

Федеральное государственное учреждение
Федеральный исследовательский центр
«Информатика и управление» Российской академии наук

На правах рукописи

Молодченков Алексей Игоревич

**МОДЕЛИ, МЕТОДЫ И АРХИТЕКТУРЫ
ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ ПОДДЕРЖКИ
ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ (НА ПРИМЕРЕ
МЕДИЦИНЫ И ПСИХОЛОГИИ)**

Специальность: 05.13.01 – Системный анализ, управление и обработка информации (информационно-вычислительное обеспечение)

**Диссертация на соискание ученой степени
кандидата технических наук**

Научный руководитель –
доктор физико-математических наук,
профессор Г.С. Осипов

Москва – 2017

Оглавление

Введение	4
Глава 1. Общее представление о технологических процессах	11
1.1. Технологический процесс: определение и структура	11
1.2. Особенности технологических процессов в гуманитарных областях на примере медицины и психологии.....	13
1.3. Применение компьютерных технологий для моделирования и поддержки технологических процессов (на примере медицины и психологии)	17
1.4. Выводы.	52
Глава 2. Модели технологических процессов	56
2.1. Модель технологического процесса.....	56
2.2. Маршруты в технологических процессах	63
2.3. Графическое представление технологического процесса	65
2.4. Онтологический подход для описания стандартов лечения и протоколов ведения больных.....	68
2.5. Выводы	74
Глава 3. Построение общего описания технологического процесса на основе прецедентной информации.....	75
3.1. Алгоритм автоматического синтеза общего описания технологического процесса на основе прецедентной информации	75
3.2. Персонализация медицинских технологических процессов	89
3.3. Экспериментальные исследования алгоритма синтеза обобщенной схемы МТП	104
3.4 Выводы	108

Глава 4. Архитектуры интеллектуальных средств поддержки технологических процессов.....	109
4.1. Архитектура системы поддержки медицинских технологических процессов	109
4.3. Архитектура системы поддержки технологических процессов в области практической психологии	112
4.4. Неоднородные семантические сети как средство описания знаний в области практической психологии.....	114
4.5. Основные структуры данных.....	120
4.6. Выводы	124
Заключение.....	126
Список литературы.....	128
Приложение А.....	145
Приложение Б	159
Приложение В.....	164

Введение

Актуальность темы исследования. Процессный подход к организации управления предприятиями в различных отраслях экономики сегодня признан наиболее перспективным. В последние годы появилось немало работ по изучению возможностей использования систем управления бизнес-процессами для автоматизации медицинских технологических процессов (МТП). Возник и соответствующий термин – «*careflow*» по аналогии с «*workflow*», означающий поток работ, направленных на оказание медицинской помощи.

Сегодня существует три основных направления исследований в области автоматизации медицинских технологических процессов:

- 1) электронные медицинские карты в различных вариантах (включая карты здоровья, карты пациентов;
- 2) поддержка принятия решений на основе клинических руководств;
- 3) управление клиническими процессами (электронные клинические пути – э-КП).

Электронные медицинские карты (ЭМК) постепенно преобразуются из простого хранилища клинических данных о пациенте в многофункциональную систему, позволяющую организовывать, распределять и использовать медицинские знания с высокой степенью защищенности, облегчать и регулировать взаимодействие всех участников лечебно-диагностических процессов

Исследования в области систем поддержки принятия решений (СППР) в последние два десятилетия были направлены, в основном, на разработку компьютеризированных клинических руководств. СППР на основе клинических руководств призваны представлять клиницистам контекстные рекомендации в нужный момент, обеспечивать взаимосвязь с ЭМК и другими информационными системами, осуществлять мониторинг этапов оказания медицинской помощи

Третье направление исследований берет начало от концепции клинических путей (*clinical pathways*), предложенной в 80-х годах XX века в качестве метода управления клиническими случаями и медсестринской помощью в Медицинском центре Новой Англии, в Бостоне. Клинический путь представляет собой пациент-ориентированный план лечебных мероприятий, которые должны быть выполнены за один эпизод оказания медицинской помощи. КП имеют мультидисциплинарный характер и применяются для внедрения клинических руководств в клиническую практику, а также для снижения нежелательных отклонений от медицинских стандартов.

Несколько иным образом дело обстоит в психологии. Основные усилия по внедрению информационных технологий направлены на разработку компьютеризированных психодиагностических методик, экспертных систем, интерпретаторов результатов тестирований, моделей прогноза, психологического сопровождения конкретных видов деятельности.

Анализ существующих систем и методов моделирования и поддержки технологических процессов в области медицины и психологии показал, что они не учитывают особенности конкретного пациента (клиента в психологии). Более того, во всех системах, направленных на поддержку технологических процессов в области медицины и психологии отсутствуют средства оценки качества оказанной медицинской и психологической помощи, анализа отклонений от технологических процессов, выявление причин отклонений от процесса лечения, средств управления технологическими процессами и адаптации МТП под конкретного пациента. Настоящая работа посвящена разработке методов, алгоритмов и архитектур программных средств поддержки технологических процессов, в области медицины и психологии, что свидетельствует о её актуальности.

Цели и задачи исследования

Целью диссертационной работы является разработка новых методов и алгоритмов анализа и поддержки технологических процессов в области медицины и психологии.

Для достижения поставленной цели в работе решены следующие задачи:

- 1) выполнен анализ существующих методов и программных средств поддержки технологических процессов в области медицины и психологии;
- 2) предложены способы представления технологических знаний с использованием онтологий и неоднородных семантических сетей;
- 3) разработан новый алгоритм автоматического синтеза обобщенной схемы технологических процессов на основе прецедентной информации;
- 4) предложены архитектуры программных средств поддержки технологических процессов в области медицины и психологии.

Методы исследования

В диссертационной работе использованы методы искусственного интеллекта, машинного обучения, теории множеств, математической логики, интеллектуального анализа данных, интеллектуального анализа процессов.

Научная новизна работы и результаты, выносимые на защиту

Современные подходы в области поддержки медицинских технологических процессов направлены на разработку систем, решающих узкий круг задач. К таким системам относятся системы поддержки принятия решений, формализации клинических руководств, лекарственных назначений и др. Аналогичная ситуация складывается и в практической психологии, где все разработки направлены на компьютеризацию диагностических методик в психодиагностике и др. В последние годы наблюдаются попытки моделирования медицинских технологических процессов. Однако построенные

модели направлены на лечение заболевания, а не больного. В отличие от этого, предлагаемые в настоящей работе методы позволяют синтезировать технологические процессы, являющиеся персонализированными и учитывающими особенности конкретного больного.

Такой подход становится возможным благодаря тому, что широкое применение медицинских информационных систем позволяет накапливать информацию о лечении нозологических форм конкретных пациентов. Эта информация содержит данные о состояниях больных, лечебно-диагностических мероприятиях, проводимых с больными, и др. Если эта информация становится доступной, то можно строить обобщенные схемы медицинских технологических процессов, которые будут учитывать индивидуальные особенности пациентов, а на их основе персонализированные МТП.

Исследователи в области анализа процессов занимаются разработкой методов и алгоритмов автоматического построения моделей технологических процессов на основе прецедентной информации. Однако современные алгоритмы автоматического построения моделей технологических процессов только частично решают проблемы петель, коротких циклов, несвободного выбора, устранения шумов и др., что отражается на качестве их результатов. Решение этой задачи также является одной из целей настоящей работы.

Основные результаты работы:

- впервые предложено матричное представление медицинских технологических процессов;
- доказаны утверждения об ассоциативности и некоммутативности последовательного применения произвольного числа операторов МТП;
- введена операция покомпонентного сложения матриц смежности экземпляров медицинских технологических процессов в задаче

автоматического синтеза обобщенной схемы технологических процессов;

- разработан новый алгоритм построения обобщенной схемы технологического процесса на основе прецедентной информации;
- предложен метод выявления точек ветвления условного маршрута на основе классификации экземпляров технологических процессов;
- разработаны архитектуры и реализованы программные средства автоматического синтеза обобщенных технологических процессов и методы их персонализации.

Практическая значимость работы

Предложенные методы и подходы к разработке систем поддержки технологических процессов могут быть использованы в крупных медицинских учреждениях для построения систем управления медицинскими технологическими процессами, в университетах для обучения студентов лечебных и психологических специальностей, в крупных частных компаниях для подбора персонала и оценки психологического состояния сотрудников, в частных компаниях, внедривших системы управления бизнес-процессами, и др.

Апробация работы

Основные положения работы докладывались и обсуждались на международных научных и научно-практических конференциях: «Интеллектуальный анализ и информационные технологии», «Системный анализ и информационные технологии», EANS congress, «Теория и практика системного анализа», «Информационные технологии в медицине», «Нечеткие системы и мягкие вычисления», «Информационные и телекоммуникационные системы: информационные технологии в научных и образовательных процессах», международная конференция по искусственному интеллекту (КИИ), всероссийская конференция по проблемам математики, информатики,

физики и химии, на семинарах ИСА РАН и Медицинского центра Центрального банка Российской Федерации.

Публикации

Основные результаты, полученные по теме диссертационной работы, опубликованы в 12 печатных работах (в том числе 5 публикаций в ведущих рецензируемых научных изданиях, рекомендованных Высшей аттестационной комиссией, 7 публикаций в трудах научных конференций).

Структура и объем работы

Диссертация состоит из введения, четырех глав, заключения, списка использованной литературы и двух приложений. Диссертация содержит 144 страницы, 28 рисунков, 12 таблиц, 146 наименований в списке используемой литературы.

Основное содержание работы

В первой главе приведено общее описание технологических процессов, указаны особенности технологических процессов в области медицины и психологии. Проведен анализ существующих методов и систем поддержки технологических процессов в области медицины и психологии. Описаны методы и алгоритмы автоматического построения модели технологического процесса на основе прецедентной информации и жизненных циклов объектов.

