических исследованиях.

Кластеризация отличается от классификации тем, что для проведения анализа не требуется иметь выделенную целевую переменную, с этой точки зрения она относится к классу задач с обучением без учителя. Эта задача решается на начальных этапах исследования, когда о данных мало что известно. Ее решение помогает лучше понять данные, и с этой точки зрения задача кластеризации является описательной задачей.

Для этапа кластеризации характерно отсутствие каких-либо различий как между переменными, так и между записями. Напротив, ищутся группы наиболее близких, похожих записей. Методы автоматического разбиения на кластеры редко используются сами по себе, просто для получения групп схожих объектов. Анализ только начинается с разбиения на кластеры. После определения кластеров используются другие методы ИА для того, чтобы попытаться установить, а что означает такое разбиение на кластеры, чем оно вызвано.

Большое достоинство кластерного анализа в том, что он позволяет производить разбиение объектов не по одному параметру, а по целому набору признаков. Кроме того, кластерный анализ, в отличие от большинства математико-статистических методов, не накладывает никаких ограничений на вид рассматриваемых объектов и позволяет рассматривать множество исходных данных практически произвольной природы. Это имеет большое значение, например, для прогнозирования конъюнктуры при наличии разнородных показателей, затрудняющих применение традиционных экономических подходов.

Кластерный анализ позволяет рассматривать достаточно большой объем информации и резко сокращать, сжимать большие массивы информации, делать их компактными и наглядными.

Задача кластеризации состоит в разделении исследуемого множества объектов на группы «похожих» объектов, называемых кластерами.

Один из способов решения задачи кластеризации – построение набора характеристических функций классов, которые показывают, относится ли объект данных к данному классу или нет. Характеристическая функция класса может быть двух типов:

1) дискретная функция, принимающая одно из двух определенных значений, смысл которых в принадлежности/непринадлежности объекта данных заданном классу;

2) функция, принимающая вещественные значения, например из интервала 0…1. Чем ближе значение функции к единице, тем больше объект данных принадлежит заданному классу.

Формальная постановка задачи. Дано: набор данных со следующими свойствами:

- каждый экземпляр данных выражается четким числовым значением;
- класс для каждого конкретного экземпляра данных неизвестен.

Найти:

- способ сравнения данных между собой (меру сходства);
- способ кластеризации;
- разбиение данных по кластерам.

Формально задача кластеризации описывается следующим образом.

Дано множество объектов данных I, каждый из которых представлен набором атрибутов. Требуется построить множество кластеров C и отображение F множества I на множество C, т.е. $F : I \to C$. Отображение F задает модель данных, являющуюся решением задачи. Качество решения задачи определятся количеством верно классифицированных объектов данных.

Множество I определим следующим образом:

$$I = \{i_1, i_2, ..., i_j, ..., i_n\},$$

где i_j – исследуемый объект.

Неотрицательное значение $d(i_j, i_p)$ называется расстоянием между элементами i_j и i_p, если выполняются следующие условия:

а) $d(i_j, i_p) \geq 0$ для всех i_j и i_p;

б) $d(i_j, i_p) = 0$ тогда и только тогда, когда $i_j = i_p$;

в) $d(i_j, i_p) = d(i_j, i_p)$;

г) $d(i_j, i_p) \leq d(i_j, i_r) + d(i_r, i_p)$.

Если расстояние $d(i_j, i_p)$ меньше некоторого значения σ, то говорят, что элементы близки и помещаются в один кластер. В противном случае говорят, что элементы отличны друг от друга и их помещают в разные кластеры.

Большинство популярных алгоритмов, решающих задачу кластеризации, используют в качестве формата входных данных матрицу отличия D. Строки и столбцы матрицы соответствуют элементам множества I. Элементами матрицы являются значения $d(i_j, i_p)$ в строке j и столбце p. Очевидно, что на главной диагонали значения будут равны нулю:

$$D = \begin{pmatrix} 0 & d(e_1, e_2) & d(e_1, e_n) \\ d(e_2, e_1) & 0 & d(e_2, e_n) \\ d(e_n, e_1) & d(e_n, e_2) & 0 \end{pmatrix}.$$

Большинство алгоритмов работают с симметричными матрицами. Если матрица несимметрична, то ее можно привести к симметричному виду путем следующего преобразования:

$$(D + D^m) / 2.$$

Рассмотрим меры близости, основанные на расстояниях и используемые в алгоритмах кластеризации.

Расстояния между объектами предполагают их представление в виде точек *m*-мерного пространства R^m. В этом случае могут быть использованы различные подходы к вычислению расстояний.

Рассмотренные ниже меры определяют расстояния между двумя точками, принадлежащими пространству входных переменных. Используются следующие обозначения:

$X_Q \subseteq R^m$ – множество данных, являющееся подмножеством *m*-мерного вещественного пространства;

$x_i = (x_{i1}, ..., x_{im}) \in X_Q,\ i = \overline{1, Q}$ – элементы множества данных;

$\bar{x} = \frac{1}{Q} \sum_{i=1}^{Q} x_i$ – среднее значение точек данных;

$S = \frac{1}{Q-1} \sum_{i=1}^{Q} (x_i - \bar{x})(x_i - \bar{x})^t$ – ковариационная матрица ($m \times m$).

Приведем наиболее известные меры близости.

Евклидово расстояние. Иногда может возникнуть желание возвести в квадрат стандартное евклидово расстояние, чтобы придать больше веса более отдаленным друг от друга объектам. Это расстояние вычисляется следующим образом:

$$d_2(x_i, x_j) = \sqrt{\sum_{t=1}^{m} (x_{it} - x_{jt})^2}.$$

Расстояние по Хеммингу. Это расстояние является просто средним разностей по координатам. В большинстве случаев данная мера расстояния приводит к таким же результатам, как и для обычного расстояния Евклида, однако для нее влияние

отдельных больших разностей (выбросов) уменьшается (так как они не возводятся в квадрат). Расстояние по Хеммингу вычисляется по формуле

$$d_H(x_i, x_j) = \sum_{t=1}^{m} | x_{it} - x_{jt} | .$$

Расстояние Чебышева. Это расстояние может оказаться полезным, когда желают определить два объекта как «различие», если они различаются по какой-либо одной координате (каким-либо одним измерением). Расстояние Чебышева вычисляется по формуле

$$d_\infty(x_i, x_j) = \max_{1 \le l \le m} | x_{it} - x_{jt} | .$$

Расстояние Махаланобиса преодолевает этот недостаток, но данная мера расстояния плохо работает, если ковариационная матрица высчитывается на всем множестве входных данных. В то же время, будучи сосредоточенной на конкретном классе (группе данных), данная мера расстояния показывает хорошие результаты:

$$d_M(x_i, x_j) = (x_i - x_j) S^{-1} (x_i - x_j)^t .$$

Пиковое расстояние предполагает независимость между случайными переменными, что говорит о расстоянии в ортогональном пространстве. Но в практических приложениях эти переменные не являются независимыми:

$$d_L(x_i, x_j) = \frac{1}{m} \sum_{t=1}^{m} \frac{| x_{it} - x_{jt} |}{x_{it} + x_{jt}} .$$

Базовые алгоритмы кластеризации. *Классификация алгоритмов.* При выполнении кластеризации важно, сколько в результате должно быть построено кластеров. Предполагается, что кластеризация должна выявить естественные локальные сгущения объектов. Поэтому число кластеров является параметром, часто существенно усложняющим вид алгоритма, если предполагается неизвестным, и существенно влияющим на качество результата, если оно известно.

Большую популярность при решении задач кластеризации приобрели алгоритмы, основанные на поиске оптимального в определенном смысле разбиения множества данных на кластеры. Во многих задачах в силу своих достоинств используются именно алгоритмы построения разбиения. Эти алгоритмы пытаются сгруппировать данные (в кластеры) таким образом, чтобы целевая функция алгоритма разбиения достигала экстремума (минимума). Рассмотрим три основных алгоритма кластеризации, основанных на методах разбиения. В данных алгоритма используются следующие базовые понятия.

- Обучающее множество (входное множество данных) M, на котором строится разбиение;
- Метрика расстояния:

$$d_A^2(m_j, c^{(i)}) = \left\| m_j - c^{(i)} \right\|_A^2 = (m_j - c^{(i)})^t A (m_j - c^{(i)}),$$

где матрица A определяет способ вычисления расстояния. Например, для единичной матрицы будем использовать расстояние по Евклиду;

- Вектор центров кластеров C;
- Матрица разбиения по кластерам U;
- Целевая функция $J = J(M, d, C, U)$;
- Набор ограничений.

Алгоритм k-средних. При работе алгоритма выбираются k произвольных исходных центров – точек в пространстве объектов [11]. Далее итерационно выполняется операция двух шагов.

На первом шаге все объекты разбиваются на k групп, наиболее близких к одному из центров. Близость определяется расстоянием, которое вычисляется одним из описанных ранее способов (например, берется евклидово расстояние).

На втором шаге вычисляются новые центры кластеров. Центы можно вычислить как средние значения переменных объектов, отнесенных к сформированным группам. Новые центры могут отличаться от предыдущих. Рассмотренная операция повторяется рекурсивно до тех пор, пока центры кластеров (соответственно, и границы между ними) не перестанут меняться.

Данный алгоритм является прообразом практически всех алгоритмов нечеткой кластеризации.

Базовые определения и понятия в рамках данного алгоритма имеют следующий вид.

- Обучающее множество $M = \{m_j\}_{j=1}^{d}$, d – количество точек (векторов) данных;
- Вектор центров кластеров $C = \{c^{(i)}\}_{i=1}^{c}$,

где $c^{(i)} = \dfrac{\sum_{j=1}^{d} u_{ij} m_j}{\sum_{j=1}^{d} u_{ij}}$, $1 \le i \le c$;

- Матрица разбиения $U = \{u_{ij}\}$,

где $u = \begin{cases} 1 \text{ при } d(m_l, c^{(l)}) = \min\limits_{l \le k \le c} d(m_j, c_k^{(l)}); \\ 0 \quad \text{в остальных случаях}; \end{cases}$

- Целевая функция

$$J(M, U, C) = \sum_{i=1}^{c} \sum_{j=1}^{d} u_{ij} d_A^2(m_j, c^{(l)});$$

– Набор ограничений

$$u_{ij} \in \{0, 1\};\ \sum_{i=1}^{c} u_{ij} = 1;\ 0 < \sum_{j=1}^{d} u_{ij} < d,$$

который определяет, что каждый вектор данных может принадлежать только одному кластеру и не принадлежать остальным. В каждом кластере содержится не менее одной точки, но менее общего количества точек.

Алгоритм представляет собой итерационную процедуру следующего вида.

Шаг 1. Проинициализировать начальное разбиение (например, случайным образом), выбрать точность δ (используется в условии завершения алгоритма), проинициализировать номер итерации $l = 0$.

Шаг 2. Определить центры кластеров по следующей формуле:

$$c_l^{(i)} = \frac{\sum_{j=1}^{d} u_{ij}^{(l-1)} m_j}{\sum_{j=1}^{d} u_{ij}^{(l-1)}},\ 1 \le i \le c.$$

Шаг 3. Обновить матрицу разбиения с тем, чтобы минимизировать квадраты ошибок, используя формулу

$$u = \begin{cases} 1 \text{ при } d(m_l, c^{(l)}) = \min\limits_{l \le k \le c} d(m_j, c_k^{(l)}); \\ 0 \quad \text{в остальных случаях.} \end{cases}$$

Шаг 4. Проверить условие $\left\| U^{(l)} - U^{(l-1)} \right\| < \delta$. Если условие выполняется, завершить процесс, если нет – перейти к шагу 2 с номером итерации $l = l + 1$.

Основной недостаток, присущий данному алгоритму в силу дискретного характера элементов матрицы разбиения, – большой размер пространства разбиения.

Одним из способов устранения данного недостатка является представление элементов матрицы разбиения числами из единичного интервала, т.е. принадлежность элемента данных кластеру должна определяться функцией принадлежности – элемент данных может принадлежать нескольким кластерам с различной степенью принадлежности.

Глава 2. РАЗРАБОТКА СИСТЕМЫ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ВЫПОЛНЕНИЯ БИЗНЕС-ПРОЦЕССОВ В ИНФОРМАЦИОННОЙ СРЕДЕ СОВРЕМЕННОГО ПРЕДПРИЯТИЯ

В главе рассмотрены алгоритмы, помогающие аналитику проследить актуальное выполнение бизнес-процессов в системе электронного документооборота (СЭД), отражая аспекты реального поведения бизнес-процесса при выполнении (в результате выполнения бизнес-процесс проходит через множество точек выбора), выявляя незафиксированные формально правила принятия решений при прохождении через множество точек выбора, помогая отследить изменение этих правил со временем. Разработанные алгоритмы базируются на технике интеллектуального анализа, расширяя ее для конкретной предметной области и улучшая показатели производительности.

Полученные в результате работы алгоритмов знания могут быть использованы для решения проблем:

- Предсказание успешного завершения экземпляра бизнес-процесса. Учитывая нынешнее состояние выполнения задач в экземпляре бизнес-процесса и используя статистику по выполнению модели автоматизированного бизнес-процесса (МАБП) в прошлом, можно, отфильтровав последовательности по списку текущих задач и применив алгоритм поиска частых подпоследовательностей, ответить на этот вопрос.
- Идентификация критических задач. Во многих МАБП некоторые задачи могут быть рассмотрены как критические, в смысле планирования их выполнения как системы управления потоками работ (Workflow systems, WFMS) в каждом удачном случае выполнения. Как правило, администратор знает на основе личного опыта, какая из задач является критической в МАБП, однако не исключены случаи необходимости использования статистики актуального выполнения системы для прояснения этого вопроса.
- Характеристика неудачного/успешного выполнения. Анализируя предыдущее выполнение, аналитик может поинтересоваться, какие дискриминантные факторы характеризуют удачное или неудачное выполнение бизнес-процесса.
- Улучшение МАБП. Информация, собранная в журнале выполнения МАБП, может быть выгодно использована для «оптимизации» выполнения. Например, критерий оптимальности может быть зафиксирован на каком-нибудь интересном параметре, таком, как качество сервиса или среднее время завершения МАБП.

Также в главе приведены результаты измерений показателей производительности предложенных алгоритмов и архитектура реализующей их системы.

2.1. РАЗРАБОТКА АЛГОРИТМА ПОЛУЧЕНИЯ КОМПЛЕКСНЫХ МАБП

Модели, основанные на графах, широко используются в большом спектре прикладных задач как интуитивно понятный и в то же время формальный способ представления разного рода данных, таких, как web-документы, химическое строение, модели процессов, поведенческие шаблоны. Применение теории графов для анализа МАБП общепризнано.

Пример такой модели бизнес-процесса, описывающей обработку заявок от клиентов, представлен ниже на рис. 1 (пример построен на основе реального бизнес-процесса компании «Softintegro»). Опишем задачи, участвующие в модели процесса:

- *a* – «Регистрация заказа»,
- *b* – «Аутенфикация клиента»,
- *f* – «Регистрация нового клиента»,
- *i* – «Надежность клиента»,
- *l* – «Принятие заказа»,
- *c* – «Проверка наличия»,
- *d* – «Запрос к поставщику»,
- *g* – «План производства»,
- *h* – «Отклонение заказа»,
- *o* – «Ускорение»,
- *n* – «Подготовка счета»,
- *m* – «Назначение скидки».

При выполнении такого процесса обнаруживается множество ограничений, например в каждом экземпляре процесса, в котором участвовала задача *b*, должна появиться задача *i*, или если выполнена задача *l*, то предшествовать ее выполнению должны задачи *i* и *g*. Что очевидно с точки зрения бизнес-логики: если в бизнес-процессе происходила аутенфикация клиента, то должен быть выполнен пункт «надежность клиента»; если в бизнес-процессе была выполнена задача «принятие заказа», то до этого обязательно были выполнены задачи «план производства» и «надежность клиента».

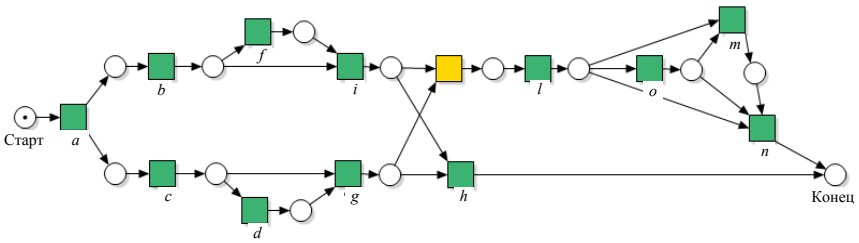

Рис. 1. МАБП «Обработка заказа от клиента»

Таким образом, *WF*-модель описывает выполнение первоначального бизнес-процесса, выполнение которого может быть запротоколировано в БД.

2.2. ОПРЕДЕЛЕНИЕ ТЕРМИНОВ

Определим понятие журнала выполнения бизнес-процесса. Пусть $T = \{A, B, C, ...\}$ – множество задач бизнес-процесса, тогда нумерованный список элементов этого множества мы будем называть *последовательностью* (или *следом*), обозначаемой как $\alpha(\alpha_1 \to \alpha_2 \to ... \to \alpha_n)$. Некоторое множество последовательностей L мы будем называть журналом выполнения автоматизированного бизнес-процесса.

Помимо приведенного определения журнала выполнения автоматизированного бизнес-процесса в работе потребуется понятие подпоследовательности. Последовательность α является *подпоследовательностью* β (обозначается как $\alpha \prec \beta$) тогда и только тогда, когда существуют числа $i_1 < i_2 < ... < i_n$ такие, что $\alpha_i = \beta_i$ для всех α_j.

Для алгоритма восстановления комплексных МАБП понятие комплексной WF-модели является ключевым.

О п р е д е л е н и е 1. Пусть P – это процесс. *Комплексная WF-модель* для P определяется как $WS(P)$ – множество $\{WS^1, ..., WS^m\}$ *WF*-моделей [11] для P. Размер $WS(P)$ определяется как $|WS(P)|$ – количество *WF*-моделей, содержащихся в ней.

Заметим также, что если последовательность в журнале принадлежит некоторой модели из множества комплексной схемы, то она также принадлежит и комплексной *WF*-модели.

О п р е д е л е н и е 2. Пусть дан след s журнала L, тогда *WF*-модель $I = \alpha + (s)$ (где $\alpha +$ – альфа плюс алгоритм ван-дер-Аальста, см. [12]) будем называть *экземпляром*.

Во многих исследованиях (работы Кука [11, 13], работы Хэрбста [14 – 18]), даже несмотря на различия используемого в них синтаксиса, подразумевается (хоть и не указывается явно), что локальные ограничения бизнес-процессов могут быть выражены с использованием трех функций IN, $OUT_{\min}$ и $OUT_{\max}$, сопоставляющих каждому узлу число:

- $\forall a \in A - \{a_0\},\ 0 < IN(a) < InDegree(a)$;
- $\forall a \in A - F,\ 0 < OUT_{\min}(a) \le OUT_{\max}(a) \le OutDegree(a)$;
- $IN(a_0) = 0$ и $\forall a \in F,\ OUT_{\min}(a) = OUT_{\max}(a) = 0$.

Здесь $InDegree(a) = |\{e = (b, a)\}|$, $OutDegree(a) = |\{e = (a, b)\}|$ и $e \in E$.

Задача a может стартовать тогда, когда как минимум $IN(a)$ ее предшественников были завершены. Две типичные ситуации: 1) если $IN(a) = InDegree(a)$, тогда a является *and-join* задачей, которая может быть выполнена только тогда, когда все ее предшественники были выполнены, и 2) если $IN(a) = 1$, тогда a является *or-join* задачей, которая может быть выполнена, если хотя бы одна из ее предшественников была выполнена. После завершения задача a может запустить одно непустое множество исходящих дуг с мощностью между $OUT_{\min}(a)$ и $OUT_{\max}(a)$. Если $OUT_{\max}(a) = OutDegree(a)$, тогда a называется *full-fork*, если $OUT_{\min}(a) = OUT_{\max}(a)$, тогда a называется *and-split*, которая активирует все ее последующие задачи. Итак, если $OUT_{\max}(a) = 1$, тогда a называется *xor-split*.