Вторая глава посвящена описанию модели технологического процесса на основе операторной теории медицинских технологических процессов, предложенной Г.С. Осиповым и Г.И. Назаренко.

В третьей главе описан алгоритм автоматического синтеза обобщенной схемы персонализированного медицинского технологического процесса на основе прецедентной информации. Приведены результаты экспериментальных исследований предложенного алгоритма.

В четвертой главе описаны архитектуры систем поддержки технологических процессов в области медицины и психологии, представлен онтологический подход к представлению знаний.

В заключении приведены основные результаты, полученные в диссертационной работе.

В приложении А описан комплекс экспертных систем «Психология», предназначенный для поддержки процессов психодиагностики и психологического консультирования.

В приложении Б описаны основные составляющие системы поддержки процесса принятия решения о проведении коронарного вмешательства.

В приложении В предложен программный код и описание основных функций программной реализации алгоритмов автоматического синтеза медицинских технологических процессов.

Глава 1. Общее представление о технологических процессах

В настоящей главе описывается понятие технологического процесса и его компонентов. Рассматриваются модели, методы и программные средства поддержки технологических процессов в области медицины и психологии.

1.1. Технологический процесс: определение и структура

Технологический процесс (ТП) представляет собой последовательность технологических операций, необходимых для выполнения определенного вида работ. Технологический процесс, как правило, хорошо структурирован. Он делится на несколько подпроцессов – частных технологических процессов. Основными отличительными чертами частного технологического процесса являются [1]:

- 1) взятый в отдельности частный технологический процесс базируется на общих представлениях и, как правило, происходит в пределах одного рабочего места, например, операционного стола;
- 2) частный технологический процесс не может обеспечить весь комплекс действий, позволяющих достичь конечного результата в конкретном случае;
- 3) частный технологический процесс подразумевает новации, как теоретические, так и практические, направленные на его совершенствование;
- 4) частному технологическому процессу соответствует конкретная специальность (специалист лучевой диагностики, хирург, офтальмолог, врач – лаборант, психодиагност и т.п.).

Структура технологического процесса являет собой не какие-то случайные построения, а закон развития, закон перехода от простого к сложному [1]. Технологический процесс, возникнув на предметно-организационном уровне при своем развитии, с одной стороны, дробится (дифференцируется, специализируется), переходя на более глубокие уровни

организации, с другой синтезируется (кооперируется, интегрируется, усложняется) по мере расширения сферы анализа или управления.

Термин «технологический процесс», а тем более «процессы», нельзя толковать и понимать узко. Эти понятия сложны и широки, охватывают целые системы, обладающие сложной структурой.

В отличие от частных технологических процессов, комплекс которых составляет физическую специфическую сущность технологического процесса, *технологические операции* отражают организационно-экономическую его сторону.

Отличительным свойством технологических операций является их упорядоченность, которую следует трактовать в том смысле, что в рамках одного рабочего места к следующей операции не приступают до окончания предыдущей. Действия внутри технологической операции характеризуются строгой последовательностью, взаимосвязью и взаимозависимостью. Таким образом, технологическая операция представляет собой чередующиеся действия, обеспечивающие заданное качество в рамках конкретного частного технологического процесса.

На основании изложенного можно заключить следующее[1]:

- 1) современный технологический процесс представляет собой систему, структурированную как по вертикали, так и по горизонтали;
- 2) структура технологического процесса иерархична. С понижением уровня иерархии растет детализация, а с его повышением – степень общности;
- 3) по мере увеличения уровня организации технологического процесса ему в большей степени присущи экономическая, социальная и управленческая направленность. На средних уровнях находятся методы диагностики и лечения, на глубоких – технология сливается с естественными процессами и изучающими их науками;
- 4) технологический процесс складывается из ряда параллельно протекающих циклов, фаз, стадий, технологических операций,

- отвечающих логике их сочетания, ориентированных на конечный результат и образующих горизонтальные уровни его организации;
- 5) структура технологического процесса диктует необходимость четкого разграничения всех уровней его организации и разработки на этой базе адекватного понятийного и методического аппарата.

1.2 Особенности технологических процессов в гуманитарных областях на примере медицины и психологии

Гуманитарные области (медицина, психология, социология и др.) являются трудноформализуемыми и обладают своими отличительными особенностями. Для этих областей не известен полный набор свойств объектов, не полностью известна структура самих объектов и взаимосвязи между ними. Не существует заранее заданного алгоритма решения задачи лечения пациента. Не существует точной математической модели задачи. В медицине существует ряд трудностей, с которыми сталкиваются специалисты при проектировании технологических процессов:

- 1) распределенность. Во многих ситуациях в процесс может быть вовлечено несколько медицинских учреждений. Каждое из этих учреждений обладает своей системой управления технологическими процессами. Требуется интеграция со всеми этими системами;
- 2) адаптация к конкретному больному. Процесс лечения пациента с определенным заболеванием должен быть адаптирован к индивидуальным особенностям пациента и иметь возможность перехода и интеграции с другим процессом лечения. Эта ситуация может возникнуть если пациенту в начале лечения был поставлен один диагноз, который в процессе лечения был скорректирован и изменен, или в случае когда у пациента помимо основного заболевания присутствуют сопутствующие;

- 3) недетерминированность. Состояния объектов не неизбежно связаны с действиями или событиями. В технологических процессах объекты являются пассивными, переход объекта из одного состояния в другое осуществляется выполнением определенных действий. В отличие от этого МТП имеют дело с пациентами – сложными активными объектами, изменения внутреннего состояния которых должны, как правило, учитываться в процессе выполнения медицинского технологического процесса;
- 4) необратимость. Большинство действий в медицинских технологических процессах являются необратимыми. Это означает, что далеко не для каждого действия А существует обратное ему, переводящее систему в то состояние, в котором она находилась до применения действия А;
- 5) процессы основаны на ожидании. В связи с недетерминированностью природы человека совместно с неполнотой диагнозов и с тем, что процессы лечения основаны на популяционных исследованиях, в большинстве случаев при лечении пытаются достичь определенных исходов. Отсюда постуловия действий иногда рассматриваются как ожидаемые. В связи с этим в зависимости от характера действия, состояния пациента и обобщенной схемы процесса лечения, последовательность постуловий, которые определены и обрабатываются, требует особого рассмотрения;
- 6) свойства и поведение участников технологического процесса не определены заранее. В медицинских технологических процессах не маловажную роль играют такие свойства и поведение участников, как опыт, текущая роль, доступность, возможность назначать лечение и принимать решения и т.д. Пренебрегать данными особенностями нельзя.

Что касается практической психологии, то все технологические процессы в данной области обладают особенностью, которая не только не позволяет четко их смоделировать, но и создает значительные трудности для разработки

систем поддержки данных процессов. Весь процесс построен на диалоге между специалистом психологом и клиентом. Невозможно расписать, о чем должны говорить психолог и клиент и что при этом делать. Как и в медицине, участником всех процессов является человек. А значит, что при создании систем моделирования и поддержки технологических процессов в области практической психологии, разработчики будут иметь дело с недетерминированным объектом, состояния которого не зависят напрямую от выполняемых действий. Все процессы должны быть адаптированы к индивидуальным особенностям каждого конкретного клиента. Почти все выполняемые действия необратимы, если клиент выполнил определенные манипуляции или каким-либо образом отреагировал на просьбы специалиста, то невозможно психолога заставить все забыть или не учитывать это в дальнейшей своей работе с клиентом. Немаловажную роль играет опыт специалиста психолога. В связи с этим, характерной особенностью разработанных систем в области психологии является их узкая специализация – в определённой сфере психодиагностики (школьной, клинической и т.п.) выделяется предметная область, для которой возможно, с точки зрения разработчиков, провести отбор критериев, по которым психолог-эксперт принимает решение. Эти критерии далее используются для организации диалога с пользователем, что позволяет, минуя работу профессионала по выдвижению и проверке психодиагностических гипотез о проблеме клиента, переходить сразу к этапу рекомендаций. В сложных случаях может быть предложено дополнительное обследование или взаимодействие со специалистом, но в целом характер рекомендаций ориентирован на принятие решения самим клиентом или заказчиком диагностики.

Несмотря на то, что все технологические процессы в области практической психологии построены на диалоге специалиста с клиентом, в них можно выделить некоторые закономерности и этапы. Например, процесс психодиагностики можно разделить на следующие этапы [2]:

- 1) формулирование практического запроса;
- 2) формулирование гипотез и выбор диагностических методов их проверки;
- 3) диагностическое обследование и обработка, количественный и качественный анализ его результатов;
- 4) формулирование основных выводов, заключений на основе полученных результатов, а также составление психологического прогноза;
- 5) определение характера и вида конкретной психологической помощи.

В практическом запросе формулируется проблема обратившегося за помощью (или направленного к специалисту) человека. Запрос формулируется на естественном языке. Основная задача диагноста на первом этапе – это перевести запрос на язык профессиональной психологической практики и сформулировать для себя непосредственную профессиональную задачу, внутри которой выделить психодиагностическую задачу. На втором этапе специалист на основе имеющейся информации о клиенте формулирует предположения и выбирает инструментарий, позволяющий их проверить. В завершении работы с клиентом, специалист выбирает характер и вид психологической помощи, адекватной полученным результатам и соответствующей реальным условиям жизни клиента.

Несмотря на указанные особенности технологических процессов в медицине и психологии и сложности их проектирования современные достижения в данных областях, информатике и искусственном интеллекте позволяют разрабатывать методы поддержки технологических процессов в области медицины и психологии.

1.3. Применение компьютерных технологий для моделирования и поддержки технологических процессов (на примере медицины и психологии)

Активно развивающиеся информационные технологии охватывают все больше сфер человеческой деятельности. В конце XX века это стало непосредственно касаться медицины и психологии.