Выше были описаны *локальные ограничения*, однако в моделях существуют и *глобальные ограничения*. Глобальные ограничения в отличие от локальных отображают отношения между задачами, не обязательно связанными друг с другом посредством дуг. Такие ограничения богаче по своей природе, и их представление зависит от частной прикладной задачи моделируемого бизнес-процесса. Часто они выражаются с использованием сложного формализма. Как пример глобального ограничения в процессе на рис. 1 $f \to \neg m$ показывает, что если задача f была выполнена, то задача m не может быть выполнена. В контексте предложенной примерной модели бизнес-процесса это означает, что если был зарегистрирован новый клиент, то к его заказу не может быть начислена скидка.

Пусть даны множества глобальных $(C_G(P))$ и локальных $(C_L(P))$ ограничений для процесса P. Допустим, дан подграф исходной *WF*-модели для процесса $P - I$, а также ограничение $c \in C_L(P) \cup C_G(P)$. Мы записываем $I \models c$, если I удовлетворяет c. Более того, если $I \models c$ для всех c в $C_L(P) \cup C_G(P)$ и содержит стартовую задачу a_0 и конечную задачу из F, тогда I называется экземпляром процесса P.

Обычно журналы записываются посредством следов. Длина следа обозначается как $lenght(s)$, а множество всех задач в следе обозначим через $tastks(s) = \bigcup_{1 \le i \le lenght(s)} s[i]$. Тогда журнал для процесса P, обозначаемый как L_P, будет мультимножеством следов: $L_P = [s \mid s \in A^*]$.

Пусть дан след s в журнале L_P, P – процесс и I – экземпляр этого процесса. Тогда s соответствует процессу P посредством I, обозначается через $P \models^I s$, т.е. в s содержатся задачи из A_I, таким образом, что каждая $(a, b) \in E_I$, $i < j$, где $s[i] = a$ и $s[j] = b$. Более того s является совместимой с P, обозначаемой как $P \models s$, если существует I с $P \models^I s$. Итак, ослабленная формулировка соответствия, которая не полагается на существование экземпляра I, может быть определена как $P \mapsto s$.

2.3. АЛГОРИТМ ПОЛУЧЕНИЯ КОМПЛЕКСНЫХ МАБП

Конечной целью работы алгоритма является получение комплексной *WF*-модели для некоторого данного изначально журнала выполнения *L*, которая могла бы генерировать этот журнал «полным» настолько, насколько это возможно [19]. Эта интуитивная посылка используется для того, чтобы ввести два критерия, называемых *завершенность* и *непротиворечивость*.

Определение 3. Под *непротиворечивостью* будем понимать $s(WS, L) = \dfrac{|\{I \mid i \models WS \wedge \neg\exists s \in L\}|}{|\{I \mid I \models WS\}|}$, т.е. процент последовательностей, принадлежащих обобщенной *WF*-модели и не имеющих соответствующих последовательностей в журнале выполнения. Под *завершенностью* будем понимать $q(WS, L) = \dfrac{|\{s \mid s \in L \wedge s \models WS\}|}{|\{s \mid s \in L\}|}$, т.е. процент последовательностей, имеющихся в журнале выполнения бизнес-процесса.

Легко видеть, что комплексная *WF*-модель, состоящая всего из одного процесса, содержит каждую последовательность в журнале бизнес-процесса, а это не совсем то, чего мы хотели бы достичь. Поэтому введем теперь еще одно ограничение, ограничение на размер комплексной модели бизнес-процесса.

Определение 4 (Математическая постановка задачи). Пусть L – это журнал процесса P. Даны три натуральных числа q, m и n, задача обнаружения комплексной *WF*-модели, обозначаемой как $MDP(P, q, m)$, состоит в поиске q-завершенной комплексной *WF*-модели, такой что $n \le |WS| \le m$, где значение $s(WS, L)$ минимально.

Иными словами, мы собираемся предоставить задачу поиска q-завершенной комплексной *WF*-модели WS (с $n \le |WS| \le m$), которая непротиворечива настолько, насколько это возможно. Ниже приводится алгоритм для решения этой проблемы (рис. 2).

Вход: натуральное число maxProps

Выход: WS-модель

Метод:

$CF_{\ddot{u}}(WS_0^1) :=$ minePrecedences(L_p);

$WS^{\vee} := WS_0^1$;

Пока $|WS^{\vee}| < m$ выполнить

$WS_i^j := \text{leastSound}(WS^{\vee})$;

$WS^{\vee} := WS^{\vee} - \{WS_i^j\}$;

$F := \text{searchProps}(L(WS_i^j), q, \text{maxProps}, CF_q)$;

$R(WS_i^j) := \text{проекция}(L(WS_i^j), F)$;

$k := |F|$;

Если $k > 1$ тогда

$j := \max\{ j \mid WS_{i+1}^j \in WS^{\vee} \}$;

$\left\langle WS_{i+1}^{j+1}, ..., WS_{i+1}^{j+k} \right\rangle := \text{k-means}(R(WS_i^j))$;

Для каждого WS_{i+1}^h выполнить

$WS^{\vee} := WS^{\vee} \cup \{WS_{i+1}^h\}$;

$CF := \text{aphaAlgoritm}(L(WS_{j+1}^h))$;

Иначе

$WS^{\vee} := WS^{\vee} \cup \{WS_i^j\}$;

Конец Если;

Конец Пока;

Вернуть $\text{WS}^{\vee}$;

Рис. 2. Алгоритм восстановления комплексных *WF*-моделей

В конечном счете для получения комплексной схемы процесса P мы используем идею последовательного и инкрементального улучшения схемы путем извлечения некоторых глобальных зависимостей, применяемых для кластеризации последовательностей. Алгоритм начинает свое выполнение с построения графа – представления последовательностей. Алгоритмы извлечения таких графов широко представлены в литературе, например в [20].

Каждая *WF*-модель WS_i^j последовательно добавляется в комплексную *WF*-модель $WS^{\vee}$, где i – количество оставшихся шагов, а j – это идентификатор схемы, позволяющий производить различие схем среди схем одного уровня. Кроме того, мы обозначаем через $L(WS_i^j)$ множество последовательностей в кластере, принадлежащих WS_i^j.

Алгоритм использует жадную эвристику: на каждом этапе предпочтение отдается схеме, которая наиболее выгодно может быть улучшена. На практике мы улучшаем в меньшей степени непротиворечивую схему среди уже обнаруженных.

Вход: Журнал выполнения L, величина q, максимальное количество свойств *maxProps*, граф зависимостей CF_q

Выход: Множество минимальных свойств

Метод:

$L_2 := \{[\alpha_1 \to \alpha_2] \mid (\alpha_1, \alpha_2) \in E_q\}$;

$k := 1, R := L_2; F := 0;$

Повтор

$\text{M} := 0; \text{k} := \text{k} + 1;$

Для всех $[\alpha_1 \to ... \to \alpha_j] \in L_k$ выполнить

Для всех $[\alpha_j \to b] \in L_2$ выполнить

Если $[\alpha_i \to ... \to \alpha_j] \not\to b$ не в F, тогда

$M := M \cup [\alpha_i \to ... \to \alpha_j \to b];$

Для всех $p \in M$ выполнить

Если p – это l-частый в L, тогда

$L_{k+1} := \{p\}$

Иначе

$F := F \cup \{[\alpha_i \to ... \to \alpha_j] \not\to b\}$;

Конец Для всех;

$R := R \cup L_{k+1}$;

Пока $L_{k+1} = 0$;

$S' := L; F' := 0;$

Выполнить

$\beta := \text{argmax}_{\beta' \in F} | w(\beta', S') |$

$F' := F' \cup \{\beta\}$;

$S' := S' - w(\beta, S')$;

Пока ($|S'|/|L_p| > q$)&&($|F'|$<maxProps);

Вернуть F';

Рис. 3. Алгоритм «поискСвойств»

Кластеризация осуществляется путем метода *k*-средних, подготовительной фазой для которого выступают процедуры *«поискСвойств»* (обнаружение свойств для кластеризации) и *«проекция»* (для представления последовательностей в удобном для метода *k*-средних формате).

Рассмотрим процедуру *«поискСвойств»* более подробно. В некоторых работах уже сталкивались с подобной задачей [21], однако наша цель – представить алгоритм нахождения таких свойств, которые были бы применимы именно в контексте получения комплексных *WF*-моделей.

Нам важны свойства, которые разделяют журнал наиболее выразительным образом. Для решения этой задачи введем еще ряд определений.

Последовательность задач $[\alpha_1 \to ... \to \alpha_h]$ является *l*-частой в журнале L, если $\frac{\{\alpha \in L \mid \alpha_{i_1}, ..., \alpha_{i_h} \wedge i_1 < ... < i_h\}}{|L|} \geq l$. Мы говорим, что $[\alpha_1 \to ... \to \alpha_h]$ *l*-предшествует α в L, обозначаемой как $[\alpha_1 \to ... \to \alpha_h] \to_l \alpha$, если обе последовательности $[\alpha_1 \to ... \to \alpha_h]$ и $[\alpha_1 \to ... \to \alpha_h \to \alpha]$ являются *l*-частыми.

Определение 5 (Свойства). Свойство β – это выражение вида $[\alpha_1 \to ... \to \alpha_h] \mapsto \alpha$ такое, что: 1) $[\alpha_1 \to ... \to \alpha_h]$ *l*-частое свойство в L; 2) $[\alpha_1 \to ... \to \alpha_h] \to_l \alpha$ не выполняется.

На основе этих определений представим алгоритм поиска наиболее дискриминирующих свойств (рис. 3).

Сначала на каждом этапе k выполнения мы добавляем в L_k все *l*-частые последовательности, чей размер равен k. Этот процесс повторяется до тех пор, пока не будут найдены все *l*-частые последовательности разных размеров. Далее это множество последовательностей фильтруется в соответствии с вводимыми при вызове метода ограничениями.

Представленный метод был реализован в программном прототипе и апробирован на реальных журналах выполнения бизнес-процессов.

2.4. РАЗРАБОТКА АЛГОРИТМА ПОЛУЧЕНИЯ ЧАСТЫХ ПОДПОСЛЕДОВАТЕЛЬНОСТЕЙ

В этом разделе мы рассмотрим проблему поиска наиболее повторяющихся шаблонов (т.е. подграфов в нашей терминологии) в экземплярах бизнес-процессов, заключающуюся в нахождении наиболее повторяющихся фрагментов в бизнес-процессе при его выполнении.

Одной из разновидностей задач поиска частых подпоследовательностей является следующая: какие задачи из модели бизнес-процесса, представленной на рис. 1, выполняются наиболее часто при выполнении задачи *h*? В качестве примера рассмотрим экземпляры на рис. 4.

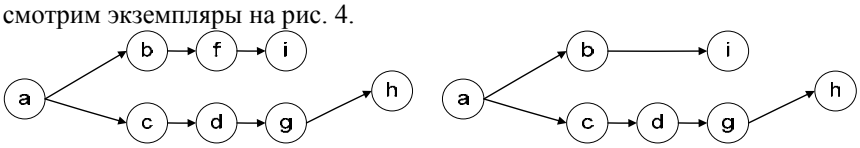

Рис. 4. Экземпляры для бизнес-процесса «Обработка заказа от клиента»

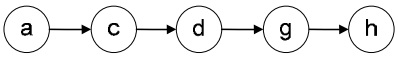

Рис. 5. Подграф для экземпляров с рис. 1

Отметим, что оба графа на рис. 4 являются подграфами первоначальной модели бизнес-процесса, показанного на рис. 5, и удовлетворяют ограничениям, там указанным. Кроме того, оба подграфа соответствуют требованию на содержание узла *h*.

Теперь подграф часто выпадает при выполнении обоих экземпляров и характеризуется выполнением задач *c*, *d*, *g*. В таком случае это может означать, что отказ от обслуживания часто характеризуется недостатком материалов на предприятии. С точки зрения анализа бизнеса становится очевидной необходимость внесения изменений в стратегию управления хранением комплектующих. Сложность решения задачи поиска наиболее частых шаблонов заключается в выборе алгоритма генерации желаемых паттернов путем «умного» изучения области поиска, на которые накладываются ограничения схемы.

При поиске частых подпоследовательностей *WF*-модель выступает в качестве стартовой точки исследования, а не результата: некоторое количество исполнений бизнес-процесса анализируется контекстуально и базируется на этой *WF*-модели с целью найти частые шаблоны выполняемых задач, таким образом исследуя полезную информацию для поддержки принятия решений в организации. С учетом этой перспективы исследование следует сопоставить с другими попытками анализа частых подпоследовательностей. Наиболее известным из этого класса исследований, базирующихся на свойстве *антимонотонности*, считается работа, представленная в [10]. В этой работе представляется алгоритм «*Apriori*»: общая идея заключается в генерации множества кандидатов $k + 1$, комбинируя в подходящем случае размер k множества частых шаблонов и проверяя их частоту. Некоторое обобщение такого подхода в рамках интеллектуального анализа последовательных шаблонов представлено в работе [22].

Совершенно иное исследование проблемы было предложено в работе [23], которое использует метод «*FP-growth*». По существу идея анализа частых шаблонов совпадает с вышеописанным, посредством рекурсивного проецирования базы дан-

ных на частые шаблоны уже найденные и затем комбинирование результатов исследования с проецированной базой данных. Расширение этой идеи для последовательных шаблонов предложено в [24]. Недавняя попытка комбинирования такого метода с алгоритмом Apriori была предпринята в [25].

Как и проблема интеллектуального анализа шаблонов в сложных предметных областях, исследование частых деревьев было произведено в работе [26] (алгоритм «*AGM*»), в то время, как первый алгоритм в этом направлении, базирующийся на свойстве антимонотонности, был предложен в [27]. Отметим, что эта проблема все еще обещает интересные задачи в различных предметных областях, таких как биоинформатика и анализ web-документов.

Например, поиск подпоследовательностей, используемый в «*AGM*», был адаптирован и улучшен в алгоритме «*FSG*» [28], в котором используется «умная» стратегия для маркировки сгенерированных графов с целью уменьшить вычислительные расходы на проверку изоморфности графов. Более того, в настоящее время появляются новые алгоритмы, основанные на методе проекции: «*gSpan*» [29], которые исследуют все подграфы без генерации кандидатов и используют позитивное усечение, в то время как «*CloseGraph*» [30] сокращает количество шаблонов, подлежащих генерации, за счет использования понятия *закрытых* шаблонов, т.е. шаблонов, которые не являются подграфами для других шаблонов с той же поддержкой.

Понятно, что подобные подходы могут быть использованы для решения проблемы поиска наиболее частых последовательностей для *WF*-моделей после адаптации этих алгоритмов с учетом специфики прикладной области применения.

Тем не менее, адаптация вышеописанных методов к поиску частых подпоследовательностей в автоматизированных бизнес-процессах – сложная задача. Действительно, генерация шаблонов посредством таких традиционных подходов не учитывает тех ограничений, которые накладывает сама схема, таких, как взаимоотношения между задачами, синхронизация и параллельное выполнение задач [31 – 33]. В противоположность им алгоритмы, представленные ниже, учитывают ограничения, накладываемые структурой *WF*-модели, а экспериментальные результаты, представленные далее, показывают, что они превосходят традиционные подходы к восстановлению частых подпоследовательностей.

Алгоритм поиска частых подпоследовательностей «f-поиск». Определим необходимые в дальнейшем понятия. Допустим, у нас есть *WF*-модель P и множество экземпляров $F = \{I_1, ..., I_n\}$. Граф $p =< A_p, E_p > \subseteq P$ называется F-шаблоном, если существует $I =< A_I, E_I > \in F$ такое, что $A_p \subseteq A_I$ и p-подграф I, включенный посредством узлов в A_p.

Допустим, $freq(p) = |\{I \in F \mid I \models p\}| / |F|$ – поддержка F-шаблона p. Тогда мы можем дать следующее определение.

О п р е д е л е н и е 6 (Математическая формулировка задачи). $FCPD(\sigma)$: задача поиска всех связанных шаблонов, поддержка которых больше чем σ.

Если подходить к решению этой задачи прямолинейно, то напрашивается алгоритм поиска шаблонов на основе простой генерации прямо связанных подграфов, а затем тестирование в полиномиальном времени: является ли этот шаблон экземпляром для P. Другое предложение основано на идее ослабления количества шаблонов, подлежащих генерации. Для того чтобы добиться этой цели, мы можем рассматривать только графы, которые «закрыты» (с учетом глобальных и локальных ограничений), т.е. такие, что $p \models c$ для всех $c \in C_L \cup C_G$. Будем называть такие графы *ослабленными шаблонами*, или просто w-шаблонами. Формализуем приведенные выше рассуждения.

О п р е д е л е н и е 7. Дана *WF*-модель $p' =< A_{p'}, E_{p'} >$, *детерминистическое закрытие* p ($ws\_закрытие(p)$) определяется, как граф $p' =< A_{p'}, E_{p'} >$ такой, что: 1) $A_p \subseteq A_{p'}$, и $E_p \subseteq E_{p'}$; 2) $a \in A_{p'}$ – *and-join* подразумевает, что для каждого $(b,a) \in E$, $(b,a) \in E_{p'}$ и $b \in A_{p'}$; 3) $a \in A_{p'}$ – *deterministic fork* подразумевает, что для каждого $(a,b) \in E$ с b – *or-join*, $(a,b) \in E_{p'}$ и $b \in A_{p'}$. Более того граф p такой, что $p = ws\_закрытие(p)$ называется $ws\_закрытым$.

О п р е д е л е н и е 8. *Ослабленный шаблон*, или просто *w*-шаблон – это $ws\_закрытый$ граф p, такой что для каждого узла a, $|\{(a,b) \mid (a,b) \in E_p\}| \le OUT_{\max}(a)$.

О п р е д е л е н и е 9. Пусть дана *WF*-модель $WS =< A, E >$, тогда для каждого $a \in A$ граф $p = ws\_закрытие(< \{a\}, \{\} >)$ называется элементарным *w*-паттерном.

Для поиска решения задачи используется алгоритм, который инкрементально строит частые w-шаблоны, стартуя с «*элементарных*» w-шаблонов (описанных ниже) и расширяя каждый частый шаблон max посредством использования двух базовых операций: добавление частой дуги и слияние с другим частым элементарным w-шаблоном.

Элементарные w-шаблоны, с которых начинается построение частых шаблонов, получаются как детерминистическое закрытие отдельных узлов. Шаблон является элементарным w-шаблоном (обозначим через ew-шаблон) для узла a, если это минимальный w-шаблон, содержащий a. Множество всех ew-шаблонов обозначается как EW. Более того, пусть p – w-шаблон, тогда EW_p обозначает множество элементарных w-шаблонов, содержащихся в p. Заметим, что, если дан ew-шаблон e, EW_e не обязательно содержит только один элемент, может содержать другие ew-шаблоны. Кроме того, данное множество $E' \subseteq EW$, $Compl(E') = EW - \bigcup_{e \in E'} EW_e$ содержит все элементарные шаблоны, которые не содержатся ни в E', ни в одном элементе E'.

Теперь введем операцию отношения $\prec$ между связанными w-шаблонами. Определим через $E^{\subseteq}$ множество всех дуг в P, источники которых не являются *and-split*, т.е. $E^{\subseteq} = \{(a,b) \in E \mid OUT_{\min}(a) < OutDegree(a)\}$. Даны два соединенных w-шаблона $p =< A_p, E_p >$ и $p' =< A_{p'}, E_{p'} >$, мы говорим, что $p \prec p'$ тогда и только тогда, когда:

1. $A_p = A_{p'}$ и $E_{p'} = E_p \cup \{(a,b)\}$, где $(a,b) \in E^{\subseteq} - E_p$ и $OUT_{\max}(a) > OutDegree_p(a)$ (т.е. p' может быть получена из p посредством добавления дуги), или

2. Существует $p'' \in Compl(EW_p)$ такой, что $p' = p \cup p'' \cup X$, где X – либо пустое множество, если p и p'' связанные, или содержит дугу в $E^{\subseteq}$ с конечными точками в p и p'' (т.е. p' получено из p путем добавления элементарного w-шаблона и возможно соединяющей дуги).

При работе алгоритма (рис. 6) инкрементально строятся частые w-шаблоны с использование двух базовых операций: добавление частой дуги и слияние с другим частым w-шаблоном.