1.3.1. Применение компьютерных технологий для моделирования и поддержки технологических процессов в медицине

Одними из первых разновидностей систем поддержки некоторых этапов медицинских технологических процессов являются экспертные системы, которые возникли в середине восьмидесятых годов прошлого века в результате 30-летнего академического периода исследований в области искусственного интеллекта.

Первой собственно медицинской экспертной системой стала система **Mycin** [3], предназначенная для диагностики и лечения инфекционных заболеваний крови.

Система **CASNET** [4] предназначена для диагностики и выбора стратегии лечения глаукомы.

Система **DXplain** [5] – пример системы поддержки клинических решений, используется для ассистирования в процессе диагностики и содержит в своей базе знаний симптомы, лабораторные данные и процедуры, связывающие их со списком диагнозов. Она обеспечивает поддержку и обоснование дифференциальных диагнозов и последующие исследования. В её базе данных содержится 4500 клинических манифестаций, которые связаны ассоциативными связями более чем с 2000 различных нозологий.

Система **Germwatcher** [6] была разработана в помощь больничному эпидемиологу. Содержит большой объем данных по различным микробиологическим культурам. Включает базу знаний, основанную на

правилах, которая используется для генерации гипотез о возможных инфекциях.

Система **PEIRS** [7] интерпретирует и комментирует отчеты по химическим патологиям. В систему встроен модуль автоматического машинного обучения, который позволяет патологу создавать новые правила без участия инженера по знаниям. В настоящее время создано 2300 таких правил. На построение каждого нового правила требуется около минуты времени. Ежедневно система комментирует 100 отчетов в области газового состава артериальной крови, теста толерантности глюкозы и др.

Система **HELP** [8] – полная госпитальная информационная система, основанная на технологиях экспертных систем. Она поддерживает не только стандартные функции госпитальных информационных систем, но функции поддержки принятия решений. Эти функции инкорпорированы в рутинные приложения госпитальной системы. Они поддерживают клинический процесс тревожными сигналами и напоминаниями, интерпретацией данных, выработкой предложений по управлению процессом лечения и клиническими протоколами. Эти функции могут активироваться из обычных приложений или включаться самостоятельно после ввода клинических данных в компьютерную историю болезни.

Отметим ещё систему **SETH** [9], область применения которой – токсичность лекарственных средств. Система основана на моделировании экспертных рассуждений, берущих в расчет для каждого токсикологического класса клинические симптомы и применяемые дозы. Система выполняет мониторинг лечебного процесса, направленный на контроль взаимодействия взаимоисключения лекарств.

Среди отечественных разработок отметим автоматизированную систему для синдромной диагностики неотложных состояний у детей **ДИН** [10], созданную в Московском НИИ педиатрии и детской хирургии. Эта система содержит информацию о 42 синдромах, которые представляют собой список

диагностических предложений-гипотез. Так как выбор лечения во многом определяется прогнозом возможных осложнений, в системе описаны взаимосвязи синдромов, определяемые причинно-следственными, временными и ассоциативными отношениями.

Несомненный интерес представляет программный комплекс **Айболит** [11] для диагностики, классификации и коррекции терапии острых расстройств кровообращения у детей, созданный в Центре сердечнососудистой хирургии имени А.Н. Бакулева и активно применяемый при оперативных вмешательствах и выборе послеоперационного лечения в условиях реанимационного отделения. Система включает математическую модель кровообращения, «реагирующую» на поступающую с датчиков текущую информацию. Она позволяет не только проводить диагностику и оценку состояния больного, но и помогать при выборе и последующей коррекции лечебных мероприятий.

Дальнейшее развитие информационных технологий позволило разрабатывать системы автоматизации и моделирования медицинских технологических процессов. Одной из основных задач, возникающих при моделировании МТП является разработка языка описания технологических процессов. Одним из самых распространенных языков описания МТП является Asbru [12-14].

Определение 1.1. [12] Asbru – это язык описания лечебно-диагностических процессов, клинических протоколов в виде планов лечения, упорядоченных во времени.

В Asbru поддерживаются следующие разделы плана: Ограничения (Preferences), Цели (Intentions), Условия (Conditions), Эффекты (Effects) и тело плана (действия). В разделе Ограничения накладываются ограничения на применимость плана (например, выбор критериев: точное или не точное соответствие) и выбирается вид поведения плана (например, вид стратегии: агрессивная или не агрессивная). В разделе Цели указываются цели, которые должны быть достигнуты, поддерживаются или необходимо избегать во время

исполнения плана. Условия (Conditions) необходимы для того, чтобы план можно было запустить, приостановить, возобновить его работу, прервать или завершить. В разделе Эффекты с помощью математических функций описываются взаимоотношения между аргументами плана и измеряемыми параметрами. Также задается вероятность возникновения событий. Тело плана содержит действия или другие планы, которые могут быть выполнены, если выполнены предварительные условия. Если план содержит в себе другие планы, то эти планы выполняются в соответствии с условиями, указанными в типе плана: в строгой последовательности, в каком-либо порядке, параллельно, периодически (пока выполнено условие, максимальное количество раз, и с минимальным интервалом между попытками), или неупорядоченно (то есть, не указано никаких ограничений).

Важную роль в определении комплекса временных аспектов планов играют так называемые временные ограничения (TimeAnnotations). Временное ограничение (TimeAnnotation) определяет четыре момента времени относительно опорной точки (которая может быть конкретным или абстрактным моментом времени, или переходом плана из одного состояния в другое): самое раннее время начала выполнения действия (ESS), самое позднее время начала выполнения действия (LSS), самое раннее время завершения выполнения действия (EFS) и самое позднее время завершения выполнения действия (OPC). Вместе взятые, эти данные указывают на временные ограничения, в рамках которых действие имеет место или условия должны быть выполнены.

Однако, с появлением языка разметки XML, который намного проще языка Asbru, и аппарата XML Schema [15] очень многие современные медицинские системы стали использовать именно XML для описания медицинских технологических процессов и клинических руководств, обмена информацией между медицинскими системами, как средство сериализации медицинских данных. Большинство программных продуктов, которые

работают с лечебно-диагностическими процессами или клиническими руководствами, описанными на языке Asbru, сначала конвертируют Asbru описание в формат XML и обратно с помощью специальных XML редакторов. Одним из таких редакторов является PEXEE.

PEXEE (Pontifex Intelligent XML Editor Extension) – это XML редактор, который является надстройкой над Pontifex [16] и конвертирует иерархию классов и документацию из спецификации языка в формат XML. В добавлении к типам DTD (document type description) он поддерживает различные типы в атрибутах (число, число с плавающей точкой и т.п.), пространства имен в атрибутах ID и IDREF и простую форму наследования. XML Schema [15] более сильный аппарат.

После разработки языков описания медицинских технологических процессов, стали появляться системы моделирования и автоматизации МТП. Одной из первых систем автоматизации МТП является **OzCare**[17], которая позволяет описывать планы лечения для конкретного пациента. OzCare использует медицинские логические модули Колумбийского Университета (Нью-Йорк), разработанные для поддержки исполнения клинических руководств, и язык описания клинических руководств ArdenSyntax. Шаблоны планов лечения, к которым прилагаются соответствующие медицинские логические модули, построены на основе правил, применяемых к элементам лечения. Эти шаблоны могут быть подогнаны под конкретного пациента. Позднее функциональность была расширена добавлением поддержки событий и очередей сигналов с использованием обычного таймера Unix.

MOBILE [18] – модульная система управления МТП предназначена для поддержки адаптации технологического процесса к индивидуальным особенностям и потребностям пациентов в процессе его выполнения. В системе поддерживаются следующие способы выполнения лечебных действий:

- последовательное;
- условное;

- параллельное.

Последовательное выполнение действий а и b технологического процесса Т означает, что действие b выполняется только после того, как завершится выполнение действия а. Условное выполнение действий означает, что если определенные условия истинны, то выполняется действие а, иначе действие b. Параллельное выполнение действий а и b означает, что действия а и b выполняются в одно и то же время. MOBILE позволяет накладывать на действия и подпроцессы различные ограничения, которые делятся на два типа: временные и условия существования или выполнения. Одним из временных ограничений является Deadline. Данное ограничение заключается в том, что если имеются два действия а и b и действие а начинает выполняться раньше, чем действие b, тогда действие а должно завершить свое выполнение до того, как начнет выполняться действие b, или прерваться, если это возможно. В случае, когда действия а и b могут выполняться параллельно, ограничение Deadline не накладывается.

Следующее временное ограничение называется «Задержка» (“Delay”). Оно заключается в том, что если имеются два действия а и b и действие а начинает выполняться раньше, чем действие b, тогда начало выполнения действия b откладывается до тех пор, пока не выполнится действие а. Система MOBILE используется не только в медицине, но и других областях.

METEOR [19] (Managing End-To-End Operations) – программный комплекс для создания и исполнения технологических процессов. Особенностью этого программного продукта является то, что он может поддерживать скоординированные распределенные процессы. METEOR был запущен в тестовом режиме в Фонде медицинского образования и исследований в Коннектикуте для моделирования и внедрения технологических процессов отслеживания иммунизации детей. Архитектурные решения METEOR сосредоточены на улучшении реализации в распределенных средах и

используют технологию CORBA. На данный момент METEOR продолжает развиваться как METEOR-S с использованием семантических web-сервисов.

CareFlow Management System (CfMS) [20, 21] – система управления технологическими процессами, разработанная в Университете Павии. В медицине она применяется для управления МТП, основанными на клинических руководствах. Одним из последних применений CareFlow явилось ее использование для лечения хронических заболеваний, таких как диабет. Разработки CareFlow являются законченными приложениями, соединяющими в себе управление технологическими процессами с клиническими руководствами (в качестве ресурса, описывающего процесс) и электронными медицинскими записями (в качестве источника и хранилища данных).