Элементарные шаблоны, с которых начинается алгоритм, получаются как $ws\_закрытие(p)$ для каждого из узлов. Далее идут вычисления в главном цикле алгоритма, где каждое новое значение для L_{k+1} получается путем расширения любого шаблона p, сгенерированного на предыдущем шаге ($p \in L_k$) двумя способами: 1) добавлением частой дуги из $E^{\subseteq}$ (посредством функции $addFrequentArc$) и 2) добавлением элементарного w-шаблона (функция $AddFrequentEWPattern$). Каждый шаблон p', генерируемый функциями, указанными выше, – допустимый подграф для WS, т.е. для каждой $a \in A_{p'}$, $OutDegree_{p'}(a) \le OUT_{\max}(a)$.

Вход: WF-модель WS, множество экземпляров $F = \{I_1, ..., I_N\}$ для модели WS

Выход: Множество частых F-шаблонов

Метод:

$L_0 := \{e \mid e \in EW, e$ является частым шаблоном с учетом $F\}$;

$k := 0; R := L_0$;

$FrequentArcs := \{(a,b), (a,b) \in E^{\subseteq}, < \{a,b\}, \{(a,b)\}$ является частым с учетом $F >\}$

$E_f^{\subseteq} := E^{\subseteq} \cap FrequentArcs$;

Повтор

$U := 0$;

Для всех $p \in L_k$ выполнить

$U := U \cup$ addFrequentArc(p);

Для всех $e \in Compl(EW_p) \cap L_0$ выполнить

$U := U \cup$ addFrequentEWPattern(p , e);

Конец Для всех;

$L_{k+1} := \{p \mid p \in U, p$-частый с учетом $F\}$;

$R := R \cup L_{k+1}$;

Пока $L_{k+1} = 0$;

Вернуть R;

Процедура $addFrequentEWPattern(p =< A_p, E_p >, e =< A_e, E_e >)$: w-шаблон;

$p' :=< A_p \cup A_e, E_p \cup E_e >$;

Если p' связанный, тогда Вернуть p'

Иначе Вернуть $addFrequentConnection(p', p, e)$;

Процедура $addFrequentConnection(p' =< A_{p'}, E_{p'} >, p =< A_p, E_p >, e =< A_e, E_e >$

$S := 0$;

Для всех $frequent(a,b) \in E_f^{\subseteq} - E_p (a \in A_p, b \in A_e) \vee (a \in A_e, b \in A_p)$ выполнить

$q :=< A_{p'}, E_{p'} \cup (a,b) >$;

Если $WS \models q$, тогда $S := S \cup \{q\}$;

Рис. 6. Алгоритм поиска частых подпоследовательностей «f-поиск»

Конец Для всех;

Вернуть S;

Процедура $addFrequentArc(p =< A_p, E_p >)$: шаблон

$S := 0$;

Для всех $frequent(a,b) \in E_f^{\subseteq} - E_p, a \in A_p, b \in A_p$ выполнить

$p' :=< A_p, E_p \cup (a,b) >$;

Если $WS \models p'$, тогда $S := S \cup \{p'\}$;

Конец Для всех

Вернуть S;

Рис. 6. Продолжение

Алгоритм поиска частых подпоследовательностей («s-поиск»). Заметим, что при поиске частых подпоследовательностей можно использовать другую стратегию: итеративно генерировать кандидата на *n*-м уровне путем слияния кандидатов на *j*-м и (*n* – *j*)-м уровне, соответственно. Ясно, что в худшем случае (для *j* = *n* – 1) мы получим алгоритм поиска, представленный в предыдущем разделе, иначе, в самом лучшем случае, поиск сходится экспоненциально меньшим количеством итераций. Также ясно, что нам необходимы дополнительные усилия для идентификации компонент для слияния. Грубо говоря, эти компоненты должны быть таковы, что их границы могут быть сопоставлены. Под границей понимается множество узлов, которые допускают входящую или исходящую дугу.

Формализуем приведенные выше рассуждения.

Определение 10. $ВХ\_ГРАНИЦА(p) = \{a \in A_p \mid InDegree_p(a) < < InDegree(a)\}$ – множество узлов в шаблоне p, которые допускают дальнейшие входящие дуги, $ИСХ\_ГРАНИЦА(p) = \{a \in A_p \mid \mid OutDegree_p(a) < OUT_{\max}(a)\}$ – множество узлов в шаблоне p, которые допускают наличие исходящих дуг. Множества $ВХ\_ГРАНИЦА(p)$ и $ИСХ\_ГРАНИЦА(p)$ представляют входящие и исходящие границы шаблона p, т.е. множество узлов внутри p, которые могут быть «достигнуты» другими узлами извне. Заметим, что по определению входящая граница для w-шаблона не может содержать *and-join* задачи. Аналогично, исходная граница не может содержать *deterministic fork*.

Границы могут быть использованы для соединения компонент. Так как дуга соединяет границы двух компонент, достаточно уделить все внимание частым дугам и итеративно генерировать новые кандидаты посредством слияния частых компонент, чьи границы соединены посредством этих дуг.

На основе этих идей был разработан алгоритм («s-поиска»), подробности которого приводятся на рис. 7.

Вход: *WF*-модель WS, множество экземпляров $F = \{I_1, ..., I_N\}$ для модели WS

Выход: Множество частых F-шаблонов

Метод:

$R := \{e \mid e \in EW, e$ является частым шаблоном с учетом $F\}$; $\Delta R := R$;

Для всех $(a,b) \in E$ выполнить $connected\_by(a,b) = \varnothing$;

Повторить

Для всех $a \in A$ выполнить

$INF(a) := \{p \in R \mid a \in$ ВХ_ГРАНИЦА(p)$\}$, $INFP(a) = \varnothing$;

$OUTF(a) := \{p \in R \mid a \in$ ИСХ_ГРАНИЦА(p)$\}$, $OUTFP(a) = \varnothing$;

Конец Для всех;

$FA := \{(a,b) \mid (a,b)$ является частой дугой с учетом F, $OUT(a) \neq \varnothing$, $INF(b) \neq \varnothing\}$;

$FP := \{p \cup q \mid p \cap q \neq \varnothing,\ p \in R, q \in \Delta R, WS \models p \cup q\}$;

Для всех $(a,b) \in FA$ выполнить

Для всех $p_1 \in OUTF(a), p_2 \in INF(b)$ с учетом $(a,b) \notin p_1 \cup p_2$ и $(p_1, p_2) \notin connected\_by(a,b)$ выполнить

$q := p_1 \cup p_2 \cup \{(a,b)\}$;

Если $WS \models q$, тогда

$FP : FP \cup \{q\}$;

ВХ_ГРАНИА(q):= $ComputeInBorder(b, p_1, p_2)$;

ИСХ_ГРАНИА(q):= $ComputeOutBorder(a, p_1, p_2)$;

Для всех $a \in$ ВХ_ГРАНИЦА(q) выполнить $INFP(a) := INF(a) \cup \{q\}$;

Для всех $a \in$ ИСХ_ГРАНИЦА(q) выполнить $OUTF(a) := OUTFP(a) := OUTFP(a) \cup \{q\}$;

$connected\_by(a,b) := connected\_by(a,b) \cup \{(p_1, p_2)\}$

Конец Если;

Конец Для всех;

$\Delta R := \{p \in FP \mid p$ -частый с учетом $F\}$;

$R := R \cup \Delta R$;

Пока $\Delta R = \varnothing$;

Рис. 7. Алгоритм поиска частых подпоследовательностей «s-поиск»

Вернуть R;

Процедура $ComputeInBorder(b, p_1, p_2)$;

Если $\mid InDegree_{p_1 \cup p_2}(b) \mid < InDegree(b) - 1$, тогда

ВХ_ГРАНИЦА := $\{b\}$;

Иначе _ГРАНИЦА := $\varnothing$;

Для всех $c \in$ (ВХ_ГРАНИЦА(p_1) $\cup$ ВХ_ГРАНИЦА(p_2))- $\{b\}$ выполнить

Если $|InDegree_{p_1 \cup p_2}(c)| < InDegree(b)$, тогда ВХ_ГРАНИЦА:= ВХ_ГРАНИЦА $\cup \{c\}$;

Вернуть ВХ_ГРАНИЦА;

Процедура $ComputeOutBorder(a, p_1, p_2)$;

Если $|OutDegree_{p_1 \cup p_2}(a)| < OUT_{\max}(a)$ -1, тогда ИСХ_ГРАНИЦА := {a}, тогда ИСХ_ГРАНИЦА:= $\varnothing$;

Для всех $c \in$ (ИСХ_ГРАНИЦА(p_1) $\cup$ ИСХ_ГРАНИЦА(p_2))-{ a } выполнить

Если $|OutDegree_{p_1 \cup p_2}(c)| < OUT_{\max}(a)$, тогда ИСХ_ГРАНИЦА:= ИСХ_ГРАНИЦА $\cup \{c\}$;

Вернуть ИСХ_ГРАНИЦА;

Рис. 7. Продолжение

Ядром алгоритма является главный цикл, в котором для каждого узла $a \in WS$ множества $INF(a)$ (или $OUTF(a)$) F-шаблонов, содержащих a на входе (или выходе), вычисляются границы. Далее переменные FA и FP заполняются значениями частых дуг, которые могут соединить паттерны, и кандидатами, которые могут быть получены составлением «совместимых» шаблонов. Далее границы пересчитываются для новых F-паттернов, и частые F-паттерны идентифицируются вычислением частоты каждого кандидата. Заметим, что границы для кандидата F-паттерна могут быть инкрементально вычислимы посредством расширения границ связанных компонент, и новые кандидаты могут быть генерированы путем слияния F-паттернов, открывая доступ к некоторым узлам. Алгоритм за-вершает работу, когда количество кандидатов равно нулю, т.е. когда вычисленные паттерны имеют пустые входные и выходные границы.

В следующем разделе приводится обзор производительности алгоритма и его сравнение с алгоритмом «f-поиска».

Экспериментальные данные работы алгоритмов. В этом разделе рассматривается поведение алгоритмов «f-поиск» и «s-поиск» с точки зрения изменения производительности и расширяемости. Количество кандидатов w-шаблонов может быть чрезмерно большим, большим настолько, что сделает недоступным использование алгоритмов на больших и сложных WF-моделях. Более того, производится сравнение производительности реализаций представленных алгоритмов с различными существующими техниками вычисления частых шаблонов, применимых в данной предметной области.

В наших экспериментах используются синтетические данные. Генерация данных для экспериментов может быть настроена следующим образом: 1) размер WS; 2) размер F; 3) размер $|L|$ частых ослабленных паттернов в F; 4) вероятность $p_{\subseteq}$ выбора $E^{\subseteq}$-дуги.

Все тесты производились на компьютере Intel Pentium IV 1600 Mhz/1Gb с операционной системой *Windows XP Professional SP2*.

Производительность алгоритма «f-поиск». При анализе производительности алгоритма использовалось несколько фиксированных WF-моделей, для которых генерировалось различное количество экземпляров.

В частности недетерминированные ветвления выполнялись в соответствии с биномиальным распределением $p_{\subseteq}$. Частые экземпляры вставлялись в результирующий журнал посредством замещения случайных экземпляров в соответствии с параметром $|L|$. На рис. 8 показано количество операций (сопоставления паттерна экземпляру) для различного числа $|F|$. Рисунок показывает, что алгоритм имеет линейную зависимость от размера входных данных (для различной поддержки).

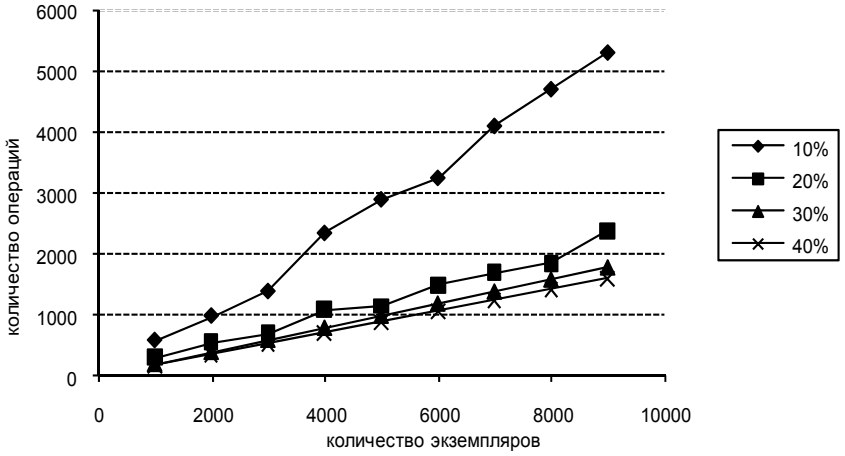

Рис. 8. Производительность алгоритма «f-поиск»: количество операций при различных количестве экземпляров и пороге уровня поддержки

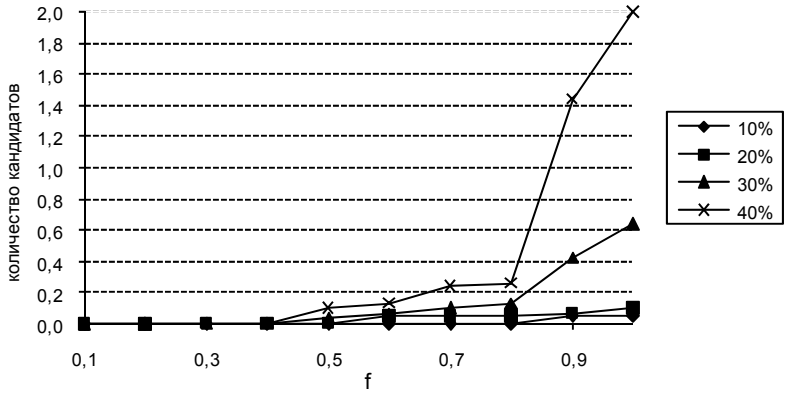

Рис. 9. Производительность алгоритма «f-поиск»: количество кандидатов при различных значениях коэффициента *f* и порога уровня поддержки. При $|F|=5000,\ |L|=2$

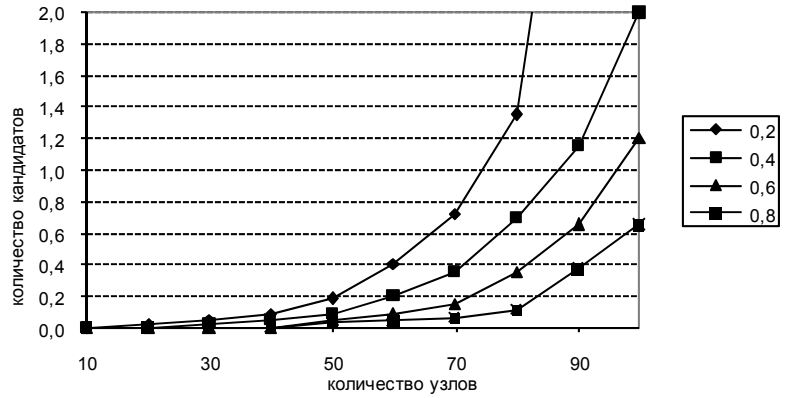

Рис. 10. Производительность алгоритма «f-поиск»: количество кандидатов при различных количестве узлов и значениях коэффициента *f*. При $|F|=5000,|L|=2$

Во втором множестве экспериментов (рис. 9) производилась случайная генерация *WF*-моделей с целью проверить эффективность работы алгоритма (с учетом структуры *WF*-модели). Для этой цели мы зафиксировали $|F|$ и сгенерировали экземпляры с учетом случайно полученной *WF*-модели. Количество узлов и дуг выбиралось на основе пуассонова распределения с фиксированным среднем значением. Для того чтобы оценить уровень сложности *WF*-моделей был введен коэффициент $f=\frac{|F^{\subseteq}|}{|E|}$, который отображает степень потенциального недетерминизма внутри *WF*-модели. Интуитивно понятно, что *WF*-модели с f, близким к 0, порождают экземпляры с малым числом кандидатов *w*-шаблонов. Наоборот *WF*-модели с f, близким к 1, порождают экземпляры с большим числом кандидатов *w*-шаблонов. Рис. 9 показывает поведение алгоритма «*f-поиск*», когда f меняется от 0 (нет детерминизма) до 1 (полный детерминизм), для различных значений *minSupp*. Интересно наблюдать, что для больших значений f (реальные *WF*-модели имеют коэффициент детерминизма меньше 0,5) поле поиска радикально увеличивает количество предполагаемых кандидатов.

И, наконец, на рис. 10 представлена зависимость между количеством потенциальных кандидатов и количеством узлов для различных значений коэффициента f.

Сравнение алгоритма «*f-поиск*» *с алгоритмом* «*Apriori*». Для анализа использовался алгоритм «*Apriori*» [10], который был адаптирован для работы с журналами выполнения МАБП. Такое сравнение важно для анализа сверхзатрат ресурсов традиционными алгоритмами, которые могут быть легко настроены на поиск частых подпоследовательностей, но не могут обладать информацией по специфике предметной области. Ниже приведено описание нескольких экспериментов, сравнивающих производительность алгоритмов «*Apriori*» и «*f-поиск*» при нарастающих значениях $|F|$ и *minSupp*. Данные генерировались посредством описанной ранее методики для схемы на рис. 4, результаты представлены на рис. 11 и 12.

Как и ожидалось, алгоритм «*f-поиск*» значительно превосходит «*Apriori*» (см. рис. 11). Что происходит из-за отсутствия у алгоритма «*Apriori*» дополнительной информации в виде *WF*-модели (в отличие от «*f-поиск*»).

Графики на рис. 12 показывают поведение алгоритмов для изменяющегося значения f (от 0 – нет детерминизма, до 1 – полный детерминизм). Заметим, что для малых значений f оба алгоритма производят малое количество кандидатов, однако в этой ситуации «*f-поиск*» все еще значительно превосходит «*Apriori*» на малых значениях *minSupp* (0,1).

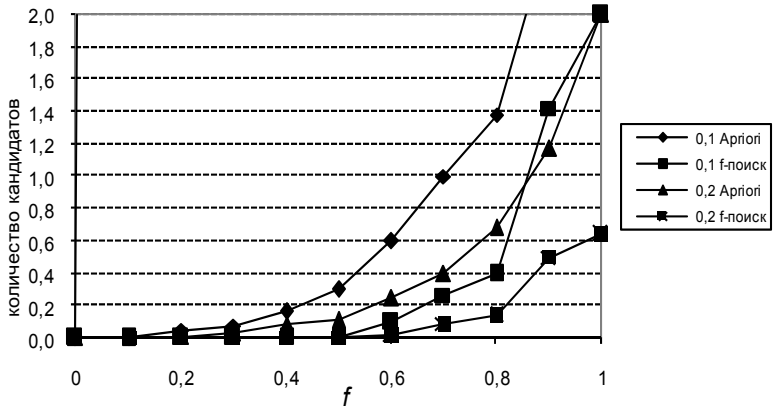

Рис. 11. Сравнение алгоритмов «Apriori» и «f-поиск»: количество кандидатов при различных значениях коэффициента *f* и порога минимальной поддержки. При $|F| = 5000, |L| = 2$

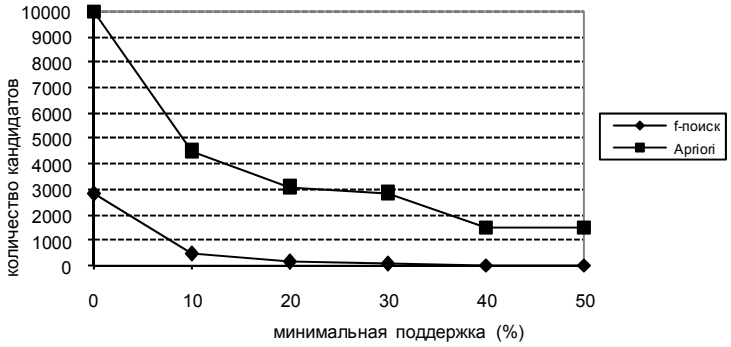

Рис. 12. Сравнение алгоритмов «Apriori» и «f-поиск»:
количество кандидатов при различном уровне порога минимальной поддержки. При $|F| = 5000, |L| = 2$

Действительно, для малых значений *minSupp* количество кандидатов существенно уменьшается, и поэтому стратегия, используемая *«f-поиск*», ведет к преимуществу. Однако обратим внимание, что подобная адаптивная модификация алгоритма «*Apriori*» является действенным решением для поиска частых подпоследовательностей в мало детерминированных *WF*-моделях, если нас интересуют сильночастые шаблоны (*minSupp* > 0,2).