METUFlow [22] – прототип реализации распределенной системы управления технологическими процессами, которая использует распределенное планирование, мониторинг и модули управления списками работ совместно со службами распределенных транзакций. Весь технологический процесс делится на блоки, задачи и подпроцессы. Задача представляет собой самое простое выполняемое действие. Блок включает в себя набор действий, выполняемых в соответствии с его типом, и описание зависимостей между действиями. В системе содержатся следующие типы блоков:

- 1) «Последовательный» блок. Все действия, находящиеся в этом блоке, выполняются последовательно один за другим. Работа блока завершается, когда успешно завершается выполнение последнего действия в блоке. Если выполнение какого-либо действия в блоке завершается некорректно (аварийно, принудительно), то и некорректно завершается работа всего блока;
- 2) «И Параллельный» блок. Все действия, находящиеся в этом блоке, выполняются параллельно. Работа блока завершается, когда успешно завершается выполнение всех действий в блоке. Если выполнение какого-либо действия в блоке завершается некорректно

- (аварийно, принудительно), то и некорректно завершается работа всего блока;
- 3) «Или Параллельный» блок. Все действия, находящиеся в этом блоке, выполняются параллельно. Работа блока завершается, когда успешно завершается выполнение одного из действий. При этом прежде, чем завершить работу блока, происходит ожидание завершения работы остальных действий. Если все действия в блоке завершают свою работу некорректно (аварийно, принудительно), то и некорректно завершается работа всего блока;
 - 4) «Исключающее Или Параллельный» блок. Все действия, находящиеся в этом блоке, выполняются параллельно. Работа блока завершается, когда успешно завершается выполнение одного из действий. При этом работа выполнения остальных действий завершается принудительно. Если все действия в блоке завершают свою работу некорректно (аварийно, принудительно), то и некорректно завершается работа всего блока;
 - 5) «Случайный» блок. Работа блока начинается с выполнения первого действия в нем. Если выполнение первого действия завершается некорректно (аварийно, принудительно), то выполняется второе действие. И так далее по цепочке. Работа блока завершается, если работа текущего выполняемого действия завершается корректно. Если все действия в блоке завершают свою работу некорректно (аварийно, принудительно), то и некорректно завершается работа всего блока;
 - 6) «Условный» блок. Данный блок содержит два действия и условие. Если условие истинно, то выполняется одно действие. Иначе другое. Успешное завершение работы блока зависит от успешности завершения выбранного действия. Если выбранное действие

завершает свою работу успешно, то и успешно завершается работа блока. Иначе работа блока завершается некорректно;

- 7) «Итеративный» блок. Данный блок идентичен «Последовательному» блоку. Однако он содержит в себе условие остановки выполнения последовательности действий, содержащихся в нем.

CPOLE (Care Plan On-Line) [23] – совместная инициатива Университета Южной Австралии, MCA (Medical Communications Associates) и Департамента Социальных Служб Южной Австралии, направленная на обеспечение врачей гибким инструментом планирования лечения, который учитывает индивидуальные особенности пациента и цели лечения, собирает данные для поддержки принятия клинических решений и включает планирование задач для выполнения плана лечения. Этот программный продукт был испытан и хорошо себя зарекомендовал в проекте SAHealthPlus[24], направленном на создание компьютерной платформы для поддержки пациент-ориентированных процессов лечения различных заболеваний.

ADEPT (Advanced Decision Environment for Process Tasks) [25] является научно-исследовательским проектом, направленным на разработку гибких графических архитектур управления технологическими процессами. С момента старта в 1996 году, было реализовано несколько прототипов систем моделирования и управления технологическими процессами, в том числе и медицинскими. Технология ADEPT используется в качестве основы для системы NematoWork, которая будет описана ниже. ADEPT включает в себя следующие ключевыми особенности:

- спецификация и управление такими временными конструкциями как «завершить действие X за два дня до начала выполнения действия Y»; такие конструкции являются общими в протоколах, описывающих лечебно-диагностические процессы;

- поддержка обработки исключений и узкоспециализированных схематических изменений с помощью операторов, изменяющих формальные пред- и постусловий действий;
- развитие схемы технологического процесса путем применения изменений к уже запущенным экземплярам ТП.
- спецификация и реализация зависимостей между технологическими процессами в распределенных средах; это достигается за счет взаимодействия выражений, поддерживающих синхронизацию параллельно взаимодействующих технологических процессов;
- масштабируемость и балансировка нагрузки серверов с помощью запуска экземпляров и передачи информации об их состоянии между серверами.

HematoWork применяется для управления технологическими процессами в онкологии (в том числе и управлением распределенной терапии рака). Эта технология разработана в Университете Лейпцига и основана на динамических системах правил [26]. Одним из прототипов является AgentWork. Основное внимание в AgentWork уделяется способности поддерживать полуавтоматическую обработку отказов процесса управления, на основе правил Событие-Условие-Действие. AgentWork и, в частности, его теоретические основы подробно описал в своей диссертации Роберт Мюллер [27].

В 2000-х годах в медицинские технологические процессы стали активно внедряться клинические руководства, разработанные на основе технологии доказательной медицины [28]. Клинические руководства представляют собой систематически разрабатываемые рекомендации для помощи врачам и пациентам в принятии решения о выборе методов лечения в конкретной клинической ситуации [29]. Таким образом, клинические руководства приводят к улучшению качества медицинской помощи. В клиническом руководстве описано, что нужно сделать, когда и в какой ситуации во время процесса лечения. Формализация клинического руководства – это процесс

трансформации, в котором текст клинического руководства переводится в эквивалентное, но более формальное представление. Процесс формализации клинических руководств достаточно сложный, длительный и дорогой. В нем задействовано большое количество специалистов из различных областей, включая информатику.

На сегодняшний момент разработано достаточно много методов формального представления клинических руководств. Все разработанные методы используют один из двух подходов к формализации: модель-ориентированный и документ-ориентированный [30]. В первом случае моделируются все процессы, описанные в клиническом руководстве, и объединяются в одну схему лечения. Данный подход используется в AsbruView [31, 32], Protégé [33], Arezzo [34], Tallis [35]. Процесс формализации состоит из трех основных шагов:

- создание компьютерных моделей процессов, описанных в клиническом руководстве (компьютеризированное клиническое руководство);
- тестирование созданного компьютеризированного клинического руководства;
- запуск клинических руководств с настройками под конкретного пациента.

Для выполнения каждого шага в системах реализованы различные модули, графические средства и способы хранения компьютеризированных клинических руководств.

В документ-ориентированном подходе производится разметка текста клинического руководства и строится полу формальная модель размеченной части текста. Чаще всего в качестве языка разметки используется XML, а сами клинические руководства должны быть в формате HTMLили XHTML. Второй подход реализован в следующих системах: GEM Cutter [36], Stepper [37, 38], the

Document Exploration and Linking Tool / Addons (DELT/A) [39, 40], Uruz [41] и др. Кроме формализации, в каждом программном продукте реализованы различные функции для упрощения использования клинических руководств в медицинских процессах. В GEM Cutter реализован функционал по адаптации клинического руководства к возможностям клиники и интеграции в рабочее пространство.

DELT/A позволяет отображать оригинальный текст клинического руководства, его интерпретацию в формате XML и фрагменты, выбранные пользователем. Кроме того, данный программный продукт позволяет задавать ссылки между оригинальным клиническим руководством и целевым представлением. Это дает пользователю возможность посмотреть, где в тексте клинического руководства находится значение, определенное в XML формате, и посмотреть фрагмент текста, откуда это значение было взято. Вторая возможность – это использование макросов. С помощью макроса можно описать собственное отображение часто встречающихся комбинаций XML-тэгов, что может существенно облегчить восприятие отображения XML формата клинического руководства и упростить XML формат.

В России аналогом клинических руководств являются протоколы ведения больных. Более того, в нашей стране культивируется сильный акцент на персонализацию медицинской помощи. Это означает, что все медицинские технологические процессы должны быть персонализированными. Строить персонализированные МТП основываясь только на клинических руководствах достаточно сложно, т.к. в них описываются рекомендации по ведению пациентов без учета их индивидуальных особенностей. Построить персонализированные МТП можно путем анализа технологических процессов лечения конкретных пациентов. Решить эту задачу силами экспертов очень тяжело, поэтому необходим алгоритм, который позволяет автоматически строить обобщенную схему медицинского технологического процесса на

основе прецедентной информации и осуществлять последующую его персонализацию на основе учёта особенностей пациентов.

1.3.2. Методы автоматического построения технологических процессов

Рассмотрим следующую задачу: пусть в базе данных хранится информация об элементах технологического процесса – множество объектов, их состояний, действий, переводящих объекты из одного состояния в другое, назначений и манипуляций, сгруппированная по определенным принципам. Требуется построить общее описание технологического процесса.

В направлении под названием анализ процессов (process mining) решением проблемы синтеза общей модели технологического или бизнес процесса на основе прецедентной информации занимаются достаточно давно. Основополагающие подходы для решения задачи построения технологического процесса на основе анализа логов событий сформулировали R. Agrawal [42], J. Cook и A. Wolf [43], и A. Datta [44]. J. Cook и A. Wolf предложили три подхода синтеза ТП [43]: метод на основе конечных автоматов, метод на основе нейронных сетей и метод на основе теории Марковских дискретных случайных процессов, которые используют статистические, алгоритмические и вероятностные методы соответственно. Они применяли свои подходы к последовательным процессам и вводили специфичные метрики для определения конкурентных процессов (т.е. выполняющихся в одно время). Однако эти методы позволяют извлекать не все типы маршрутов в технологическом процессе.