Сравнение алгоритмов «*f-поиск*» *и* «*s-поиск*». Сравнение было сделано между двумя алгоритмами «*s-поиск»* и «*f-поиск*», результаты этого сравнения показаны на рис. 13 и 14. В первом эксперименте коэффициент f был зафиксирован и равен 0,7, при этом количество генерируемых экземпляров равно 5000. На рис. 13 показаны зависимости производительности алгоритмов «f-поиск» и «s-поиск». Следует отметить, что алгоритм «*f-поиск*» значительно превосходит «*s-поиск*» на маленьких значениях минимальной поддержки.

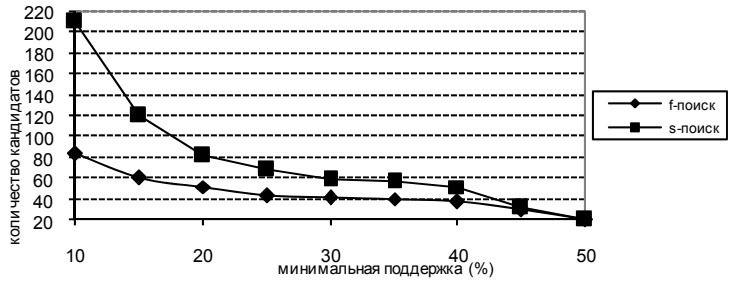

Рис. 13. Сравнение производительности алгоритмов «f-поиск» и «s-поиск»: количество кандидатов для различного порога минимальной поддержки. При $|F| = 5000, f = 0,7$

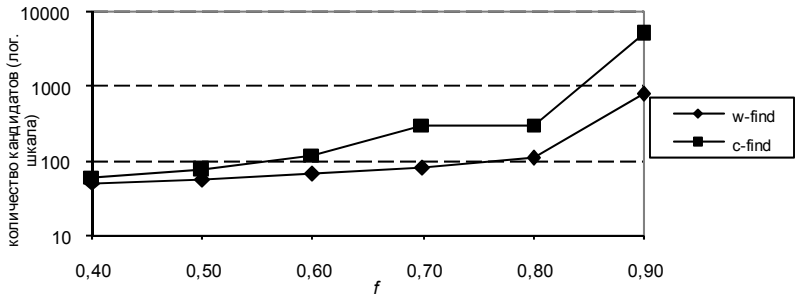

Рис. 14. Сравнение производительности алгоритмов «f-поиск» и «s-поиск»: количество кандидатов для различных значений коэффициента f. При $|F| = 5000, \min Supp = 0{,}2$

Во втором эксперименте (рис. 14) значение *minSupp* фиксировалось равным 0,2 и сравнение делалось для различных значений f. Этот эксперимент подтвердил способность алгоритма «*f-поиск*» генерировать меньшее количество кандидатов для разного типа *WF*-моделей (моделей с разным уровнем детерминизма).

Фактор, который поможет положительно оценить производительность алгоритма «*s-поиск*», – количество выполняемых шагов. Рис. 14 показывает количество кандидатов, генерируемое на разных шагах выполнения алгоритма s-поиска. На рис. 15 поведение алгоритмов «*s-поиск*» и «*f-поиск*» сильно различается. Алгоритм «*s-поиск*» генерирует на каждом этапе все больше кандидатов, что стимулирует быструю сходимость. Алгоритм «*f-поиск*», напротив, после некоторого центрального момента, когда количество кандидатов максимально, с каждым новым шагом генери-

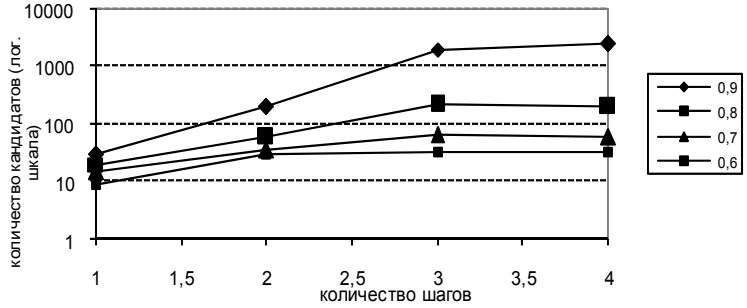

Рис. 15. Алгоритм «s-поиск»:
количество кандидатов на различных этапах выполнения. $|F| = 5000, \min Supp = 0{,}2$

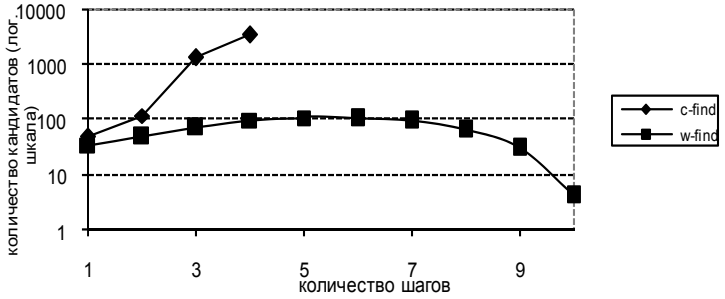

Рис. 16. Сравнение производительности алгоритмов «f-поиск» и «s-поиск»:
количество кандидатов на различных этапах выполнения. $|F| = 5000, f = 0{,}9$

рует заметно меньшее количество кандидатов, требуя большего времени выполнения. Рис. 16 наглядно представляет, что более быстрая сходимость алгоритма «*s-поиск*» окупается за счет большего количества генерируемых кандидатов. Очевидно, что количество шагов связано с количеством обращений к базе данных. При больших объемах базы данных алгоритм «*s-поиск*» становится все более выгодным.

Также алгоритм «*s-поиск*» показал себя более продуктивным на *плотных базах журналов*. Под плотной базой понимается база, в которой количество частых паттернов достаточно велико (сравнительно с размерами базы). Если база экземпляров обладает такой особенностью, то количество генерируемых кандидатов сопоставимо с количеством частых шаблонов (количество нечастых шаблонов относительно частых шаблонов мало). В этом случае оба алгоритма генерируют схожие множества кандидатов. Однако стратегия «постпросмотра» гарантирует алгоритму «*s-поиск*» более быстрый порядок сходимости.

2.5. ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ СИСТЕМЫ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ВЫПОЛНЕНИЯ БИЗНЕС-ПРОЦЕССОВ В СИСТЕМЕ ЭЛЕКТРОННОГО ДОКУМЕНТООБОРОТА

Интеллектуальный анализ выполнения бизнес-процессов в системе электронного документооборота (СЭД) – молодое направление, обещающее интересные задачи и их решения. Авторами были разработаны три алгоритма: алгоритм восстановления комплексных МАБП и два различных алгоритма поиска частых подпоследовательностей.

Алгоритм восстановления комплексных МАБП предоставляет аналитику возможность не только получить МАБП на основе журнала выполнения бизнес-процесса, но и увидеть наиболее типичные варианты выполнения бизнес-процесса, отражающие зачастую бизнес-правила принятия решений. Подобный инструмент может улучшить принимаемые аналитиком решения на этапе анализа бизнес-процессов, помочь новым сотрудникам в понимании новых для них бизнес-процессов, проливает свет на имеющиеся в организации взаимодействия.

В контексте дальнейшего развития алгоритмов возникают интересные задачи, решение которых только предстоит найти. Например, задача обнаружения таких дискриминантных правил, которые характеризуют неудачное или, напротив, удачное завершение бизнес-процесса, или какой выбор ведет к наиболее желанному завершению бизнес-процесса.

Алгоритмы, предложенные для поиска частых подпоследовательностей при выполнении МАБП, по существу являются адаптацией традиционных алгоритмов поиска частых подпоследовательностей к применению в более структурированной предметной области, где имеется дополнительная информация о природе исследуемых последовательностей. В частности, это адаптация алгоритма «*Apriori*» [10] для работы с данными WFMS.

Кроме того, произведено тестирование алгоритмов поиска частых подпоследовательностей, где было подтверждено их превосходство над традиционными подходами поиска.

Дальнейшее развитие алгоритмов поиска частых подпоследовательностей наблюдается в применении техник без генерации кандидатов [34].

При реализации этих алгоритмов в системе интеллектуального анализа выполнения бизнес-процессов необходимо учитывать их требования к уровню производительности при доступе к журналам выполнения, необходимость доступа к функционалу, реализующему другие алгоритмы восстановления МАБП, взаимодействие с редактором МАБП (подсветка частых подпоследовательностей в МАБП), сохранение вычисленных данных для дальнейшего анализа.

Описанные алгоритмы (как разработанные в рамках этой работы, так и разработки других авторов, указанные в первой главе) предъявляют жесткие требования к реализующей их системе:

– Скорость доступа к журналам выполнения бизнес-процессов без снижения производительности СЭД, при этом данные должны находиться в актуальном состоянии.

– Наглядность работы алгоритмов для конечного пользователя (например, выделение особым цветом в МАБП наиболее частых подпоследовательностей).

– Возможность конвертации МАБП в различные нотации (EPC, IDEF3, Control Flow, Activity Diagrams и т.п.).

– Возможность ввода уникальных параметров алгоритмов.

– Сопоставление журнала выполнения и версионности МАБП.

Реализация всех этих требований при сохранении приемлемой производительности конечной системы – сложная задача, решая которую, необходимо учитывать предоставляемую конечной архитектурой возможность легкого добавления новых методов и алгоритмов в систему интеллектуального анализа.

Функциональное описание системы интеллектуального анализа. Наиболее общий взгляд, с точки зрения пользователя, на систему интеллектуального анализа выполнения бизнес-процессов в управлении электронным документооборотом может дать диаграмма использования, которая отображает в разрезе СЭД основные предоставляемые функции.

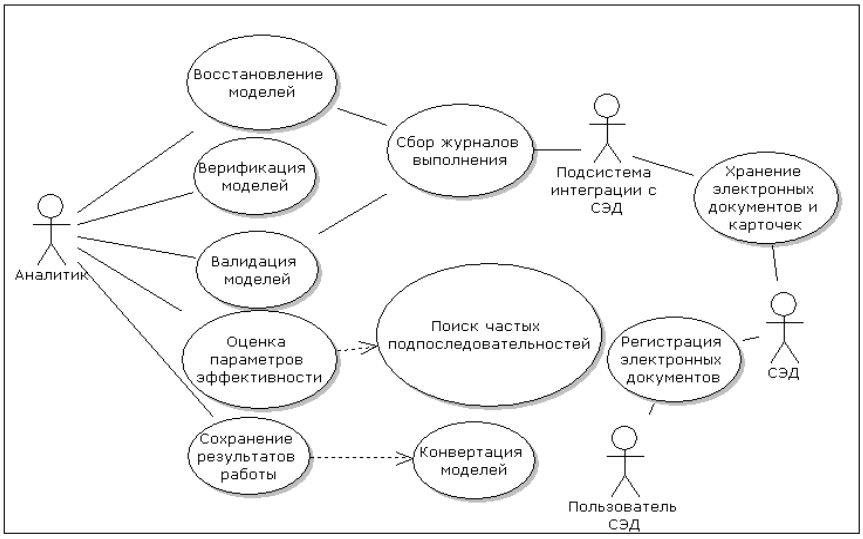

Рис. 17. Диаграмма использования системы интеллектуального анализа бизнес-процессов в электронном документообороте

На рис. 17 показана такая диаграмма использования. Как видно из рисунка, основным действующим лицом при использовании системы выступает аналитик. Под аналитиком понимается роль, выполнять которую могут различные сотрудники отдела, выполняющие внедрение WFMS на базе СЭД. При обычном течении процесса аналитик осуществляет моделирова-

ние процесса вручную, рисуя схему в редакторе бизнес-процессов. С использованием системы интеллектуального анализа бизнес-процесса у него появляются качественно новые возможности, посредством следующих функций:

- Восстановление моделей бизнес-процессов. На основе журналов бизнес-процессов, которые хранятся в локальной БД, аналитик может получать восстановленные МАБП, которые в дальнейшем можно редактировать во встроенном редакторе. Такой интегрированный подход к формированию МАБП позволяет сократить временные затраты на переключение между внешним редактором моделей и программой интеллектуального анализа.

Восстановление моделей может осуществляться с использованием нескольких алгоритмов, доступ к которым предоставляется аналитику [136]. Часто в результате работы таких алгоритмов получаются различные модели. Выявление различий программным образом ускоряет процесс разработки конечной версии МАБП.

- Верификация. Восстановленную модель необходимо проверить на корректность в соответствии с заданными критериями корректности, в зависимости от WFMS, для которой предназначаются эти МАБП.
- Валидация. Аналитик должен ясно представлять, какие из этапов в полученной МБАП наиболее адекватны реальному выполнению бизнес-процесса.
- Сбор журналов выполнения. Ввиду различия конкретных СЭД и, в частности, представления информации в них необходимо разработать методы получения журналов выполнения бизнес-процессов для каждой из систем в отдельности.
- Оценка параметров эффективности. Помимо МАБП, на основе журналов выполнения бизнес-процессов можно получить большое количество дополнительной, полезной информации: о возможности циклического взаимодействия между руководителем и подчиненным, получения социальных сетей, оценки загруженности сотрудников в разные периоды времени и т.п.
- Поиск частых подпоследовательностей. Позволяет выявить наиболее часто используемые маршруты, по которым работает бизнес-процесс. Анализ доступен за некоторый заданный период времени. Подобная функциональность может прояснить многие процессы, происходящие внутри организации, глобальные зависимости в бизнес-процессе.
- Сохранение результатов работы. Система должна предоставлять возможность возобновить работу без потери времени пользователя.
- Конвертация моделей. Различные WFMS работают с использованием зачастую уникальных нотаций отображения бизнес-процессов, таких, как EPC, сети Петри, IDEF3 и пр. Предоставление средств конвертации моделей позволяет не только создавать модели для конкретной WFMS, но и переносить модели из одной WFMS в другую.
- Редактирование МАБП. Наиболее используемый функционал, недостатки работы которого резко снижают применимость всей системы.
- Сравнение МАБП. Дополнительный функционал, позволяющий найти отличия в двух моделях, графически выделяя их.

Сбор журналов бизнес-процессов также логически относится к системе интеллектуального анализа [35]. Алгоритм сбора реализуется индивидуально для каждой системы электронного документооборота в отдельности, предоставляя лишь общий интерфейс для взаимодействия с системой интеллектуального анализа. Подсистема интеграции взаимодействует с системой электронного документооборота через этот плагин.

На СЭД лежит функция по регистрации и хранению документов (рис. 18). Предполагается, что информация о движении документов между пользователями хранится в базе данных системы в особом формате. Для каждого типа документа существует уникальный набор атрибутов, присутствующих в регистрационной карточке. Набор атрибутов определяется на стадии внедрения системы электронного документооборота, кроме того, количество различных типов документов в системе постоянно меняется – появляются новые, исчезают некоторые старые и т.п., иными словами, это динамическая эволюционирующая система.

Процесс регистрации нового типа документа описан на рис. 19 и состоит из следующих этапов.

- Определение списка атрибутов документа нового типа. Определение необходимых атрибутов производится аналитиком (администратором записей) на основе исследования заявки на создание нового типа документа. В данном случае достаточно популярным является подход последовательного улучшения карточки путем чередующихся этапов моделирования карточки и консультаций с реальными пользователями этого типа документа. В качестве результата данного этапа получаем список параметров, которые должны быть внесены еще на этапе регистрации документа.
- Составление дизайна регистрационной карточки. Здесь происходит механическое воплощение предыдущего этапа с помощью редактора регистрационных карточек документов.
- Написание скрипта регистрационной карточки и связывание с необходимыми шаблонами. Часто, помимо регистрационной карточки у документа еще есть и шаблон, на основе которого он генерируется (т.е. в карточке заполняются ее атрибуты, которые автоматически вставляются в шаблон в прописанные предварительно места). Получаемый таким образом документ заметно выигрывает в сравнении с создаваемым вручную – в нем не возможны опечатки, расхождения с данными в БД, получаемые документы единообразны и т.п.

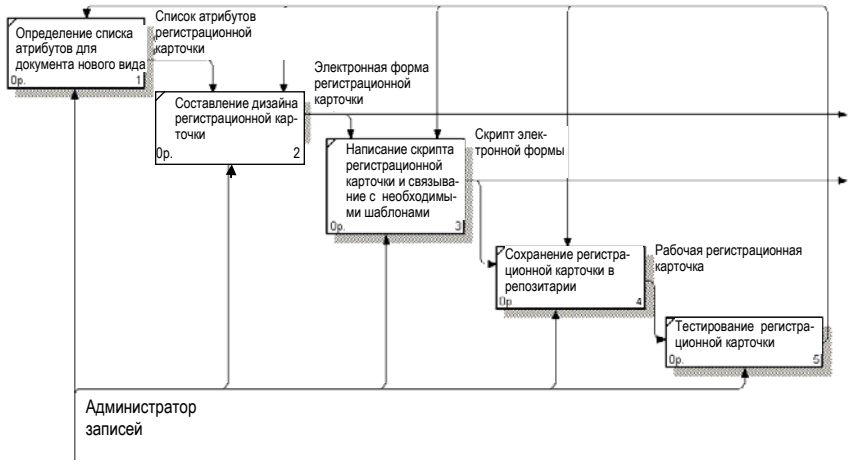

Рис. 18. Разработка регистрационных карточек электронных документов

- Сохранение регистрационной карточки в репозитории. Дальше регистрационная карточка должна быть сохранена в БД, должна быть предоставлена возможность сохранения различных версий карточек, редактирования, поиска нужных карточек.
- Тестирование регистрационной карточки. Перед промышленным использованием регистрационной карточки ее необходимо тщательно протестировать. Часто тестирование регистрационной карточки и ее создание осуществляются разными людьми, что повышает вероятность ошибок.

Воздействие системы интеллектуального анализа на бизнес-процессы СЭД. Внедрение и использование системы интеллектуального анализа не приводят к изменению бизнес-процессов работы с СЭД. Появляется лишь новый инструментарий для выполнения задач в рамках бизнес-процесса (рис. 19).

Рассмотрим подробнее качественные изменения, происходящие внутри бизнес-процесса.

Кроме нового типа документа и его регистрационной карточки разрабатывается маршрут (МАБП), по которому должны двигаться документы данного типа. Процесс разработки показан на рис. 19, он состоит из следующих этапов.

– Разработка нового маршрута для документа. МАБП должен быть связан с пользователями, которые вовлечены в бизнес-процесс, с другими типами документов.

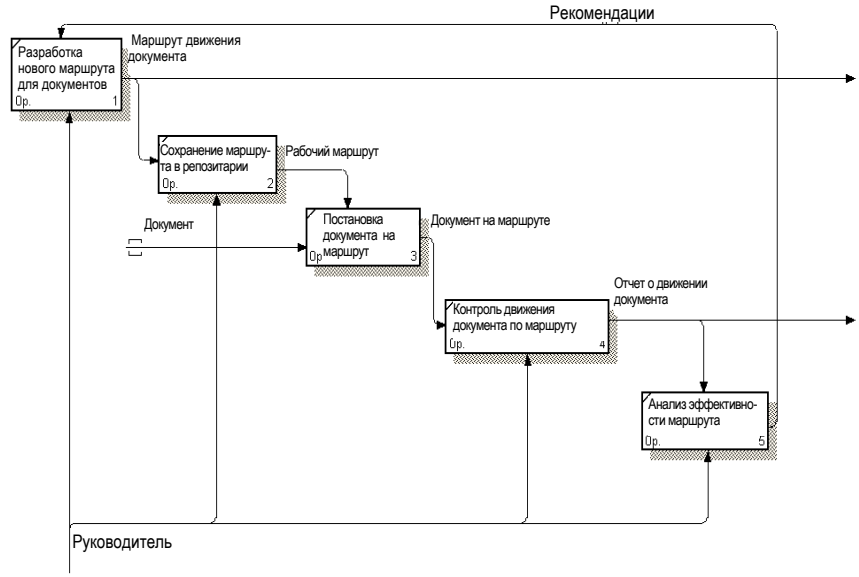

Рис. 19. Управление маршрутами движения

– Сохранение маршрута в репозитории. Как и регистрационная карточка, маршрут должен быть сохранен в репозитории для возможности дальнейшей модификации его аналитиком. Современная СЭД поддерживает версионность маршрутов.

– Постановка документа на маршрут. Аналитик при редактировании маршрута должен иметь возможность связать его с той или иной регистрационной карточкой документа.