Подходы для внедрения методов автоматического синтеза ТП на основе лог данных в системы управления потоками работ были предложены R. Agrawal [42] и A. Datta [44]. В качестве математического аппарата для моделирования процессов чаще всего используются сети Петри [45]. Сеть Петри представляет собой направленный граф с двумя типами узлов: позиции (места) и переходы. Позиции соответствуют условиям в процессе, а переходы –

действиям. Состояние сети Петри (или процесса) описывается путем добавления меток в позиции. Динамика сети Петри определяется правилом срабатывания. Переход может быть исполнен (т.е. действие имеет место в процессе) только в том случае, если все позиции (или предусловия в процессе), соединенные с переходом исходящими дугами, имеют метки. После исполнения переход удаляет метки из всех входных позиций и добавляет метки во все исходящие позиции. Кроме того, сеть Петри, используемая для моделирования технологических процессов, имеет только одну начальную и конечную позиции. Такие сети Петри называются сетями потоков работ (workflow nets). Сеть потоков работ будем обозначать WF-сеть. На рисунке 1.1 изображен пример сети потока работ для медицинского технологического процесса сосудистого заболевания. В WF-сети последовательному маршруту соответствует упорядоченная последовательность {позиция, переход, позиция, переход, ..., переход, позиция,..}. Построение итеративного маршрута осуществляется с помощью конструкций «И-разделение» и «И-слияние» [45]. Конструкции «И-разделение» соответствует переход с двумя или более выходными местами, «И-слияние» – переход с двумя и более входными местами. Построение условного маршрута осуществляется с помощью конструкций «ИЛИ-разделение» и «ИЛИ-слияние» [45]. Конструкции «ИЛИ-разделение» соответствует место с двумя или более выходными переходами, «ИЛИ-слияние» – место с двумя или более входными переходами.

На рисунке 1.1 изображены следующие конструкции для построения маршрутов: переходы «регистрация», «прием у кардиолога» соответствуют конструкциям «И-разделение», «Диагноз подтвержден», «Диагноз не подтвержден» и «Принятие решения о хирургическом вмешательстве» – «И-слияние», позиции с4, с5, с8 и с10 – «ИЛИ-разделение», с6, с9 и с 11 – «ИЛИ-слияние».

Одним из первых алгоритмов по автоматическому построению модели технологических процессов с использованием Сетей Петри является α -

алгоритм, разработанный Wil M.P. van der Aalst [46]. На основе этого алгоритма были разработаны многие другие методы и алгоритмы, в которых удалось исправить его недостатки.

Для описания α -алгоритма введем следующее определение.

Определение 1.2 [46]. Лог событий – это множество прецедентов технологических процессов, в котором каждое событие встречается минимум в одном прецеденте. Путь в лог событий – это упорядоченное множество событий (действий).

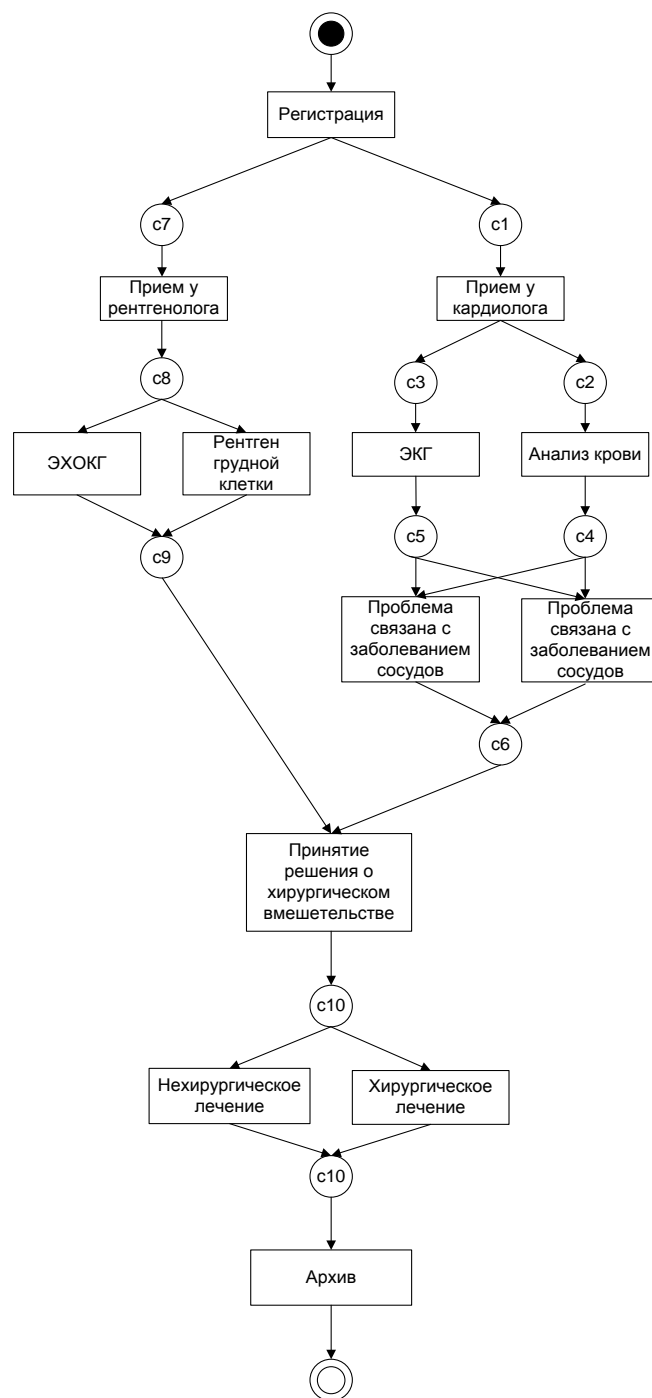


Рисунок 1.1. Пример сети медицинского технологического процесса.

Пусть L – лог событий (лог файл), T – множество действий, σ – «путь» в L , состоящий из упорядоченной последовательности действий из A . На вход α -алгоритму подается лог файл L . На выходе α -алгоритма получается WF-сеть. В процессе обработки лог файла L α -алгоритм строит между всеми действиями следующие бинарные отношения [46, 47]: «следует за», «строго следует за», «параллельно» и «не связано». Отношение «следует за» между действиями a и

$b \in T$ ($a >_L b$) устанавливается в том случае, если существует $\sigma = \langle t_1, t_2, t_3, \dots, t_n \rangle \in L$ и $i=1, \dots, n-1$, такие что $a = t_i$ и $b = t_{i+1}$. Отношение «строго следует за» между двумя действиями a и $b \in T$ ($a \rightarrow b$) устанавливается тогда и только тогда, когда в L имеет место $a >_L b$ и не имеет место $b >_L a$. Если в L имеет место $a >_L b$ и $b >_L a$, то между действиями a и b устанавливается отношение «параллельно» ($a // b$). Если между действиями a и b не установлено ни одно из вышеперечисленных отношений, то они являются несвязанными ($a \# b$). В результате получается следующая матрица:

$$\begin{array}{ccccc}
 & a & b & c & d & e \\
 a & \# & \rightarrow & \rightarrow & \# & \rightarrow \\
 b & \leftarrow & \# & \parallel & \rightarrow & \# \\
 c & \leftarrow & \parallel & \# & \rightarrow & \# \\
 d & \# & \leftarrow & \leftarrow & \# & \leftarrow \\
 e & \leftarrow & \# & \# & \rightarrow & \#
 \end{array} \tag{1}$$

На основе этой матрицы строится WF-сеть как показано на рисунке 1.2. Алгоритм состоит из следующих шагов [46]:

- 1) на основе анализа «путей» в лог файле строится множество переходов

$$T_L = \{t \in T \mid \exists_{\sigma \in L} t \in \sigma\};$$

- 2) создается множество выходных переходов для начальной позиции

$$T_I = \{t \in T \mid \exists_{\sigma \in L} t \in first(\sigma)\};$$

- 3) создается множество входных переходов для конечной позиции

$$T_O = \{t \in T \mid \exists_{\sigma \in L} t \in last(\sigma)\};$$

- 4) создается множество вершин, находящихся между собой в отношении «строго следует за»,

$$\begin{aligned}
 X_L = \{ (A, B) \mid & A \subseteq T_L \wedge A \neq O \wedge B \subseteq T_L \wedge B \neq O \wedge \forall_{a \in A} \forall_{b \in B} a \rightarrow b \wedge \\
 & \wedge \forall_{a_1, a_2 \in A} a_1 \# a_2 \wedge \forall_{b_1, b_2 \in B} b_1 \# b_2 \};
 \end{aligned}$$

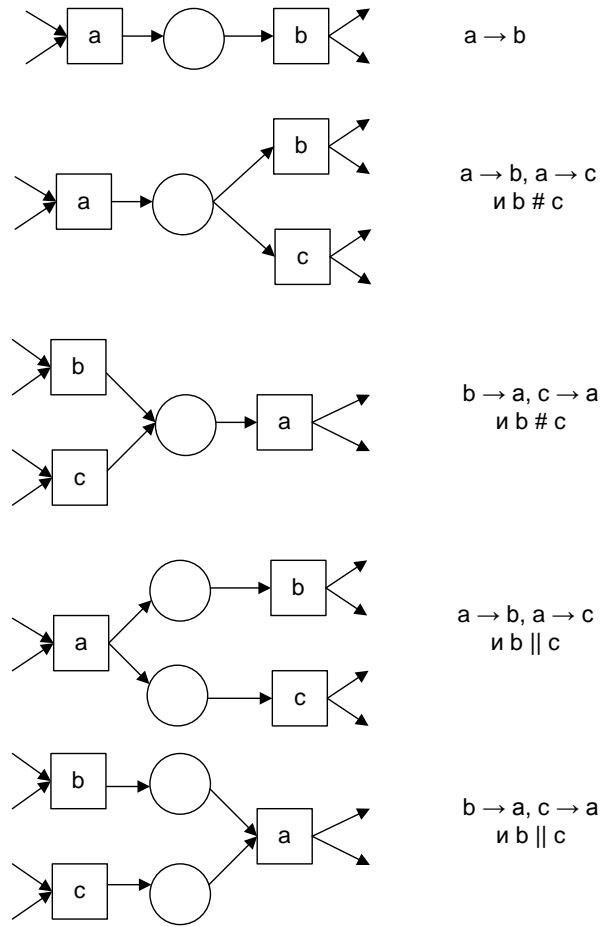


Рисунок 1.2. Маршруты WF-сети, соответствующие отношениям между действиями в лог файле.