– Контроль движения документа по маршруту. При движении документа по маршруту возможность выявления, на каком этапе он находится, и при необходимости воздействия на это движение должна входить в функционал системы управления электронным документооборотом.

– Анализ эффективности маршрута. После введения маршрута в эксплуатацию по нему накапливается некоторая статистика, анализ которой может значительно улучшить бизнес-процесс. На этапе происходит использование системы интеллектуального анализа, которая качественно улучшает выполнение этого пункта посредством предоставления информации об актуальном положении вещей. Наиболее частыми недостатками являются движение документа через начальника другому подчиненного (здесь нужно применить движение от подчиненного к подчиненному), направление документа не тому сотруднику, перегруженность некоторых сотрудников документами и т.п.

Пользователь СЭД производит регистрацию документов в системе электронного документооборота для того, чтобы передать документы быстрым способом от одного пользователя к другому, сохранить документы в БД СЭД с дальнейшей возможностью их найти, с целью генерации отчетов по документообороту и т.п.

Разработка архитектуры системы интеллектуального анализа. Разрабатывая архитектуру системы [36], следует учесть требования к системе, указанные выше. Результирующая архитектура не должна являться ограничением при добавлении новых алгоритмов в систему, вариация количества алгоритмов системы позволяет увеличить гибкость, снизить стоимость владения и является дополнительным стимулом к внедрению.

Проектирование взаимодействия компонент системы. Центральным моментом в реализации архитектуры системы [36] является реализация ее на идеологии плагинов, адаптированной под специфику интеллектуального анализа.

Разработанная система интеллектуального анализа предоставляет возможность добавлять новые алгоритмы восстановления МАБП к уже существующим, что возможно благодаря реализации алгоритмов за пределами самой платформы [37]. Для пользователя доступ к этим алгоритмам осуществляется посредством пунктов главного меню, количество которых меняется в зависимости от количества сборок, представленных в системе на текущий момент. Сборки, содержащие алгоритмы, представляются в виде «черного ящика» с четко определенным внешним интерфейсом и подключаются независимо от других сборок [38]. При выполнении алгоритма восстановления МАБП управление полностью передается основному вычислительному потоку приложения, что накладывает жесткие требования по безопасности и производительности при реализации [39]. Запуск сборок в отдельном вычислительном потоке является единственным средством защиты системы интеллектуального анализа от некорректной реализации.

Поддержка выполнения алгоритмов восстановления МАБП со стороны платформы заключается в предоставлении интерфейсов для работы с бизнес-процессами в базовом формате (*WF*-сети), алгоритмов работы с сетями Петри, алгоритмов работы с эвристическими сетями, с записями журналов выполнения, средствами подсчета статистики в журналах выполнения бизнес-процессов, средствами конвертации бизнес-процессов, предоставление средств для чтения и конвертации журналов из разных форматов.

Сборки NET хранятся в подпапке рабочей папки системы интеллектуального анализа вместе с конфигурационными ini-файлами (рис. 20), степень вложенности этих папок не ограничена. Ini-файл содержит в себе секции, описывающие тип плагина (алгоритм восстановления МАБП, алгоритм валидации, алгоритм верификации и т.п.).

При загрузке системы интеллектуального анализа выполнения бизнес-процессов ее подсистема «Загрузчик плагинов» (см. рис. 20) производит поиск всех доступных на текущий момент сборок, загружает их в слу-

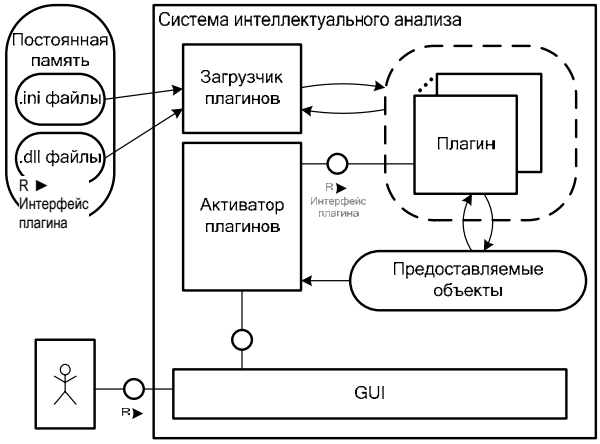

Рис. 20. Загрузка плагинов в среду

чае корректной реализации всех интерфейсов, игнорируя загрузку некорректных плагинов. Для этого система интеллектуального анализа передает в загрузчик путь на диске к сборкам NET, получаемый ею из конфигурационного файла приложения.

Плагин реализует интерфейс, вызывая который, система интеллектуального анализа показывает диалоговое окно (GUI) с параметрами работы алгоритма. Более того, блок GUI (рис. 21) включает в себя изменение главного меню для предоставления возможности вызова алгоритма. В плагине содержатся: описания главного и контекстных меню, ресурсные строки для диалоговых окон, классы, реализующие предписанные интерфейсы. Возможность вызова алгоритма определяется на основе ряда параметров: доступ к хранилищу журналов бизнес-процессов, наличие модели бизнес-процесса в редакторе, изменения в модели с момента предыдущего вызова. Наличие этих условий проверяется подсистемой «Активатор плагинов» посредством логики, включенной в сам плагин (рис. 20). При успешной проверке условий корректности «Активатор плагинов» создает экземпляры классов по заданному интерфейсу. Одновременно в системе присутствует множество плагинов, для каждого из которых создается свой набор экземпляров объектов, используемых всей системой. Решение о создании нового экземпляра принимает пользователь посредством вызова команды меню (т.е. пользователь может воздействовать на этот процесс исключительно посредством GUI, см. рис. 20).

Реализация системы интеллектуального анализа выполнения движения документов в системе электронного документооборота, основанной на использовании идеологии плагинов, обеспечивает ей гибкость, возможность внесения нового функционала, дополнительных алгоритмов, расширение и изменения существующих алгоритмов без модификации платформы, что особенно выгодно при относительно новой и интенсивно развивающейся предметной области.

Работа компоненты восстановления МАБП в системе интеллектуального анализа. Ключевая компонента в системе интеллектуального анализа – компонента с алгоритмами восстановления МАБП. Выше отмечалось различие подходов при

восстановлении моделей – можно использовать различную структуру журнала выполнения, получать на выходе модели в различных нотациях, нужны особые структуры представления данных (например, хеширование следов вне зависимости от последовательности задач), статистика особого рода (причино-следственные матрицы, матрицы параллельного выполнения, необходимые, например, для алгоритмов ван-дер-Аальста). Все эти условия были учтены при разработке архитектуры плагнов восстановления МАБП и их использовании системой (рис. 21).

Ключевым элементом системы интеллектуального анализа выполнения бизнес-процессов в системе электронного документооборота является модуль реализации функционала по восстановлению МАБП (рис. 21).

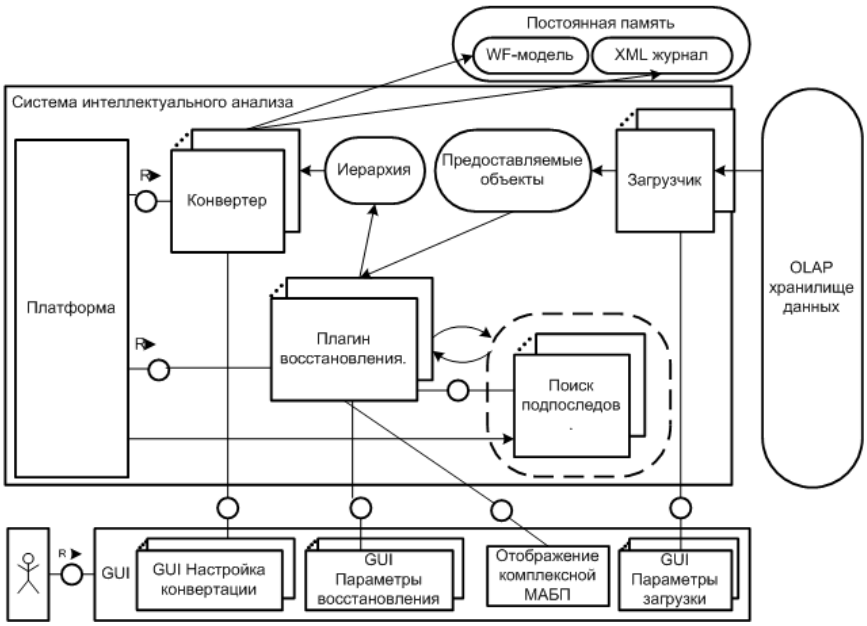

Рис. 21. Плагин восстановления комплексных МАБП в среде системы интеллектуального анализа бизнес-процессов

Пользователь взаимодействуют с системой посредством графического интерфейса, указывая на первом этапе «Параметры загрузки», которые используются модулем «Загрузчик». Загрузка журналов выполнения бизнес-процессов осуществляется из специального «OLAP хранилища данных». В хранилище помимо журналов выполнения бизнес-процессов содержится статистика по этому выполнению (причино-следственная матрица, матрица параллельности), используемая алгоритмом восстановления МАБП. В зависимости от типа системы электронного документооборота и алгоритма восстановления комплексных МАБП могут использоваться разные загрузчики.

«Предоставляемые объекты» – результат работы загрузчика хранится в оперативной памяти, что увеличивает скорость работы алгоритма восстановления.

Система интеллектуального анализа допускает использование различных алгоритмов восстановления через реализацию плагинов, отвечающих заданному интерфейсу. В зависимости от текущего плагина восстановления меняется пользовательский интерфейс «Параметры восстановления», в который выносятся эмпирически задаваемые пользователем коэффициенты.

В случае с восстановлением комплексных МАБП алгоритму необходимо осуществлять поиск свойств, на основе которых производится кластеризация журнала выполнения. Этот функционал реализуется посредством плагина «Поиск подпоследовательностей», загрузку и взаимодействие с которым производит «Плагин восстановления». Для одного реализуемого алгоритма восстановления могут использоваться отличные алгоритмы поиска свойств кластеризации, что указывается в конфигурационном файле плагина восстановления. Загрузку конфигурационного файла производит «Платформа».

Ход выполнения восстановления сопровождается отображением статистических данных в асинхронном режиме. В итоге компонента «Отображение комплексной МАБП» выводит на экран в режиме редактирования результирующую комплексную МАБП. В системе интеллектуального анализа эта модель представлена как «Иерархия» (см. рис. 21).

После восстановления и редактирования комплексной МАБП пользователь может сохранить ее для дальнейшей работы. За этот функционал отвечает подсистема «Конвертер», которая может конвертировать и сохранять модель в различных нотациях (WF-модель, EPC модель, эвристическая сеть, XPDL и т.д.). Выбор способа хранения модели производится пользователем через графический модуль «Настройка конвертации». В постоянной памяти итоговая модель хранится вместе с журналом выполнения. Аналогично работе модуля «Плагин восстановления» модуль «Конвертер» отображает статистику по своему последовательному выполнению.

Работа компоненты валидации МАБП в системе интеллектуального анализа. Плагины классифицируются в соответствии с диаграммой использования: валидации, верификации, восстановления, конвертации, оценки параметров эффективности, сохранения результатов работы в заданном формате.

На рис. 22 представлены наиболее важные аспекты внутреннего взаимодействия плагина валидации бизнес-процессов с системой интеллектуального анализа.

Плагин валидации состоит в свою очередь из части валидации (содержащей алгоритм проверки адекватности разработанной модели) и части предствления диалога с пользователем (GUI плагином). В GUI плагине имеются средства отображения результатов и настройки параметров валидации, специфичные для этого алгоритма.

Работа системы начинается с загрузки всех имеющихся плагинов, в том числе и плагина валидации. После загрузки он становится доступным в пункте главного меню «Анализ».

Первым этапом после загрузки системы становится открытие пользователем из постоянной памяти МАБП и коррелирующего с ним журнала выполнения (МАБП может быть получена и посредством восстановления журнала). Этот функционал выполняется подсистемой «Плагин импорта сетей Петри». Этап запускается пользователем через вызов пункта главного меню «Загрузить МАБП», без которого пункт меню «Анализ» будет недоступен. В качестве «Плагин импорта сетей-Петри» может использоваться любая .NET сборка, реализующая заданный интерфейс, т.е. возможно загружать МАБП, представленную в разных XML форматах, или представленную на физическом носителе либо СУБД.

Второй этап – инициализация внутренних предоставляемых объектов, отображение данных из XML журнала и отображение в пользовательском интерфейсе МАБП, выполняемые подсистемой «Плагин импорта сетей Петри».

Третий этап – запуск алгоритма валидации пользователем (посредством выбора пункта меню «Анализ» и заполнения параметров в форме «Плагин валидации»). В результате работы алгоритма получается объект «Исследуемая *WF*-модель», сопоставляемая посредством алгоритма с исходной МАБП и отображаемая графической подсистемой плагина.

Все предоставляемые объекты доступны в пределах платформы, т.е. возможно использование результатов валидации в других подсистемах.

Алгоритм может быть применен к группе МАБП и их журналов.

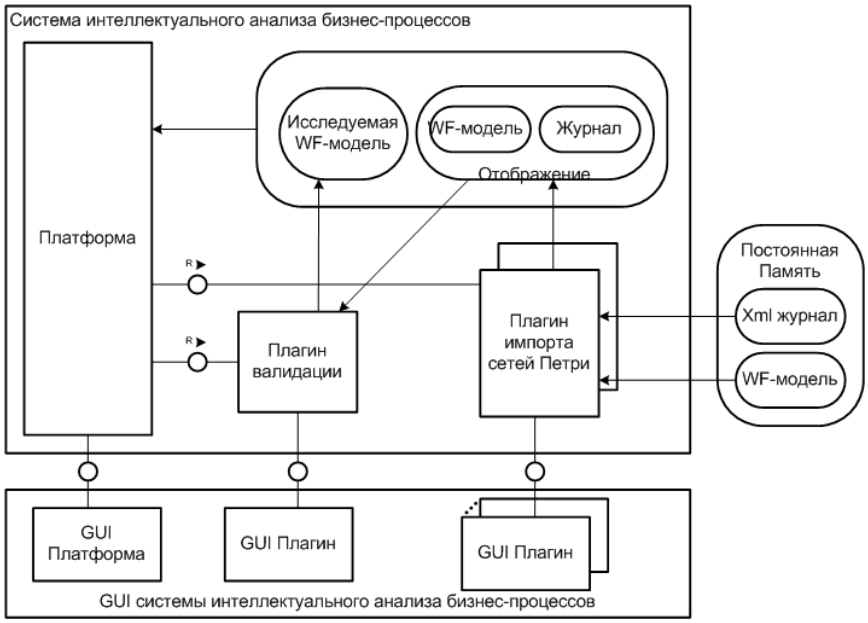

Рис. 22. Плагин валидации в среде системы интеллектуального анализа бизнес-процессов

Обнаружение узких мест при выполнении бизнес-процесса в системе управления автоматизированными бизнес-процессами может быть произведено на основе следующих симптомов, идентифицируемых плагином валидации:

- Число одновременно обрабатываемых экземпляров (слишком) велико. Если одновременно обслуживается много экземпляров, то это может означать наличие проблем. Большое число экземпляров может быть вызвано сильными колебаниями в поступлениях экземпляров или недостаточной гибкостью ресурсов. Однако причиной может быть и то, что процесс содержит слишком много шагов, которые нужно выполнять последовательно.
- Время обслуживания (слишком) велико по сравнению с реальным временем работы. Реальное время работы с экземпляром иногда составляет только малую часть от всего времени обслуживания. Если это так, то, скорее всего, можно найти целый ряд решений для сокращения времени обслуживания.
- Уровень обслуживания слишком низок. Уровень обслуживания в потоках работ показывает, в какой мере организация способна решать обслуживание экземпляров в пределах отведенного для них срока. Если время обслуживания сильно колеблется, то уровень обслуживания является низким. В этом случае нельзя гарантировать конкретное время обслуживания. О низком уровне обслуживания свидетельствует и большое число ситуаций типа «срыв» (здесь под термином «срыв» понимается потеря экземпляра из-за долгого времени ожидания). Если клиент знает, что обслуживание экземпляра (например, заявки на получение ссуды) займет много времени, он обратится в другую компанию. Низкий уровень обслуживания может служить признаком недостаточной гибкости, плохой организации процесса или недостатка структурных возможностей.

Эти три симптома указывают на возможные узкие места. Для их идентификации нужны контрольные значения для измерения симптомов, например известные значения для аналогичных процессов. Обычно не имеет смысла бороться с симптомами путем экстренных мер. Важно выяснить их причины.

Внутренняя организация работы плагина валидации с указанием направления движения информации между компонентами представлена на рис. 23. Семантически плагин состоит из трех главных компонентов, ответственных за получение и валидацию журнала, вычисление метрик и визуализацию результатов. Следовательно, плагин поддерживает некоторые исследуемые структуры данных за пределами его собственной памяти, обращение к которым происходит посредством компоненты «Валидатор». В качестве таких структур данных выступают предоставляемые объекты: журнал бизнес-процесса и его модель. После выполнения алгоритма валидации формируются «Исследуемая WF-модель» и «Исследуемый журнал» –

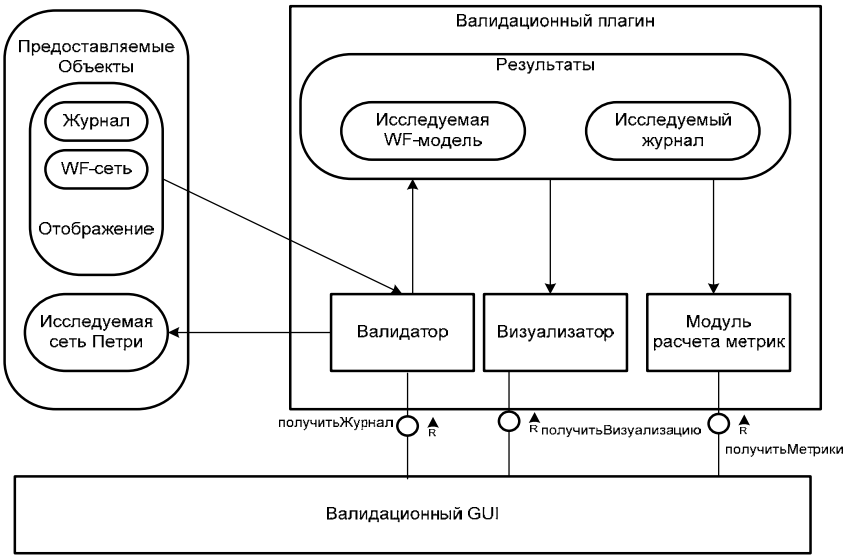

Рис. 23. Архитектура плагина валидации

структуры обработанных данных, предоставляемые для компонент «Визуализатор» (отображение результатов пользователю) и «Модуль расчета метрик».

Для доступа к валидатору пользовательский интерфейс системы интеллектуального анализа вызывает метод *получитьЖурнал*. Результаты работы отображаются посредством вызова методов *получитьВизуализацию* и *получитьМетрики*.

Прикладной программный интерфейс базового функционала системы. Помимо плагинов, система интеллектуального анализа выполнения бизнес-процессов в управлении электронным документооборотом предоставляет базовый функционал, который в дальнейшем используется плагинами. По реализации функционала классы можно разделить на две группы: классы работы с журналом выполнения бизнес-процесса и классы для работы с сетями Петри. Реализуются следующие классы:

- «Состояние».
- «ПространствоСостояния».
- «Диагностируемое состояние» (НачальноеСостояние:Диагности-руемоеСостояние, построитьПокрывающийГраф ()).
- «ДаигностируемоеСостояние».
- «Переход» (Доступен (),СработатьБыстро ()).
- «СетьПетри» (ЗаписатьКТочке).
- «Место».
- «ДиагнозируемоеМесто».
- «ДиагнозируемыйПереход» (Активирован (), СработатьБыстро ()).
- «ДиагнозируемаяСетьПетри» (Записать (), ПолучитьОставшиесяФишки (), ПолучитьПотерянныеФишки (), ПолучитьЧислоВидимыхЗадач()).
- «ВизуализацияСетиПетри» (Записать ()).
- «ВизуализацияЖурнала» (Сохранить ()).
- «РезультатПроверкиСоответствия».
- «ДиагностируемаяКонтрольнаяЗапись».
- «КонтрольнаяЗапись».
- «ЭкземплярПроцесса».
- «ЧтениеЖурнала» (ЕстьСледующая (), Обнулить (), Следующая()).
- «ДиагнозируемыйСледЖурнала».
- «РезультатПроверкиСоответствия».

Опишем наиболее важные связи между классами:

- Класс «СетьПетри» содержит экземпляры классов «Переход» и «Место».
- «ДиагностируемаяКонтрольнаяЗапись» содержит экземпляры классов «ДиагностируемыйПереход» и «ДиагностируемоеМесто».
- «ЧтениеЖурнала» ссылается на «ЭкземплярПроцесса».
- «ДиагностируемыйПереход» ссылается на «ДиагностируемоеПространствоСостояния».
- «ДиагностируемоеПространствоСостояния» ссылается на «ДиагностируемоеСостояния».
- «РезультатПроверкиСоответствия» содержит экземпляры «ДиагнозируемаяСетьПетри».
- «ДиагностируемоеСостояние» является наследником класса «Состояние».
- Класс «ВизуализацияСетиПетри» является наследником класса «ДиагностируемаяСетьПетри».

Класс «СетьПетри» содержит базовый функционал для работы с сетями Петри: список переходов (класс «Переход»), список мест (класс «Место»), дуги их соединяющие. Класс «Переход» содержит два метода: «Доступен()», определяющий доступность данного перехода при текущей разметке и «сработатьБыстро()», позволяющий сработать переходу и изменить текущую разметку.

Набор наследующих их классов: «ДиагнозируемоеМесто», «ДиагнозируемыйПереход», «ДиагнозируемаяСетьПетри» – предназначен для вычисления достижимости разметки, обнаружения блокировок и тупиков, эмуляции работы сети.

Представленного набора классов достаточно для потребностей различных алгоритмов восстановления, реализуемых в качестве плагинов.

Описание архитектуры компоненты восстановления комплексных МАБП. Представим классы (и их методы) плагина восстановления комплексных МАБП [40]:

- «Кластер» (получениеКластеров (), получениеСвойств (целое, целое, дробное, дробное), получитьСвойства (), получитьСледЛ2 (), получитьЭвристическуюСеть (), существуетДугаВСети (целое, целое), существуетПутьВСети (целое, целое), установитьЭвристическуюСеть (), фильтроватьСледы (дробное, спиоск)).
- «Свойство» (получитьЗаголовок (), получитьОстов (), происшествия (ЭвристическаяСеть).
- «ВекторноеПространство» (методКСредних (Целое), получитьТочку (целое)).
- «След» (возможностьКонкатенации (След), конкатенация (След), наложение (След), первыйЭлементВПоследовательности), получениеЗаголовкаДляСвойства (), получениеОстоваДляСвойства (), последнийЭлементВСледе (), распадНаложения (След), Содержит (След)).
- «ВекторнаяТочка» (проекция (Свойство, След), расстояние (ВекторнаяТочка)).
- «CWFИзвлечение» (извлечение (ЧтениеЖурнала), извлечение (WFУстановки)).
- «ЭвристическоеИзвлечение» (извлечение (ЧтениеЖурнала), рассчитатьД1МетрикуЗависимости (Целое), рассчитать-МетрикуЗависимости (Целое, Целое), сделатьБазовыеЗависимости (Дробное, ЧтениеЖурнала)).
- «УправлениеЖурналом» (инициализацияИнформацииПоследующих (Целое), инициализацияЗакрытыхВИнформации (Целое), операцияОдиночки(), получитьДанныеОдиночки (), построитьСвязи (), экземпляр ()).
- «ЧтениеЖурнала» (естьСледующая (), обнулить (), следующая ()).
- «ПлагинИзвлечения» (извлечение (ЧтениеЖурнала)).

Опишем наиболее важные связи между классами:

- Класс «CWFИзвлечение» содержит экземпляры классов «ЭвристическоеИзвлечение», «След», «Кластер», «Свойство».
- «ЭвристическоеИзвлечение» является наследником класса «ПлагинИзвлечения».
- «CWFИзвлечение» является наследником класса «ПлагинИзвлечения».
- «Кластер» содержит экземпляр «ВекторноеПространство».
- «ВекторноеПространство» содержит экземпляр «ВекторнаяТочка».
- «ЧтениеЖурнала» содержит экземпляры «ПлагинИзвлечения» и «УправлениеЖурналом».
- «Свойство» содержит экземпляры «След» и «Кластер».
- «След» реализует интерфейс «ИСравнение».

Основным классом является «CWFИзвлечение», вызов перегруженного метода которого начинает процесс восстановления. Первый метод «извлечение(ЧтениеЖурнала):СетьПетри» является унаследованным от базового класса [41] для всех плагинов восстановления. Система интеллектуального анализа, используя полиморфизм, создает экземпляр класса «CWFИзвлечение», оперируя объектом с типом «ПлагинИзвлечения».

Класс «ЭвристическоеИзвлечение» предназначен для получения эвристической сети (сети без асинхронности, в которой ветвления игнорируются в пользу наиболее частых дуг), он также унаследован от «ПлагинИзвлечения» (т.е. реализует простейший алгоритм восстановления МАБП). Метод «рассчитатьД1МетрикуЗависимости(Целое):Дробное» «ЧтениеЖурнала» предназначен для получения информации из журнала выполнения бизнес-процесса. Информация о журнале получается последовательно для каждой записи, метод «естьСледующая» проверяет, является ли текущая запись конечной в журнале, метод «Обнулить» сбрасывает маркер позиции на начальную позицию в журнале, «Следующая» возвращает следующую запись в журнале и переводит маркер позиции журанала на следующую позицию. В своей работе этот класс базируется на функционале класса «УправлениеЖурналом», который реализован в виде паттерна объектно-ориентированного программирования «одиночка» [42]. Посредством этого класса осуществляется доступ к файлу журнала непосредственно, либо к хранилищу данных в СУБД.

Алгоритму восстановления комплексных МАБП необходимо производить разбиение заданного журналом выполнения множества следов на кластеры, что осуществляется набором классов: «Векторная точка», «Векторное пространство», «Кластер», «Свойство», «След».

Класс «Кластер» предназначен для работы с кластером следов: инициализации кластера, фильтрации следов, поиска свойств. Метод «установитьЭвристическуюСеть» вызывается первым при работе алгоритма восстановления комплексных МАБП – он устанавливает объект эвристической сети, используемый в дальнейшей работе алгоритма. Следующим вызывается метод «получениеКластеров», который производит разбиение исходного множества следов на заданное количество кластеров. Для этих целей используется вызов метода «получениеСвойств», вычисляющий свойства, выступающие в качестве критерия кластеризации. В работе этого метода используются вспомогательные: «фильтроватьСледы», уменьшающий количество исследуемых следов; «существуетДугаВСети», проверяющий на основе следа и эвристической сети частоту дуги во всем журнале выполнения; «существуетПутьВСети», проверяющий на основе следа и эвристической сети частоту следа во всем журнале выполнения; «получитьСледЛ2» для исключения повторяющихся переходов.

Класс «Кластер» предоставляет данные в формате, доступном для применения в алгоритме *k*-средних, реализация которого содержится в «ВекторноеПространство», метод «методКСредних».

«След» является классом представления следа журнала выполнения и содержит часть логики алгоритма восстановления комплексных МАБП. «ВозможностьКонкатенации» проверяет, возможно ли произвести слияние двух *w*-шаблонов, в случае возможности можно произвести конкатенацию вызовом «конкатенация». В реализации алгоритма удобно производить разделение следа на заголовок (один элемент) и остальную часть. Для работы со следом в таком формате есть методы: «первыйЭлементВПоследовательности», «получениеЗаголовкаДляСвойства», «полученииеОстоваДля Свойства», «последнийЭлементВСледе».

Организация хранения журналов выполнения бизнес-процессов. Для хранения журналов в базе данных необходимо разработать ее структуру, учитывающую особенности работы алгоритмов восстановления МАБП (построение причинно-следственных матриц) [43]. В связи с тем что записей в журнале, как правило, очень много и они непрерывно добавляются, необходимо обеспечить высокую скорость добавления новых записей (что влияет на скорость работы системы электронного документооборота) и высокую скорость получения информации о задачах, пользователях, экземплярах бизнес-процессов, движении документов, подразделениях (что влияет на производительность системы интеллектуального анализа). Анализ МАБП происходит относительно редко, что является также существенным фактором в разработке архитектуры хранилища данных [44].

Системы управления автоматизированными процессами располагают средствами поддержки выполнения бизнес-процессов, предоставляющими предписанные заранее модели, в которых журналы содержат информацию по экземплярам задач, выполняемым в каждом экземпляре процесса.

В целях улучшения архитектурных характеристик хранилища под МАБП понимается множество взаимосвязанных между собой задач, участников и зависимостей между ними. Задачи связаны с индивидуальными шагами в бизнес-процессе, участники (информационные системы или люди) отвечают за выполнение задач, и, наконец, зависимости определяют, в какой последовательности будут выполняться задачи.

Описание хранилища данных можно разделить на две части: описание прикладного программного представления хранилища (классов и зависимостей между ними) и структуры базы данных.

Диаграмма классов для работы с журналами выполнения МАБП.

Динамические и статические аспекты хранения журналов могут быть представлены посредством следующих сущностей:

- «Подразделение».
- «Пользователь».
- «Роль».
- «Участник».
- «Экземпляр составной задачи».
- «Экземпляр элементарной задачи».
- «Экземпляр внешней задачи».
- «Экземпляр задачи».
- «Экземпляр МАБП».
- «МАБП».
- «Задача».
- «Составная задача».
- «Внешняя МАБП».
- «Элементарная задача».
- «WFМодель».
- «Элемент модели».
- «Переход модели».
- «СпецПереход».
- «Инцидент задачи модели».
- «Инцидент контроля модели».

Опишем наиболее важные связи между сущностями:

- «Подразделение» ссылается на родительское и дочерние подразделения, представленные объектом типа «Подразделение».
- «Участник» является наследником сущностей «Роль» и «Пользователь».
- «Экземпляр составной задачи» содержит экземпляр сущности «Экземпляр МАБП».
- «Экземпляр задачи» ссылается на предыдущую и последующую задачи в МАБП типа «Экземпляр задачи», использует «Экземпляр МАБП», содержит ссылку на дочерний объект типа «Экземпляр составной задачи».
- «Экземпляр МАБП» является экземпляром «МАБП», содержит экземпляр «Участник».
- «МАБП» ссылается на автора МАБП типа «Участник», использует объект типа «Задача», является наследником сущности «Задача», ссылается на сущность «СпецИнцидент».
- «Задача» является наследником «Внешняя МАБП», ссылается на объект типа «Составная задача».
- «Составная задача» ссылается на объект типа «СпецИнцидент».

МАБП, которую мы здесь предполагаем, при проектировании хранилища может быть представлена с использованием различных техник: структурированных или неструктурированных МАБП, в стиле текстовых языковых конструкций или в стиле грифовых интерпретаций.

Диаграмма классов состоит из нескольких уровней. Уровень спецификации содержит описание типов МАБП и типов задач совместно с их композицией посредством происшествий (рис. 26). Уровень модели содержит расширенные спецификации МАБП, например инциденты задач могут иметь расширенные характеристики (время завершения, агенты и т.п.). Уро-

вень экземпляров содержит информацию о характеристиках выполнения конкретных экземпляров задач и МАБП. И наконец, организационный уровень содержит информацию об участниках, их ролях, структуре всего предприятия.

МАБП состоит из задач, которые в свою очередь могут быть МАБП, внешними МАБП (моделями бизнес-процессов, находящихся в другом домене), элементарными задачами, составными задачами. Составные задачи состоят из других задач, представленных как инциденты в составной задаче. Модель допускает наличие иерархической зависимости между задачами, в которой указываются родительская и дочерние задачи. Внутри составной задачи частная задача может иметь несколько инцидентов, что однозначно идентифицируется сущностью инцидент. Инцидент ассоци-ирован в точности с одной задачей и представляет место, где задача была использована в спецификации составной задачи. Каждый инцидент, между тем, имеет разных предшественников и последователей, что выражается ассоциативным классом «СпецПереход».

Есть две причины, по которым произведено выделение сущностей «Задача» и «Инцидент задачи». Во-первых, это возможность повторного использования задачи в разных МАБП. Во-вторых, это упрощение поддержки, что подразумевает изменение лишь одного подпроцесса, влекущего за собой автоматическое изменение МАБП, в которых он используется. Это подразумевает, что новые МАБП могут быть получены путем композиции из существующих задач. Такая композиция называется спецификацией процесса. Идентификация множественного вхождения очень важна с точки зрения обработки хранилища журналов выполнения, так как это позволяет агрегировать данные о выполнении в различных позициях МАБП и даже в различных МАБП.

Когда задача используется несколько раз в рамках одной МАБП, мы также различаем инциденты, обозначая итоговые элементы с использованием сущности «Элемент модели». Аналогично уровень спецификаций МАБП содержит информацию об элементах модели, иерархиях и отношениях переходов в этих элементах модели.

На уровне экземпляров классы «Экземпляр МАБП», «Экземпляр задачи» используются для представления экземпляров МАБП и их задач во время выполнения. Аналогично МАБП экземпляр МАБП состоит из экземпляров задач, для которых описаны предшествующие и последующие задачи. Кроме того, экземпляр МАБП всегда соответствует в точности одной МАБП.

Participant (участник) ответственен за выполнение бизнес-процессов. В этой метамодели участник может быть смоделирован с помощью пользователей и ролей. В идеальном случае должна использоваться концепция роли, которая дает более обобщенное описание пользователей. Однако в метамодели непосредственное описание единственного пользователя также возможно. Обобщенно предполагается, что пользователи, роли и пользователи в некоторых ролях могут участвовать в различных бизнес-процессах различных структурных подразделений организации.

Разработка метамодели хранения данных определяется набором информации, который может представить система управления автоматизированными бизнес-процессами, и потребностями системы интеллектуального анализа выполнения бизнес-процессов. Тем не менее, каждое хранилище данных должно быть адаптировано под специфические черты конкретной корпоративной информационной системы (КИС). Здесь показывается прототипное хранилище данных, в каждом конкретном случае возможно внесение незначительных изменений для хранения большей или меньшей информации по выполнению процессов.

Рассмотрим более подробно взаимодействие хранилища выполнения бизнес-процессов и хранилища статистики их выполнения. Основным компонентом бизнес-системы является миссия предприятия, определяющая назначение компании, ее значимость для общества и направление деятельности. С точки зрения организации бизнеса, изменение миссии эквивалентно построению новой компании. Практически столь же устойчивой является иерархия целей предприятия и стратегия его развития. Изменения в развитии являются, как правило, следствием таких внешних событий, как технологическая революция, радикальное изменение условий рынка или законодательства. Критические факторы успеха и состав тактических задач по реализации сформированных планов периодически пересматриваются на основе анализа тенденций развития рынка и статистики результатов деятельности. Определение результатов деятельности и подсчет значений показателей эффективности проводится столь часто, сколь это возможно для конкретного сектора рынка, групп изделий, услуг, клиентов, поставщиков и т.д. Однако и в этом случае достоверные данные являются результатом накопленной статистики, хотя и за менее продолжительный период. Все последующие компоненты входят в состав корпоративной информационной системы, задачей которой является информационная поддержка выполнения бизнес-процессов и принятие управленческих решений. При этом динамику развития этой системы в наибольшей степени определяют собственно бизнес-процессы. С точки зрения организации бизнес-системы основной задачей технологии управления автоматизированными бизнес-процессами является отделение правил выполнения бизнес-процессов от прикладных систем и систем управления базами данных, что обеспечивает принципиально большую гибкость и адаптируемость информационной системы. Иными словами, технология управления автоматизированными бизнес-процессами предоставляет возможность оперативной модификации правил выполнения бизнес-процессов без перестройки прикладного программного обеспечения или изменения структуры корпоративной базы данных.

Другим важным направлением использования технологии управления автоматизированными бизнес-процессами служит интеграция различных приложений и данных вокруг бизнес-процесса. В этом отношении системы управления автоматизированными бизнес-процессами можно рассматривать как определенный шаг в развитии архитектуры открытых систем. Стандарты, разработанные WfMC, подтверждают эффективность и результативность усилий, нацеленных на развитие этого направления.

Хранилище журналов не должно служить ограничением при изменении структуры бизнес-системы, правильным решением здесь было бы собирать данные посредством системы управления автоматизированным бизнес-процессами. При этом различные компоненты бизнес-системы можно свободно менять, не опасаясь за последствия.

Интеграция СЭД с системой интеллектуального анализа выполнения бизнес-процессов. Хранение журналов выполнения бизнес-процессов организовано с использованием технологии OLAP для первоначальной агрегации, анализа и интерпретации информации. OLAP хранилища отличаются от традиционных реляционных баз данных рядом аспектов: они разработаны для получения ответов на сложные вычислительные запросы в ущерб возможности сохранять большие массивы

информации; также они отличаются, как правило, большими объемами хранимых данных; содержат данные за определенный период (снимок данных).

Наиболее часто используемая архитектура для хранилищ данных – многомерные кубы данных, где транзакционные данные (ячейки, факты или измерения) описаны посредством иерархий измерений. В нашем случае транзакционными являются данные о выполнении задач бизнес-процесса в системе электронного документооборота, агрегированные в измерения.

OLAP операции (детализация, обобщение, срезы, проецирование) позволяют аналитику быстро получать отчеты на различном уровне абстракции и рассматривать данные с различных перспектив [45].

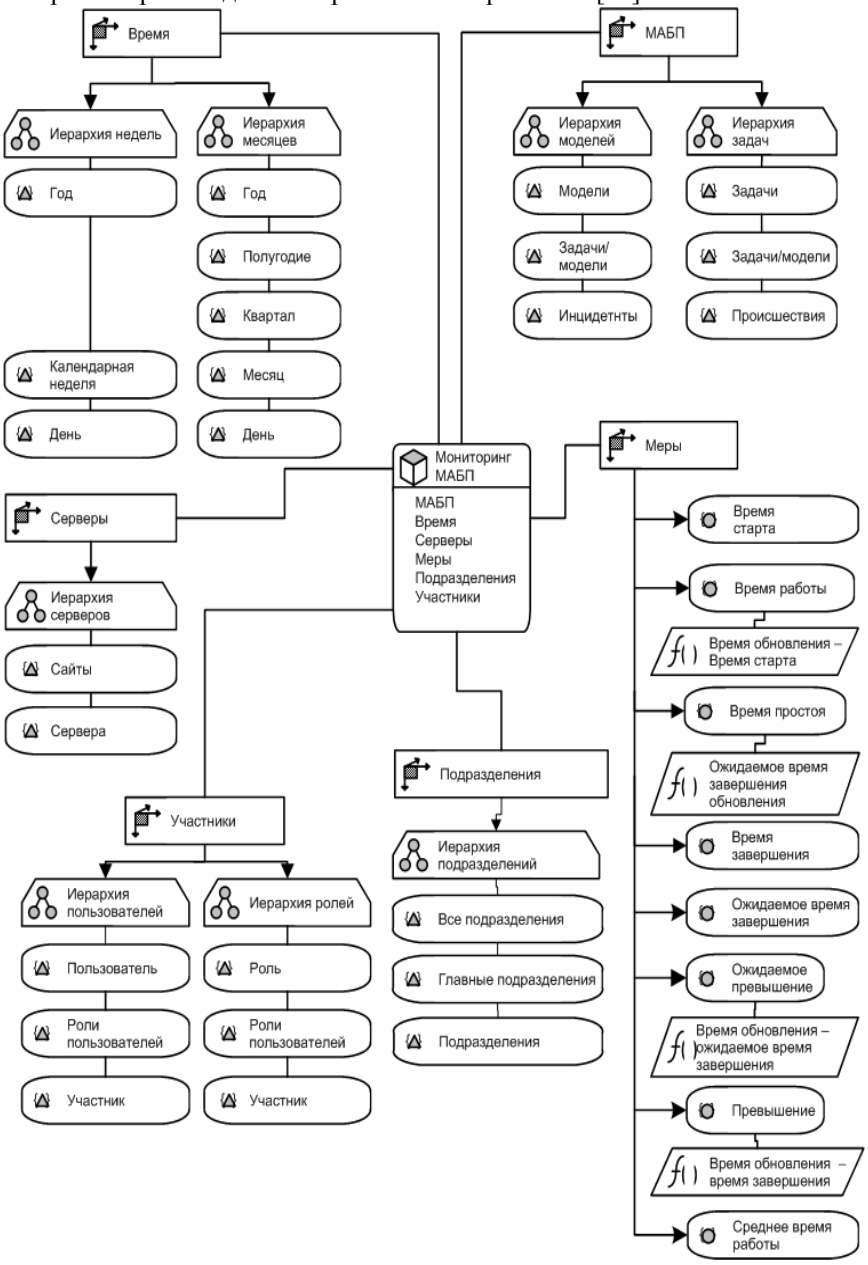

Рис. 24. OLAP хранилище данных для журналов выполнения бизнес-процессов

В данном случае МАБП – это коллекция задач, участников и зависимостей между ними. Задачи соответствуют индивидуальным шагам в бизнес-процессе, участники (пользователи или программные подсистемы) отвечают за выполнение этих задач, зависимости определяют последовательность выполнения задач и потоки данных между ними.