5) создается множество $Y_L = \{(A, B) \in X_L \mid \forall_{(A', B') \in X_L} A \subseteq A' \wedge B \subseteq B' \Rightarrow (A, B) = (A', B')\}$;

6) создается множество позиций $P_L = \{p(A, B) \mid (A, B) \in Y_L\} \cup \{i_L, o_L\}$;

7) к созданным позициям добавляются их входные и выходные переходы, т.е. создается множество

$$F_L = \{(a, p(A, B)) \mid (A, B) \in Y_L \wedge a \in A\} \cup \{(p(A, B), b) \mid (A, B) \in Y_L \wedge b \in B\} \cup \{(i_L, t) \mid t \in T_I\} \cup \{(t, o_L) \mid t \in T_O\};$$

8) полученная сеть возвращается алгоритмом, т.е. $\alpha(L) = (P_L, T_L, F_L)$.

Этот алгоритм обладает рядом ограничений и недостатков [48]. α -алгоритм может хорошо извлечь модель ТП только в том случае, если все

экземпляры процесса содержат все проводимые в них действия, сами процессы не прерывались и отсутствуют шумы. Следующим недостатком является невозможность обрабатывать циклы длиной в две задачи и петли. Если встречается цикл длиной в две задачи, например, $a \rightarrow b$ и $b \rightarrow a$, то алгоритм в этой ситуации построит отношение «параллельно» между действиями a и b . Кроме коротких циклов и петель, α -алгоритм не может обрабатывать скрытые действия, принудительный выбор и дублирование действий. Скрытые действия – это действия, которые имеют место быть в WF-сети, но не их нет в лог файле. Проблема обнаружения принудительного выбора заключается в том, что в некоторый момент времени выбор действия, которое необходимо выполнить, зависит от того, какие действия были выполнены ранее. Приведем пример. Пусть имеется лог событий по процессу получения водительских прав, пример которого приведен в таблице 1.1.

Таблица 1.1 –Пример лога событий с принудительным выбором

Идентификатор	Экземпляр процесса
1	Старт, Подать заявку на обучение для получения прав, Пройти обучение управлению автомобилем, Сдать теоретический экзамен, Сдать практический экзамен по управлению автомобилем, Получить результаты, Конец
2	Старт, Подать заявку на обучение для получения прав, Пройти обучение управлению мотоциклом, Сдать теоретический экзамен, Сдать практический экзамен по управлению мотоциклом, Получить результаты, Получить водительское удостоверение, Конец
3	Старт, Подать заявку на обучение для получения прав, Пройти обучение управлению автомобилем, Сдать теоретический экзамен, Сдать практический экзамен по управлению автомобилем, Получить результаты,

	Получить водительское удостоверение, Конец
4	Старт, Подать заявку на обучение для получения прав, Пройти обучение управлению мотоциклом, Сдать теоретический экзамен, Сдать практический экзамен по управлению мотоциклом, Получить результаты, Конец

α -алгоритм построит WF-сеть, изображенную на рисунке 1.3а. Однако, для рассматриваемого примера более корректной является WF-сеть, изображенная на рисунке 1.3б.

Дублирование действий возникает в тех случаях, когда в технологическом процессе действие с одним названием выполняется в разных местах. Проблему принудительного выбора α -алгоритм не может разрешить ввиду того, он анализирует локальную иную информацию о связи двух действий. Обнаружение конструкции принудительного выбора требует более отдаленных отношений между действиями. Дублирование действий не может быть извлечено т.к. α -алгоритм принимает отношение «один к одному» между действиями в лог файлах и их аналогами в WF-сети.

Кроме всех выше перечисленных недостатков, α -алгоритм уязвим к шуму, т.е. к вкраплениям неправильно записанной или просто неверной информации в логе. Это связано с тем, что этот алгоритм не учитывает частоту встречаемости отношений между действиями.

Было разработано несколько модификаций α -алгоритма, позволяющих устранить его недостатки.

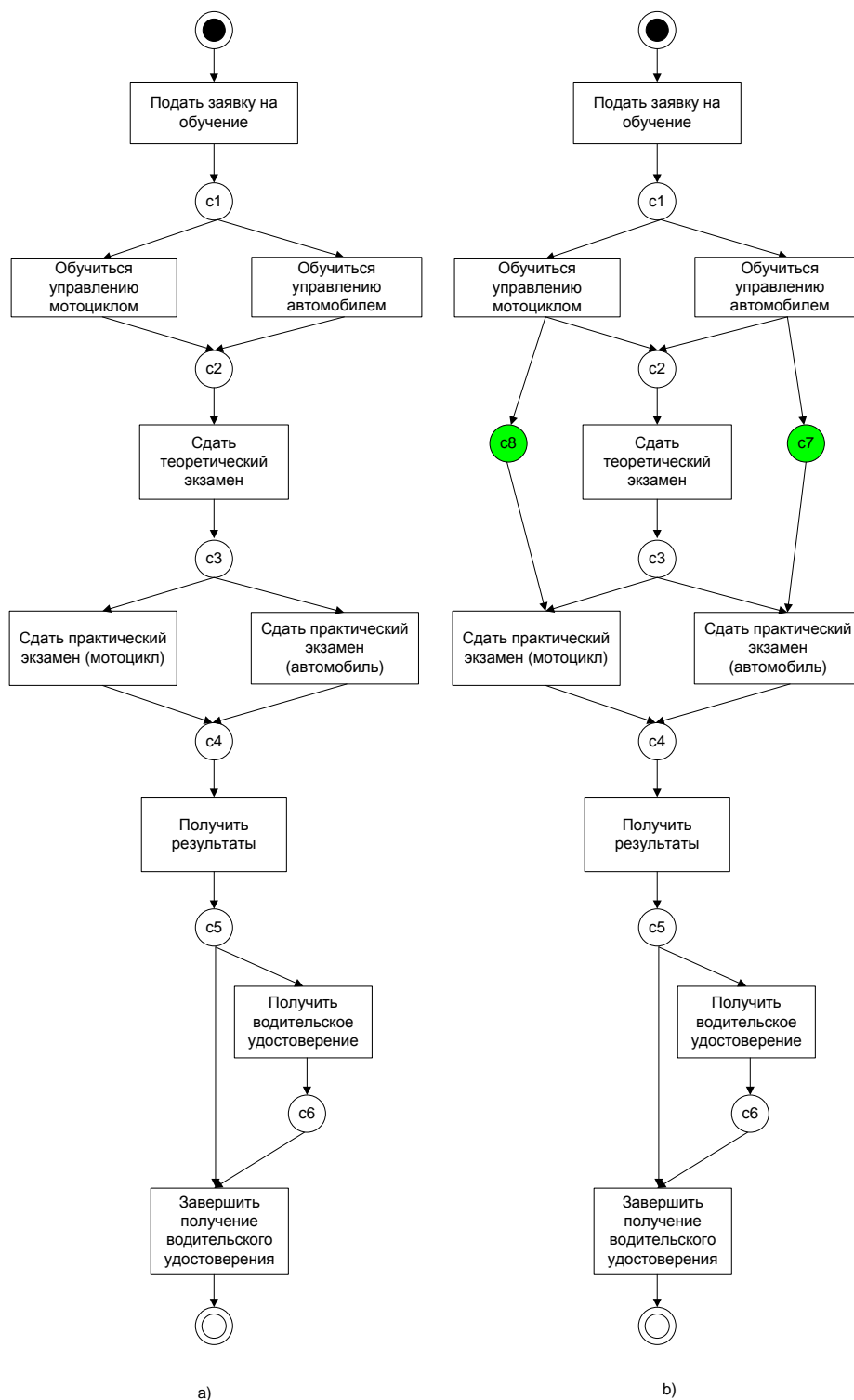


Рисунок 1.3. WF-сети процесса с учетом несвободного выбора и без.

Алгоритмом, позволяющим решить все вышеперечисленные проблемы, является генетический алгоритм, предложенный Alves de Medeiros [80]. Фитнес-функция приспособленности этого алгоритма включает в себя меры точности и полноты, которые направляют алгоритм к подходящим моделям.

Алгоритм работает следующим образом [80, 47]: вначале строится начальная популяция особей. Каждой особи сопоставляется мера соответствия желаемому результату, которая характеризует ее качество. Особь — это возможная модель процесса, а степень соответствия — функция оценки поведения особи в лог файле. Популяции эволюционируют путем выбора наилучших особей и создания новых особей, используя генетические операторы пересечения (crossover — совмещение частей двух и более особей) и мутации (случайная модификация особи). Каждая особь в популяции представлена матрицей причинно-следственных связей, аналогичной (1).

У генетического алгоритма есть свой недостаток, нехарактерный для остальных алгоритмов. Его выполнение занимает очень много времени, что фактически сводит на нет все его преимущества. Большие временные затраты возникают вследствие того, что одной из основных метрик, используемой для вычисления меры соответствия особи логике работы ТП, является процент правильно завершенных частных ТП в логге. Это означает, что алгоритм пропускает все имеющиеся в логге экземпляры ТП, через каждую особь (модель) в популяции. Хотя такой подход позволяет адекватно оценить меру соответствия модели логге, это также вызывает как минимум прямую зависимость между размером логга и временем выполнения алгоритма, что делает нецелесообразным его использование для серьезных задач из реальной жизни.