МАБП состоит из задач, которые могут быть в свою очередь либо МАБП, либо внешней МАБП, либо элементарной задачей, либо сложной задачей. Сложные задачи состоят из других задач, представленных как композиция.

Структура хранилища данных определяется наличием информации в СЭД и потребностями аналитиков и используемых алгоритмов в системе интеллектуального анализа [46, 47].

Помимо алгоритмов анализа системы, хранилищем могут пользоваться и системные администраторы, пользователи и администраторы WFMS, аналитики. Список возможных запросов:

- Как часто запускается конкретная МАБП? (Информация, полученная в результате этого запроса, может быть использована для идентификации ключевых процессов, которые подлежат пересмотру и оптимизации в первую очередь.)
- Какие задачи из этих МАБП наиболее часто используются?
- Выполнение каких МАБП регулярно не укладывается в сроки?
- Выполнение каких задач приводит к задержке по срокам?
- Каково количество просрочек?
- Как часто МАБП, вызывающие просрочки, инициируются к запуску?

- Как много различных пользователей участвуют в выполнении наиболее частых МАБП?
- Участие каких пользователей регулярно приводит к просрочкам?
- Какие из пользователей регулярно перегружены, а какие периодически имеют свободные ресурсы?
- Какие пользователи являются участниками данной МАБП?
- Сравнение характеристик различных версий МАБП.
- Анализ выполнения задачи в разных МАБП.
- Анализ выполнения задачи различными пользователями.
- Анализ производительности работы пользователей в разные периоды времени.

Другим важным параметром анализа является время. Обнаружение пиковой загрузки может помочь решить проблемы оптимизации выполнения процесса (распределение нагрузки на серверы, распределение заданий между пользователями).

Модель, представленная на рис. 24, состоит из шести измерений:

- МАБП;
- Участники;
- Подразделения;
- Серверы;
- Время;
- Измерения.

Инциденты, необходимые для повторного использования, являются уровнем гранулярности [48] в измерении МАБП. Возможно несколько альтернативных путей для построения структуры измерения МАБП. Один из возможных – разделение задач и МАБП в два разных измерения с целью устранить проблему участия одной задачи в нескольких МАБП. Недостаток подобного подхода заключается в отсутствии возможности развернуть инциденты, ассоциированные с данной МАБП. Другой путь заключается в объединении инцидентов в значимые задачи и агрегировании этих задач соответствующими МАБП. В этом решении также обнаруживается недостаток: каждая задача может принадлежать множеству МАБП и их значения должны быть пропорциональны для каждой МАБП.

Решение, предложенное в разработанном хранилище журналов, не содержит описанных выше недостатков. На первом этапе происходит объединение инцидентов в элементы, представляющие собой комбинации МАБП и задач. Далее эти элементы могут быть объединены в МАБП или задачи. В хранилище данных участие одной задачи в различных МАБП предоставляет возможность проанализировать параметры производительности этой задачи применительно к разным контекстам выполнения.

Для представления пользователей и подразделений также существует несколько альтернатив. Во-первых, можно произвести разделение пользователей и их ролей по двум разным измерениям, где подразделения являются объединением пользователей. Смыслом подобного решения является потенциальная возможность каждого пользователя выступать в его департаменте в любой роли. На практике такого решения недостаточно, так как часто один пользователь может работать в нескольких подразделениях. В предложенном хранилище данных произведено разделение участников и подразделений на два различных измерения. Нижний уровень измерения участников представлен элементами – комбинацией пользователей и ролей. На следующем шаге эти элементы объединяются либо в пользователей, либо роли.

Причиной для разделения пользователей и подразделений на разные измерения послужила возможность пользователей участвовать в различных МАБП от различных подразделений.

Введение измерения серверов имеет технические потребности и предоставляет возможность получить информацию о загрузке различных серверов, что может быть использовано для распределения нагрузки. Структура измерения серверов очень простая и самоописываемая.

В измерении времени были выбраны календарные дни, которые объединяются в месяцы, кварталы и годы. Недели представлены в отдельной иерархии, так как в разных годах неделя может состоять в различных месяцах.

Для измерения мер было выбрано следующее множество:

- время старта;
- время работы (время обновления – время старта);
- время простоя (ожидаемое время завершения – время обновления);
- время завершения;
- ожидаемое время завершения;
- ожидаемое превышение (время обновления – ожидаемое время завершения);
- превышение (время обновления – время завершения);
- среднее время работы.

Обзор отечественных систем электронного документооборота. Российский рынок СЭД растет, и количество представленных систем постоянно увеличивается. Ниже приводится описание наиболее популярных систем с анализом возможности применения к ним методов интеллектуального анализа.

«*Docs Fusion*» и «*Docs Open*». Это одна из самых популярных в мире систем, относящихся к классу «электронных архивов». Различные поколения и компоненты продукта получили различные названия, и поэтому при ознакомлении с ним возникает путаница. Изначально существовала система «Docs Open» – клиент-серверное приложение с «толстым» клиентом. Далее был разработан сервер приложений «Docs Fusion», позволивший избавиться от необходимости иметь «толстого» клиента, обращающегося напрямую к базе данных. К нему есть два клиента: Windows-клиент «PowerDocs» и Web-клиент «CyberDocs». Перспективной для компании является платформа «Docs Fusion». Для простоты мы далее будем называть систему словом Docs, имея в виду «Docs Fusion», и клиенты – «PowerDocs», «CyberDocs».

В России система «Docs Open» представлена достаточно давно и уже применяется во многих организациях. «Docs Open» может эффективно применяться и в крупных организациях с большим числом сотрудников (тысячи человек), и в небольших фирмах, где работают пять-шесть человек. Система в первую очередь позиционируется для организаций, которые за-

нимаются интенсивным созданием документов и их редактированием (головные офисы компаний, консалтинговые компании, органы власти и т.д.).

Клиент «PowerDocs» – это «Windows»-интерфейс, по идеологии построения напоминающий «MS Outlook». Пользователь может обращаться к Docs через интерфейс самого «MS Outlook» и даже в окне «Windows Explorer», что позволяет работать с папками Docs как с обычной файловой системой. Клиент «PowerDocs» позволяет осуществить мобильный доступ с возможностью синхронизации при подключении, в том числе и по медленным линиям. Эта функция также позволяет обеспечить стабильную работу пользователя в режиме неустойчивой работы локальной сети. Клиент «CyberDocs» обеспечивает практически ту же функциональность, что и «PowerDocs», но через Internet-браузер.

В одном комплексе может быть установлено несколько серверов «DocsFusion», при этом автоматически реализуется балансировка нагрузки и устойчивость к сбоям. Это значит, что при сбое одного из серверов пользователи почувствуют лишь некоторое замедление работы системы, а сама система действительно может обеспечить одновременную работу с ней достаточно большого количества пользователей. Для хранения данных системы необходимо использовать «Microsoft SQL Server» или «Oracle». В качестве хранилища для документов используется файловая система. Поддерживается механизм иерархического хранения данных HSM.

Система позволяет осуществить интеграцию и стыковку с другими прикладными системами как на уровне клиента «PowerDocs», так и на уровне сервера. Docs – это открытая платформа, к ней поставляются средства разработки для создания специализированных приложений или интеграции с другими системами.

Продукт не ориентирован на применение в области инженерно-конструкторского документооборота, в нем нет интеграции с системами CAD/CAM. В территориально распределенных организациях могут возникнуть проблемы, так как в системе нет механизмов репликации информации. В СЭД имеются средства поддержки совместной работы на уровне рабочей группы, однако для больших организаций этих средств недостаточно.

«*Documentum*». «Documentum» — это система управления документами, знаниями и бизнес-процессами для крупных предприятий и организаций. Система только начинает внедряться в России, но уже давно и прочно заслужила позицию одного из лидеров индустрии. «Documen- tum» – это платформа, в большей степени, чем готовый продукт, предназначенная для создания распределенных архивов, поддержки стандартов качества, управления проектами в распределенных проектных группах, организации корпоративного делопроизводства, динамического управления содержимым корпоративных интранет-порталов.

В продукте предусмотрено все, что нужно крупной организации, – это интегрированная система, позволяющая комплексно решать достаточно широкий спектр задач. Она включает необходимую функциональность для автоматизации деловых процессов: маршрутизацию, утверждение, распределение, уведомление и контроль исполнения. «Documentum» достаточно масштабируема, вся информация, которая хранится в системе, управляется выделенным серверным компонентом – хранилищем «DocBase». «Documentum» содержит механизмы, позволяющие управлять хранением информации: она поддерживает управление версиями, публикацией, доступом, местонахождением информации и дает возможность осуществлять архивацию. Система может эффективно работать в распределенной архитектуре в территориально разобщенных подразделениях благодаря реализованным механизмам репликации и синхронизации информации, а также централизованного администрирования.

«Documentum» поставляется в нескольких редакциях, ориентированных на различные задачи: создание порталов, управление знаниями, обеспечение соответствия стандартам/управление качеством, организация B2B-взаимодействия. Важной особенностью для многих отраслей является возможность полного документирования всех событий и жесткого отслеживания выполнения определенных процедур.

Продукт включает в себя средства, позволяющие создавать приложения в среде «Documentum», в том числе Web-приложения. Но для разработки приложений для «Documentum» и интеграции его с другими приложениями можно использовать и внешние средства разработки: продукт построен на современных открытых технологиях. Благодаря такой открытости для его внедрения в существующую информационную среду не потребуется существенных расходов на модификацию инфраструктуры. «Documentum» отличается мощной поддержкой форматов и средствами автоматической генерации файлов форматов PDF и HTML из любых хранимых данных. Одним из преимуществ использования этого продукта для промышленных предприятий является возможность его интеграции с ERP и CAD/CAM-системами.

«Documentum» имеет относительно высокую стоимость внедрения за счет того, что является «конструктором», из которого собирается необходимая функциональность, и далек от «коробки», а кроме того, сложен в освоении, что является очевидной оборотной стороной его функциональной полноты. Поэтому оснащение этим продуктом отдельных рабочих групп или организаций с числом сотрудников порядка одного-двух десятков имеет мало смысла, разве что в случае, если предполагаются быстрые темпы роста.

Безусловно, «Documentum» является одним из наиболее мощных продуктов, однако позволить себе такую систему могут только организации, которые очень серьезно относятся к задаче автоматизации документооборота и готовы выделить на нее достаточные финансовые и интеллектуальные ресурсы.

Система «*LanDocs*». Система «LanDocs» в первую очередь ориентирована на делопроизводство и архивное хранение документов. Она состоит из нескольких компонентов: системы делопроизводства, сервера документов (архива), подсистемы сканирования и визуализации изображений, подсистемы организации удаленного доступа с использованием Internet-клиента, почтового сервера.

Компонент делопроизводства реализован в клиент-серверной архитектуре на базе промышленной СУБД: «Oracle» или «Microsoft SQL Server». Программное обеспечение для централизованного управления хранением документов в электронном архиве реализовано в виде отдельного сервера. В качестве отдельной опции поставляется модуль полнотекстового поиска документов с учетом правил русского языка. Почтовая служба «LanDocs» сделана так, что сотрудники, у которых установлен специальный клиентский компонент «LanDocs», могут получать сообщения-задания и отчитываться по ним, используя стандартный почтовый ящик «Microsoft Exchange» или «Lotus Notes».

Продукт открыт для разработчиков – имеется API для встраивания «LanDocs» в Windows-приложения сторонних разработчиков. Компонент сканирования и работы с изображениями имеет достаточно продвинутую функциональность: он позволяет фильтровать изображения, исправлять перекос, возникший после сканирования, распознавать текст в случае необходимости.

Система «LanDocs» не ориентирована на поддержку коллективной работы и процесса создания документов.

«*Microsoft SharePoint Portal Server*». Система является электронным архивом с развитыми средствами поддержки совместной работы. Поддерживает: совместное создание документов, ведение версий документов, изъятие и возврат документов в архив. В нем нет Windows-клиента как такового. Для доступа к архиву используется Web-клиент (сторонние разработчики могут дописывать для него свои компоненты) и компонент, интегрированный в «Windows Explorer», что позволяет обращаться к архиву как к набору файлов.

В систему встроены достаточно мощные средства индексации и поиска. Причем поиск может осуществляться как по внутренним хранилищам информации (файлы, интранет-сайты, базы «Microsoft Exchange», базы «Lotus Notes»), так и по внешним (интернет). Система способна индексировать и публиковать документы, которые находятся в файловой системе на серверах локальной сети. В качестве альтернативы документы можно переместить в хранилище самого сервера (которое аналогично хранилищу «MS Exchange 2000»). Регистрационные данные о документах всегда помещаются в хранилище сервера, при этом нет необходимости в использовании отдельного сервера баз данных.

Система достаточно открыта, к ней можно добавлять различные компоненты. Опора на Web-технологии делает такое расширение технологичным.

Продукт наиболее эффективен в качестве базы информационной инфраструктуры для компаний, которые делают ставку не на иерархическое управление, а на матричную организацию взаимодействия людей и плоскую структуру управления. Для традиционных фирм она может стать звеном в интранет-инфраструктуре для «оживления» последней, так как концепции, заложенные в эту систему, позволяют сделать процесс публикации информации на портале частью каждодневной работы с документами, не требующей особо сложных процедур, ресурсов и организационных усилий.

Система «*Oprima Workflow*», кроме общего механизма организации потока работ, позволяет хранить на время проведения работ все документы, относящиеся к процессу. Для этого в качестве хранилища используется механизм общих папок «Microsoft Exchange». Полезной возможностью является отслеживание критических путей и представление комплекса взаимосвязанных работ в виде диаграмм Ганта. Впрочем, эту работу можно производить и в среде «MS Project» с использованием всех ее возможностей, так как «Optima Workflow» позволяет экспортировать данные о ходе работ в эту программу.

Система автоматизирует процессы регистрации документов по правилам делопроизводства, реализует механизмы аннотирования и сбора резолюций, доставки отчетов об исполнении поручений.

Тот факт, что «Optima Workflow» использует в качестве основного хранилища и транспорта «Microsoft Exchange», определяет все ее возможности по надежности хранения, защите от сбоев, возможности применения медленных линий связи, синхронизации данных, ограничения доступа к данным. Для регистрации версий документов используется СУБД, к которой осуществляется доступ через ODBC.

Как уже указывалось выше при классификации систем, workflow-система удобна для формализации типовых процедур работы с документами в организациях, где такая работа является ежедневной практикой. Так как «Optima Workflow» в качестве сервера использует «Microsoft Exchange», его легко внедрить в тех компаниях, где он уже применяется по своему прямому предназначению – как почтовый сервер. Не нужно рассчитывать на то, что «Optima Workflow» позволит вам задействовать «Microsoft Exchange» в качестве электронного архива – для этого есть другие продукты, к примеру описанный выше «Microsoft SharePoint Portal Server». «Optima Workflow» хранит документы только в процессе, пока работы, связанные с ним, не завершены.

Система «*БОСС-Референт*» относится к категории систем, ориентированных на поддержку управления организацией, эффективной работы сотрудников и на накопление знаний, и при этом имеет развитые дополнительные сервисы.

Основное применение – создание корпоративной системы, охватывающей деятельность сотрудников на своих рабочих местах и поддерживающей управленческие бизнес-процессы. Поддерживает делопроизводство, организационное управление, согласование документов. Отличительная особенность ее в том, что, будучи полноценной системой документооборота, она уже обладает всей необходимой функциональностью для реализации делопроизводства. В ней с самого начала фигурируют понятия, роли и функции, присущие организациям со сложной иерархической структурой в России. Другая отличительная черта системы «БОСС-Референт»: в ней реализованы функции CRM-системы, контроля договоров, учета материальных ценностей, потокового сканирования и распознавания (в «БОСС-Референт» интегрирована система «FineReader»), электронной конференции и доски объявлений.

Система реализована на платформе «Lotus Notes». Благодаря этому вдобавок к функциям «БОСС-Референт» пользователи получают в свое распоряжение все богатство функциональности самой среды «Lotus Notes», включая электронную почту, репликацию данных, возможность удаленной работы и т.д. «БОСС-Референт» является наиболее открытой во всех смыслах системой – она поставляется вместе с полными исходными текстами. К ней дополнительно прилагается инструментарий разработчика с полным описанием функций прикладного программного интерфейса.

На «БОСС-Референт» стоит обратить внимание в первую очередь тем организациям, которые уже используют «Lotus Notes». Остальные должны отдавать себе отчет в том, что, выбирая эту систему, они получают и «Lotus Notes» в качестве основы своей информационной инфраструктуры и должны приобрести лицензию «Lotus Notes» на каждое рабочее место.

Система «*Дело*», которая до недавнего времени называлась «Дело-96», является типичным представителем систем автоматизации делопроизводства и именно в этом качестве приобрела популярность у нас в стране. Продукт поддерживает идеологию делопроизводства, суть которой в следующем: чтобы было совершено любое действие в организации, нужен документ, для которого обеспечивается движение. Движение документов (при том, что физически они, естественно, не перемещаются) происходит за счет изменения учетных записей о документах в базе данных.

Для хранения документов компания ЭОС недавно представила отдельный продукт, интегрированный с системой «Дело», обеспечивающий функции электронного архива. В системе реализован web-интерфейс, что удобно для организации удаленного доступа и построения интранет-порталов. Система имеет API, позволяющий интегрировать ее с различными приложениями. «Дело» хранит учетные записи средствами промышленной СУБД – «Oracle» или «Microsoft SQL Server»; осуществляет полное протоколирование действий пользователей с документами. Последняя версия интегрирована с системой распознавания «FineReader» для занесения в нее данных с бумажных документов.

Продукт в первую очередь интересен для организаций, которые сталкиваются с необходимостью внедрения формализованного делопроизводства.

«*Евфрат*» является простым электронным архивом с базовыми возможностями контроля исполнения.

«Евфрат» построен на парадигме «рабочего стола» с папками. Документы раскладываются по папкам, которые могут иметь любую степень вложенности. Собственного хранилища файлов «Евфрат» не имеет – система хранит только ссылки на файлы или на страницы в интернет. Для хранения реквизитов документов используется СУБД собственной разработки. В комплект продукта входят утилиты, позволяющие уплотнять и архивировать базу данных этой СУБД.

Отличительной особенностью является возможность открыть и просмотреть любой документ поддерживаемого системой формата с помощью встроенной программы просмотра, правда, без форматирования и иллюстраций, что, впрочем, не составляет проблемы, так как документ можно открыть во внешнем «родном» приложении. К сожалению, «Евфрат» не дает возможности отслеживать получение и возврат документов и хранение версий, что может усложнить коллективную работу с документами. Система позволяет описать категории документов и приписать любой из категорий любые реквизиты.

Для ввода информации с бумажных носителей в комплект продукта входит система потокового ввода, основанная на другом продукте компании – системе распознавания текстов «Cuneiform». По сути, «Евфрат» представляет собой средство сканирования, распознавания, регистрации документов, присвоения им реквизитов, индексации, полнотекстового поиска, назначения заданий, связанных с документом, и контроля их исполнения. Это недорогое решение, которое может оказаться полезным в малом офисе или на предприятиях, не предъявляющих высоких требований к масштабируемости информационной системы.

ЗАКЛЮЧЕНИЕ

Трудности, с которыми приходится сталкиваться при разработке сложных МАБП, стимулируют в последнее время разработку технологий восстановления моделей автоматизированных бизнес-процессов и интеллектуального анализа их выполнения в целом. Цель данной работы заключалась в автоматическом получении модели для процесса на основе аккумулированных в течение предыдущего выполнения данных. Существующие техники позволяют получить МАБП, хорошо подходящие для применения в WFMS, ввиду высокой актуальности (как следствия разработки на реальных данных выполнения МАБП).

Несмотря на большой рост интереса к методам и алгоритмам интеллектуального анализа выполнения МАБП, все еще не существует системы, позволяющей их использовать при реальной работе аналитика. Авторами была разработана архитектура системы интеллектуального анализа, позволяющая реализовать существующие методы.

В монографии сформированы и получены следующие основные выводы и результаты:

1. Проведена классификация методов ИА и рассмотрены наиболее востребованные отрасли их практического применения.
2. Сделана математическая постановка задачи ИА выполнения бизнес-процессов, а именно:
   - задача получения комплексных *WF*-моделей;
   - задача поиска частых подпоследовательностей при данном журнале выполнения и МАБП.
3. Разработан алгоритм получения комплексных *WF*-моделей на основе журнала выполнения.
4. Разработаны два алгоритма поиска частых подпоследовательностей, а также произведен анализ их производительности на синтезированных данных.
5. Разработана система интеллектуального анализа выполнения бизнес-процессов в управлении электронным документооборотом, а именно:
   - архитектура взаимодействия компонент внутри системы с учетом особенностей рассмотренных методов и алгоритмов интеллектуального анализа;
   - прикладные программные интерфейсы, предоставляемые системой с учетом обобщенных требований рассмотренных методов и алгоритмов интеллектуального анализа;
   - структура хранения восстановленных журналов и сопоставленных с ними МАБП;
   - хранилище данных, позволяющее интегрировать систему интеллектуального анализа с существующими СЭД.
6. В предложенной системе интеллектуального анализа реализованы и апробированы на реальных данных алгоритмы восстановления комплексных *WF*-моделей, поиска частых подпоследовательностей.

СПИСОК ЛИТЕРАТУРЫ

1. Шапот, М. Интеллектуальный анализ данных в системах поддержки принятия решений / М. Шапот // Открытые системы. – 1998. – № 1. – С. 30 – 55.

2. Буров, К. Обнаружение знаний в хранилищах данных / К. Буров // Открытые системы. – 1999. – № 5–6.

3. Давыденко, В. Data Mining – интеллектуальный анализ данных / В. Давыденко // Программные продукты и системы. – 2007. – № 3(79). – С. 20 – 31.

4. Корнеев, В.В. Базы данных. Интеллектуальная обработка информации / В.В. Корнеев, А.Ф. Гареев, С.В. Васютин. – М. : НОЛИДЖ, 2001. – 351 с.

5. Загоруйко, Н.Г. Прикладные методы анализа данных и знаний / Н.Г. Загоруйко. – Новосибирск : Изд-во Ин-та математики СО РАН, 1999 – 268 с.

6. Гусев, А. Медицинские информационные системы: анализ рынка / А. Гусев, Ф. Романов, И. Дуданов // PCWeek/RussianEdition. – 2005. – № 47. – С. 18 – 32.

7. Гаврилова, Т.А. Базы знаний интеллектуальных систем / Т.А. Гаврилова, В.Ф. Хорошевский. – СПб. : ПИТЕР, 2001. – 382 с.

8. Батищев, Д.И. Генетические алгоритмы решения экстремальных задач : учеб. пособие / Д.И. Батищев ; под ред. Я.Е. Львовича. – Воронеж, 1995. – 69 с.

9. Вагин, В.Н. Достоверный и правдоподобный вывод в интеллектуальных системах / В.Н. Вагин, Е.Ю. Головина. – М. : Физматлит, 2004.

10. Agrawal, R. Fast algorithms for mining association rules / R. Agrawal, R. Srikant // Proc. Of the 20th Int'l Conference on Very Large Databases, 1994. – P. 487 – 499.

11. Мендель, И.Д. Кластерный анализ / И.Д. Мендель. – М. : Финансы и статистика, 1988. – 176 с.

12. Van der Aalst, W.M.P. Workflow mining: discovering process models from event logs. QUT Technical report, FIT-TR-2003-03 / W.M.P. Van der Aalst, A.J.M.M. Weijters, L. Maruster ; Queensland University of Technology. – Brisbane, 2003.

13. De Medeiros, A.K.A. Process mining for ubiquitous mobile systems: an overview and a concrete algorithm / A.K.A. De Medeiros, B.F. Van Dongen, W.M.P. Van der Aalst, A.J.M.M. Weijters // Ubiquitous Mobile Information and Collaboration Systems (UMICS 2004), Springer-Verlag. – Berlin, 2004. – V. 3272 of Lecture Notes in Computer Science. – P. 154 – 168.

14. Cook, J.E. Discovering models of software processes from event-based data / J.E. Cook, A.L. Wolf // ACM Transactions on Software Engineering and Methodology. – 1998. – V. 7. – I. 3. – P. 215 – 249.

15. Herbst, J. Integrating machine learning and workflow management to support acquisition and adaptation of workflow models / J. Herbst, D. Karagainnis // Proceedings of the Ninth International Workshop on Database and Expert Systems Applications, IEEE. – 1998. – P. 745 – 752.

16. Herbst, J. A machine learning approach to workflow management / J. Herbst // Procceedings 11th European Conference on Machine Learning, Lecture Notes in Computer Science, Springer-Verlag. – Berlin, 2000. – V. 1810. – P. 183 – 194.

17. Herbst, J. Dealing with concurrency in workflow induction / J. Herbst // European Concurrent Engineering Conference, SCS Europe, 2000.

18. Herbst, J. An inductive approach to the acquisition and adaptation of workflow models / J. Herbst, D. Karagiannis // Proceedings of the IJCAI'99 Workshop on Intelligent Workflow and Process Management: The New Frontier for AI in Business, Sweden, Stockholm. – August 1999. – P. 52 – 57.

19. Herbst, J. Integrzting machine learning and workflow management to support acquisition and adaptation of workflow models / J. Herbst, D. Karagiannis // International Journal of Intelligent Systems in Accounting, Finance and Management. – 2000. – V. 9.– P. 67 – 92.

20. Ляпин, Н.Р. Получение комплексных моделей бизнес-процессов на основе журналов их выполнения / Н.Р. Ляпин, Б.С. Дмитриевский // Телекоммуникационные и информационные системы : тр. междунар. конф. – СПб., 2007. – С. 259 – 265.

21. Agrawal, R. Mining process models from workflow logs / R. Agrawal, D. Gunopulos, F. Leymann // Sixth International Conference on Extending Database Technology. – 1998. – P. 469 – 483.

22. Lesh, N. Mining features for sequence classification / N. Lesh, M.J. Zaki, M. Ogihara // Proc. 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining. – 1999. – P. 342 – 346.

23. Dehnert, R.E. Reactive Petri nets for Workflow modeling // Application and theory of Petri nets. – 2003. – P. 285 – 301.

24. Han, J. Mining frequent patterns without candidate generation / J. Han, J. Pei, Y. Yi // Proc. Int. ACM Conf. On Management of Data (SIGMOD'00). – 2000. – P. 1 – 12.

25. Pei, J. PrefixSpan: mining sequential patterns efficiently by prefix projected pattern growth / J. Pei, J. Han, B. Mortazavi-Asl // Proc. 2001 Int. Conf. Data Engineering (ICDE'01), Germany, Heidelberg. – April 2001. – P. 215 – 224.

26. Pei, J. H-Mine: Hyper-structure mining of frequent patterns in large databases / J. Pei, J. Han, H. Lu, S. Nishio, S. Tang, D. Yang // Proc. IEEE Int. Conf. on Data Mining (ICDM'01). – 2001. – P. 441 – 448.

27. Zaki, M. Efficiently mining frequent trees in a forest / M. Zaki // Proc. 8th Int. Conf. On Knowledge Discovery and Data Mining (SIGKD02). – 2002. – P. 71 – 80.

28. Inokuchi, A. An apriori-based algorithm for mining frequent substructures from graph data / A. Inokuchi, T. Washi, H. Moroda // Proc. 4th European Conf. on Principles of Data Mining and Knowledge Discovery. – 2000. – P. 13 – 23.

29. Kuramochi, M. Frequent subgraph discovery / M. Kuramochi, G. Karypis // Proc. IEEE Int. Conf. on Data Mining (ICDM'01). – 2001. – P. 313 – 320.

30. Yan, X. gSpan: graph-based substructure pattern mining / X. Yan, J. Han // Proceedings of the 2002 IEEE International Conference on Data Mining (ICSM 2002), IEEE Computer Society. – 2002. – P. 721 – 724.

31. Yan, X. CloseGraph: mining closed frequent graph patterns / X. Yan, J. Han // Proc. ACM Int. Conf. on Knowledge Discovery and Data Mining (KDD'03). – 2003. – P. 286 – 295.

32. Koksal, P. Workflow history management / P. Koksal, S.N. Arpinar, A. Dogac // SIGMOD Record. – 1998. – V. 27. – I. 1. – P. 67 – 75.

33. Van der Aalst, W.M.P. Workflow management: models, methods, and systems / W.M.P. Van der Aalst, K.M.Van Hee // MIT Press. – 2002.

34. Cook, J.E. Automating process discovery through event-data analysis / J.E. Cook, A.L. Wolf // Proc of the 17th Conference on Software Engineering. – Seattle, Washington, 1995.

35. Han, J. Mining frequent patterns without candidate generation / J. Han, J. Pei, Y. Yi // Proc. Int. ACM Conf on Management of Data (SIGMOD'00). – 2000. – P. 1 – 12.

36. Ляпин, Н.Р. Разработка информационной системы восстановления моделей автоматизированных бизнес-процессов / Н.Р. Ляпин, Б.С. Дмитриевский // Программные продукты и системы. – 2007. – № 3(79). – С. 80 – 81.

37. Фаулер, М. Архитектура корпоративных программных приложений / М. Фаулер ; пер. с англ. – М. : Издательский дом «Вильямс», 2004. – 544 с.

38. Олищук, А. Теория разработки framework-систем / А. Олищук // PHP Inside. – 2004. – № 9. – С. 5 – 18.

39. Аллен, Э. Типичные ошибки проектирования / Э. Аллен. – СПб. : Питер, 2003.

40. Холтыгина, Н.А. Обобщение модели проектирования сложных систем на основе объектно-ориентированного подхода / Н.А. Холтыгина // Надежность инструмента и оптимизация технологических систем. – Краматорск : ДГМА, 2003. – С. 209 – 215.

41. Бертран, М. Объектно-ориентированное конструирование программных систем / М. Бертран. – М. : Русская Редакция, 2005. – 1204 с.

42. Грэхем, И. Объектно-ориентированные методы. Принципы и практика / И. Грэхем. – М. : Диалектика-Вильямс, 2004. – 880 с.

43. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. – СПб. : Питер, 2006. – 366 с.

44. Шаллоуей, А. Шаблоны проектирования. Новый подход к объектно-ориентированному анализу и проектированию / А. Шаллоуей, Р. Трот Джейс ; пер. с англ. – М. : Вильямс, 2002. – 288 с.

45. Соммервилл, И. Инженерия программного обеспечения / И. Соммервилл ; пер. с англ. – 6-е изд. – М. : Издательский дом «Вильямс», 2002. – 624 с.

46. Толстов, Е.В. Использование объектно-реляционного отображения для связи с базами данных / Е.В. Толстов // Моделирование процессов управления : сб. ст. / Моск. физ.-техн. ин-т. – М., 2004. – С. 109 – 115.

47. Волков, Л.М. Хронологические структуры данных. Способы представления в памяти / Л.М. Волков // Компьютерные науки. – Екатеринбург : Известия УрГУ, 2006. – № 1. – С. 15 – 25.

48. Волков, Л.М. Задачи целостности для хронологических структур данных / Л.М. Волков // Проблемы теоретической и прикладной математики : труды 34-й региональной молодежной конференции. – Екатеринбург : УрО РАН, 2003. – С. 250 – 253.

49. Тенненбаум, Э. Распределенные системы. Принципы и парадигмы / Э. Тенненбаум, М. Ван Стен. – СПб. : Питер, 2003. – 877 с. ; ил. – (Серия «Классика Computer Science»).

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ ……..... 3
Глава 1. ИНТЕЛЛЕКТУАЛЬНЫЙ АНАЛИЗ ДАННЫХ …….. 5
1.1. Классификация задач интеллектуального анализа …… 6
1.2. Практическое применение интеллектуального анализа 10
1.3. Модели интеллектуального анализа …… 13
1.4. Методы интеллектуального анализа ……... 15
1.5. Процесс обнаружения знаний …….. 17
1.6. Задача поиска ассоциативных правил ……. 18
1.7. Кластеризация …… 27
Глава 2. РАЗРАБОТКА СИСТЕМЫ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ВЫПОЛНЕНИЯ БИЗНЕС-ПРОЦЕССОВ В ИНФОРМАЦИОННОЙ СРЕДЕ СОВРЕМЕННОГО ПРЕДПРИЯТИЯ ……. 33
2.1. Разработка алгоритма получения комплексных МАБП … 34
2.2. Определение терминов ……. 35
2.3. Алгоритм получения комплексных МАБП ……. 37
2.4. Разработка алгоритма получения частых подпоследовательностей …… 40
2.5. Проектирование и реализация системы интеллектуального анализа выполнения бизнес-процессов в системе электронного документооборота ……. 56
ЗАКЛЮЧЕНИЕ …… 87
СПИСОК ЛИТЕРАТУРЫ …… 88

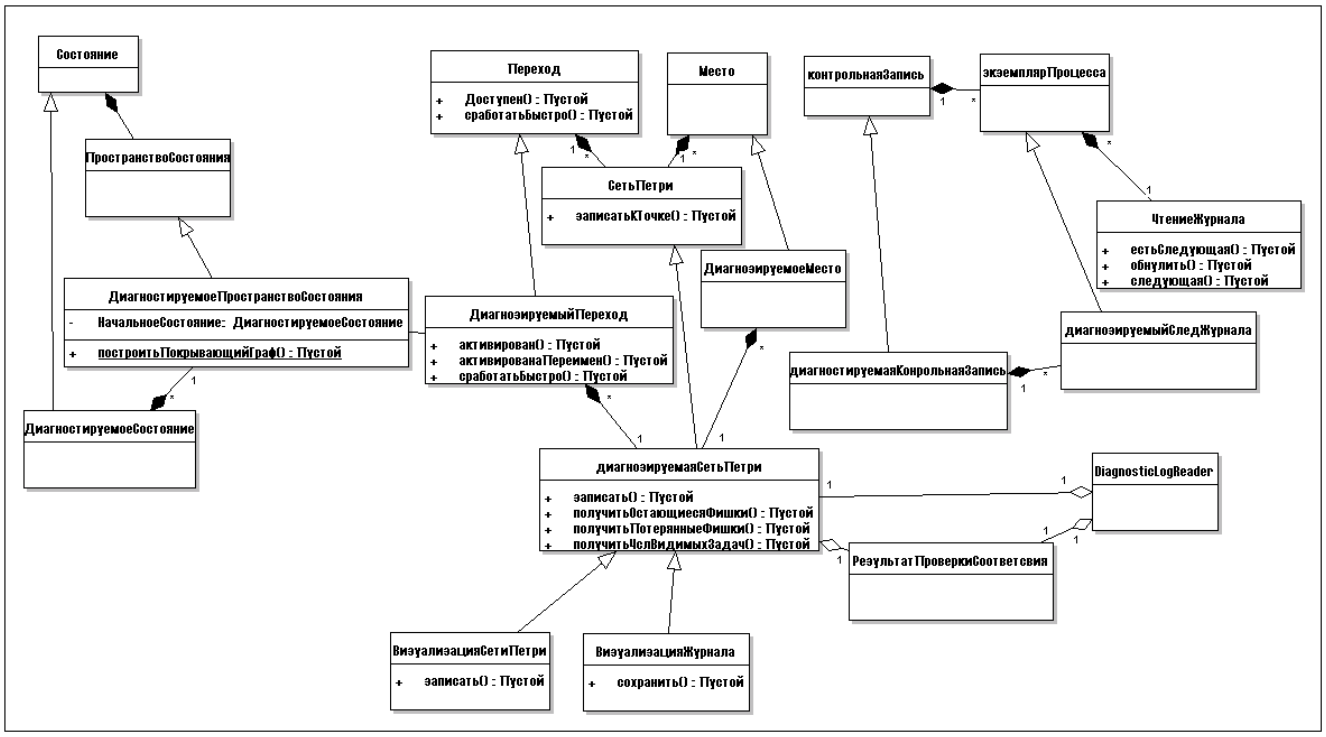

Рис. 24. Диаграмма классов для системы интеллектуального анализа выполнения бизнес-процессов в электронном документообороте

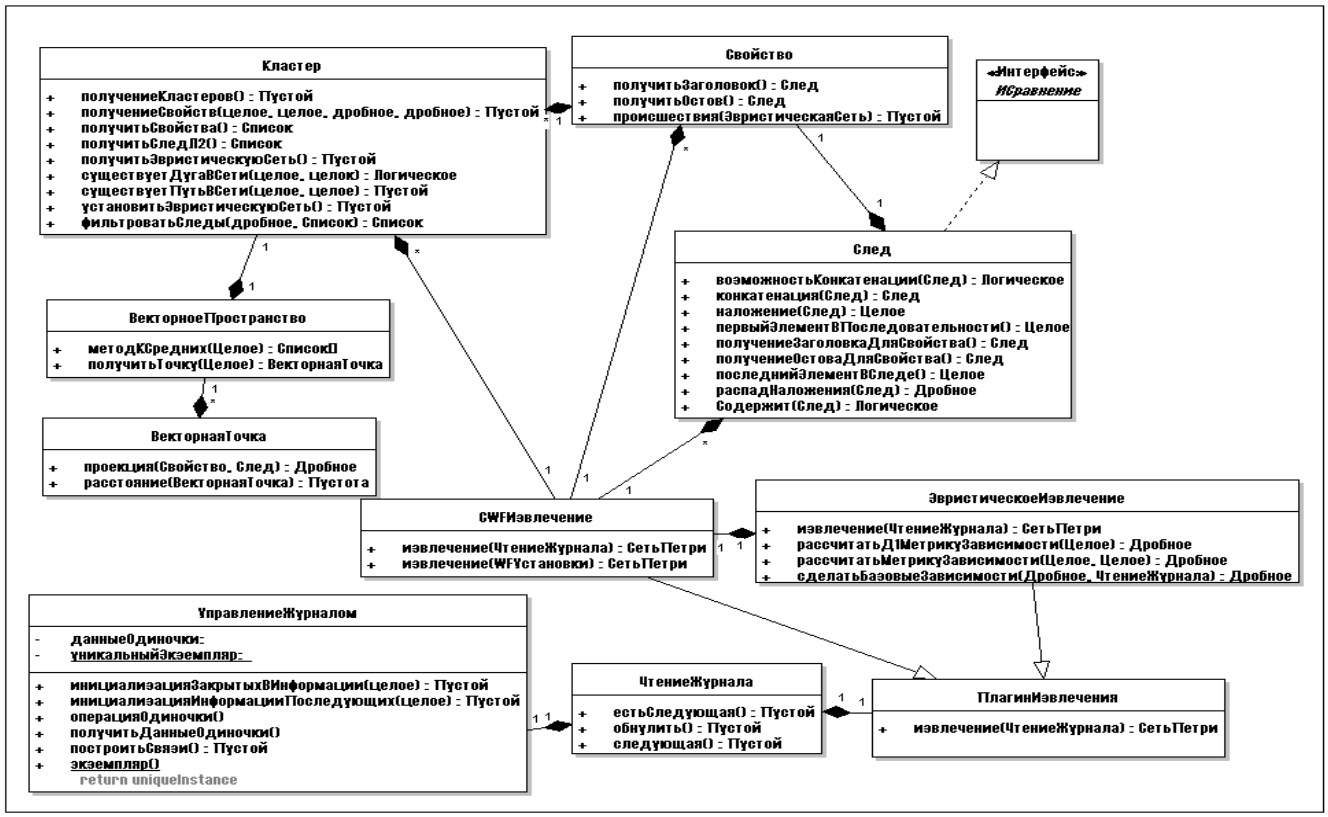

Рис. 25. Диаграмма классов плагина для восстановления комплексных МАБП