В таблице 1.2 представлен список методов и алгоритмов построения общей схемы технологического процесса на основе информации, хранящейся в лог файле[49-79].

Таблица 1.2. Обзор методов извлечения технологических процессов




Название	Авторы	Год	Нерешенные проблемы					Методы реализации					
			Шум	Дублирование действий	Скрытые задачи	Конструкции принудительного выбора	Петли	Генетический алгоритм	Метод классификации	Метод кластеризации	Алгоритмический подход	Индуктивное логическое программирование	Вероятностный подход
General DAG	R. Agrwala и др.	1998	+		+						+		
B-F(k, c)-algorithm	A. Datta	1998	+				+				+		
Rnet, Ktail, Mrakov	J. Cook, A. Wolf	1998	+		+		+		+		+		+
Global partial orders	H. Mannila, C. Meek	2000	+				+				+		+
Process Miner	G. Schimm	2002		+			+		+	+			
α/α^+	W.M.P. van der Aalst и др.	2004									+		
InWoLvE—splitpar	J. Herbst, D. Karagiannis	2004	+	+	+						+		+
Multi-phase	B.F. van Dongen,	2005									+		

Miner	W.M.P. van der Aalst												
Workflow Miner	W. Gaaloul и др.	2005									+		
Heuristics Miner	A.J.M.M. Weijters и др.	2006	+		+	+	+				+		
DWS Mining	G. Greco и др.	2006	+			+			+	+			
Rule-based approach	L. Maruster и др.	2006	+						+				
Rule-based approach	H. Ferreira, D. Ferreira	2006										+	
Genetic Miner	A.K. Alves de Medeiros и др.	2007	+		+	+	+	+					
DT Genetic Miner	A.K. Alves de Medeiros и др.	2007	+	+	+	+	+	+					
Fuzzy Miner	C.W. Gunther, W.M.P. van der Aalst	2007	+				+			+	+		
α^{++}	L. Wen и др.	2007				+					+		

DecMiner	E. Lamman и др.	2007										+	
AWS Mining	G. Greco и др.	2008	+				+			+			
AGNEsMiner	S. Goedertier и др.	2009	+	+	+	+	+		+			+	
β (orTshingua α)	L. Wen и др.	2009					+				+		
Enhanced WfMiner	F. Folino и др.	2009	+	+	+	+	+				+		
EM-approach	D.R. Ferreira, D. Gilblad	2009	+										+
ILP Miner (Parikh)	J.M.E.M. van der Werf и др.	2009			+						+		
FSM Miner/Petrify	W.M.P. van der Aalst и др.	2010	+	+	+	+	+				+		
FSM Miner/Genet	J. Carmona и др.	2010					+				+		

Специалисты фирмы IBM разработали алгоритм построения общего описания технологических процессов на основе жизненных циклов объектов [81-85]. Технологический процесс содержит набор объектов, характеризующих обмен данными между действиями. С каждым объектом связано множество состояний, описывающих его статус в определенный момент времени. Действия процесса переводят объекты из одного состояния в другое.

Определение 1.3 (J.M.Kuster, K. Ryndina, H. Gall[86]). Пусть дан объект типа o , тогда жизненным циклом объекта называется пятерка $OLC_o = \langle S, s_\alpha, S_\Omega, \Sigma, \delta \rangle$, где S - конечное число состояний, $s_\alpha \in S$ - начальное состояние, $S_\Omega \subseteq S$ - множество конечных состояний, Σ - множество событий (действий), $\delta: S \times \Sigma \rightarrow S$.

На рисунке 1.4 изображены примеры жизненных циклов объектов «Заявка на предоставление кредита» и «Платеж по кредиту». Стрелками обозначены действия, которые переводят объекты в различные состояния, значком  - начальное состояние объекта,  - состояния объекта,  - конечные состояния объекта.

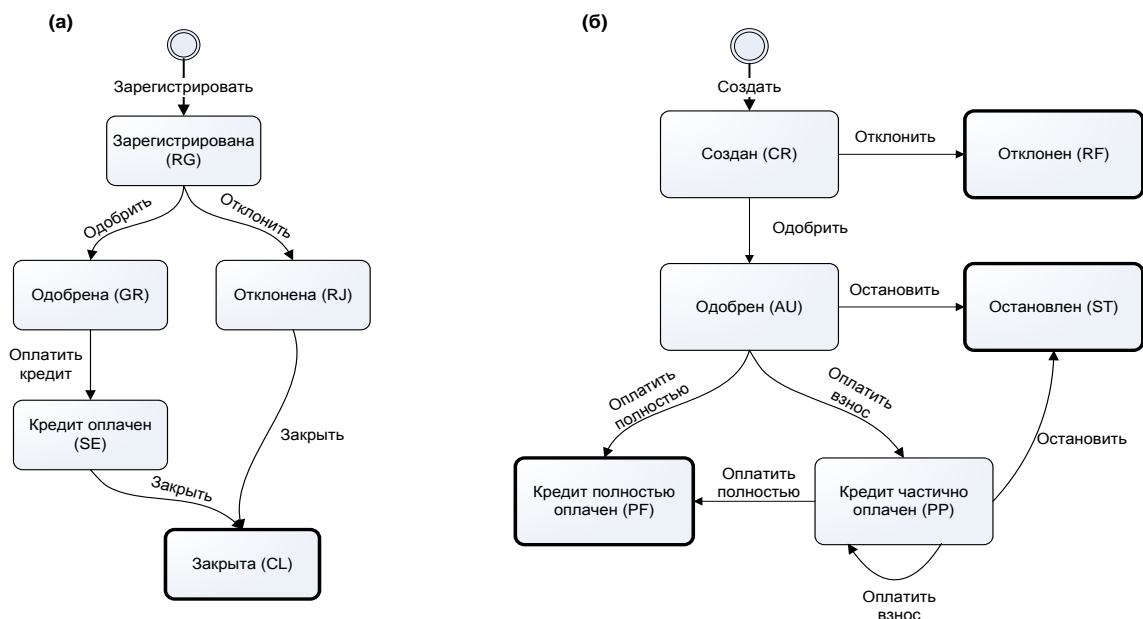


Рисунок 1.4. Примеры жизненных циклов объектов «Заявка на предоставление кредита» (а), «Платеж» (б)

На рисунке 1.5 изображена упрощенная схема технологического процесса, соответствующего жизненному циклу, изображенному на рисунке 1.10 (а).

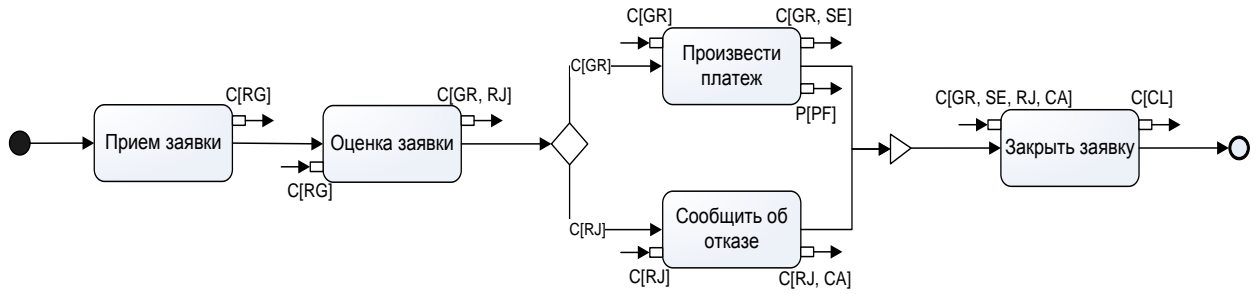


Рисунок 1.5. Схема технологического процесса по обработке заявки на получение кредита

Практически во всех технологических процессах присутствует несколько объектов различных типов, которые каким-то образом взаимодействуют друг с другом. Каждый из этих объектов имеет свой жизненный цикл. Так, в рассматриваемом примере присутствуют два объекта «Заявка на предоставление кредита» и «Платеж по кредиту». При этом известно что, объект «Платеж по кредиту» создается после того как заявка на предоставление кредита принимает статус «Одобрена». В тех случаях, когда в технологическом процессе задействованы несколько объектов различных типов, перед созданием общей схемы технологического процесса строится композиция жизненных циклов объектов.

Определение 1.4 (J.M. Kuster, K. Ryndina, H. Gall[86]). Пусть даны жизненные циклы

$$OLC_{o1} = \langle S_1, s_{\alpha1}, S_{\Omega_1}, \Sigma_1, \delta_1 \rangle \quad \text{и}$$

$$OLC_{o2} = \langle S_2, s_{\alpha2}, S_{\Omega_2}, \Sigma_2, \delta_2 \rangle,$$

композицией жизненных циклов называется следующая конструкция

$$OLC = \langle S_1 \times S_2, (s_{\alpha1}, s_{\alpha2}), S_{\Omega_1} \times S_{\Omega_2}, \Sigma_1 \cup \Sigma_2, \delta \rangle,$$

где

$$\delta((s_1, s_2), e) = \begin{cases} \delta_1(s_1, e) \times \delta_2(s_2, e), & \text{если } e \in \Sigma_1 \cap \Sigma_2 \\ \delta_1(s_1, e) \times \{s_2\}, & \text{если } e \in \Sigma_1 \setminus \Sigma_2 \\ \{s_1\} \times \delta_2(s_2, e), & \text{если } e \in \Sigma_2 \setminus \Sigma_1 \end{cases}$$

На рисунке 1.6 изображена композиция жизненных циклов объектов, изображенных на рисунке 1.4.

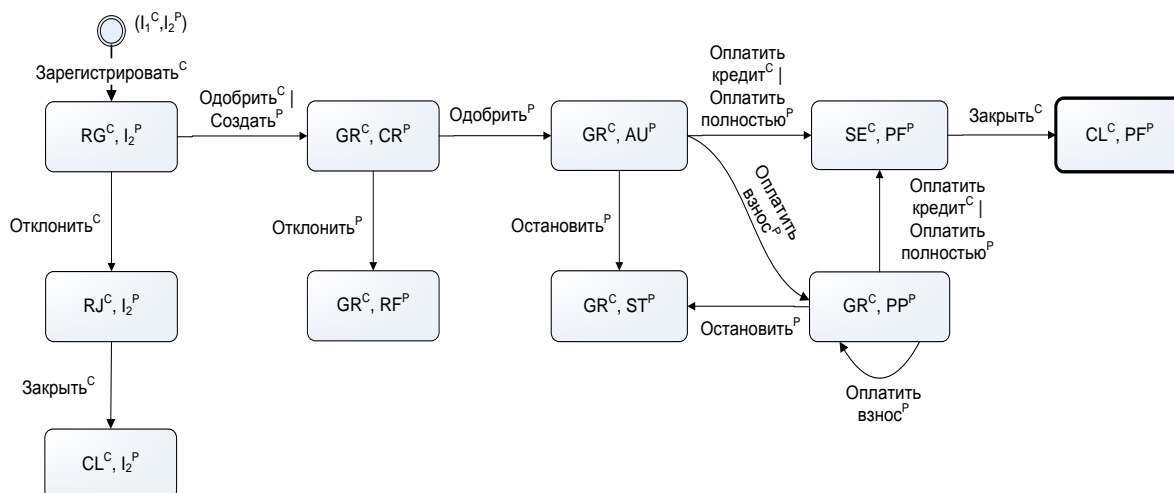


Рисунок 1.6. Композиция жизненных циклов объектов «Заявка на предоставление кредита» и «Платеж по кредиту»

Для автоматического создания модели технологического процесса сотрудниками IBM был предложен следующий алгоритм [86], на вход которому подается либо жизненный цикл объекта, либо композиция жизненных циклов:

- 1) на основании входной информации строится множество действий технологического процесса;
- 2) строится порядок выполнения действий в технологическом процессе;
- 3) из действий строятся фрагменты технологического процесса, в которых действия соединяются с точками ветвления или объединения различных ветвей;
- 4) полученные фрагменты объединяются в общую схему технологического процесса.

Пусть дан жизненный цикл $OLC = \langle S, s_\alpha, S_\Omega, \Sigma, \delta \rangle$, который является композицией жизненных циклов объектов типов o_1, \dots, o_n . Множество действий строится путем обхода всех дуг жизненного цикла OLC . Каждому действию приписываются входные и выходные состояния объектов и их типы, которые подаются на вход и получаются на выходе. Если встречается

синхронизирующее событие, то строится действие с аналогичным именем. Алгоритм построения множества действий на основе композиции жизненных циклов подробно описан в [86].

Порядок выполнения действий строится на основе следующего определения [86]:

Определение 1.5 (J.M. Kuster, K. Ryndina, H. Gall[86]). Даны два действия a_1 и a_2 ; выходом действия a_1 являются объекты типов o_{11}, \dots, o_{1k} , входом действия a_2 – объекты типов o_{21}, \dots, o_{2m} , тогда действие a_2 выполняется после a_1 тогда и только тогда, когда для всех объектов, типы которых $o \in \{o_{11}, \dots, o_{1k}\} \cap \{o_{21}, \dots, o_{2m}\}$, пересечение выходного состояния действия a_1 и входного состояния действия a_2 не является пустым множеством (т.е. $outstate_o(a_1) \cap instate_o(a_2) \neq \emptyset$).

На третьем шаге алгоритма производится обход всех действий технологического процесса, строятся его фрагменты согласно правилам, изображенным на рисунке 1.7. Для каждого действия a , если число действий на предыдущем и следующем этапе работы технологического процесса равно 0 или 1, то строится фрагмент ТП, состоящий только из действия a (rule 3.1). Если число действий на предыдущем и следующем этапе работы технологического процесса больше 1, то строится фрагмент ТП, состоящий из действия a , соединяющего узла и ветвления ТП (rule 3.2). Если число действий на предыдущем этапе работы технологического процесса равно 0 или 1, а на следующем больше 1, то строится фрагмент ТП, состоящий из действия a и ветвления ТП (rule 3.3). Если число действий на предыдущем этапе работы технологического процесса больше 1, а на следующем равно 0 или 1, то строится фрагмент ТП, состоящий из действия a и соединяющего узла (rule 3.4). Если существует только одно действие такое, что на предыдущем шаге выполнения ТП нет никаких действий, то строится фрагмент, содержащий узел, обозначающий начало технологического процесса (начальный узел). Если количество действий таких, что на

предыдущем шаге выполнения ТП нет никаких действий, больше 1, то строится фрагмент, начальный узел и точку ветвления ТП (rule 3.5). Если существует только одно действие такое, что пересечение множества его выходных состояний и множества коечных состояний не пусто, то строится фрагмент ТП, содержащий завершающий узел ТП. Если таких действий несколько, то строится фрагмент ТП, содержащий завершающий и соединяющий узлы ТП (rule 3.6).

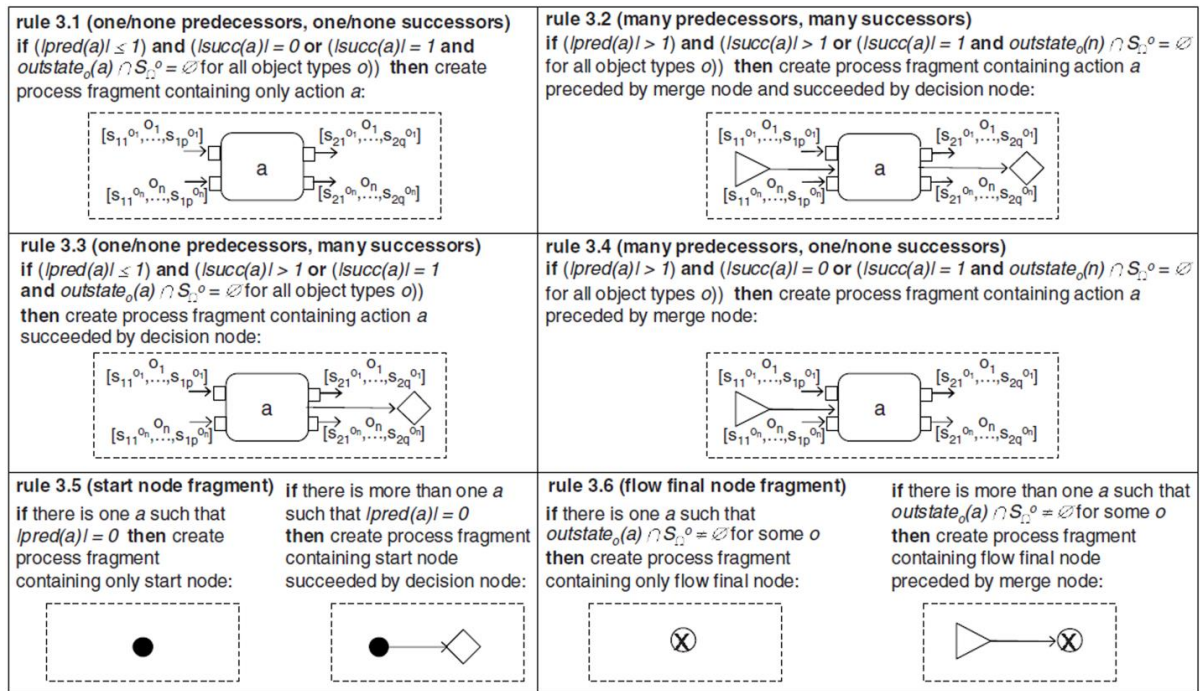


Рисунок 1.7. Правила для построения фрагментов технологического процесса

На последнем шаге работы алгоритма производится соединение всех фрагментов в общую схему технологического процесса согласно правилам, представленным на рисунке 1.8. Если для действия a число действий на предыдущем этапе выполнения ТП равно 0, то соединяются фрагменты, содержащие начальный узел и действие a . Если для действия a число действий на последующем этапе выполнения ТП равно 0 или пересечение множества выходных состояний a и множества конечных состояний не пусто, то соединяются фрагменты, содержащие действие a и завершающий узел. Если для действия a число действий на последующем этапе выполнения ТП

больше 0, то производится соединение фрагментов, содержащих действие **a** и следующих за ним действий. Последнее более подробно изображено на рисунке 1.9.

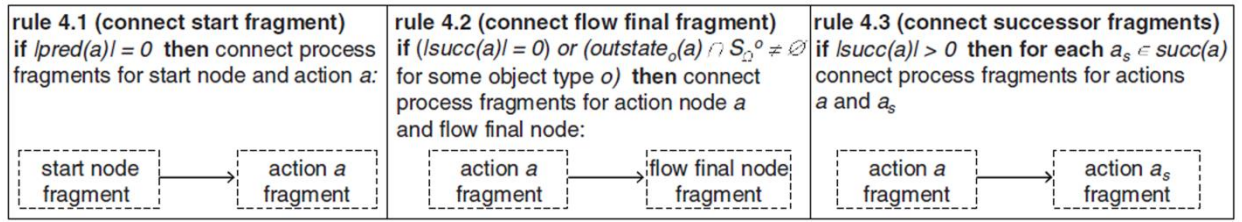


Рисунок 1.8. Правила для соединения фрагментов технологического процесса

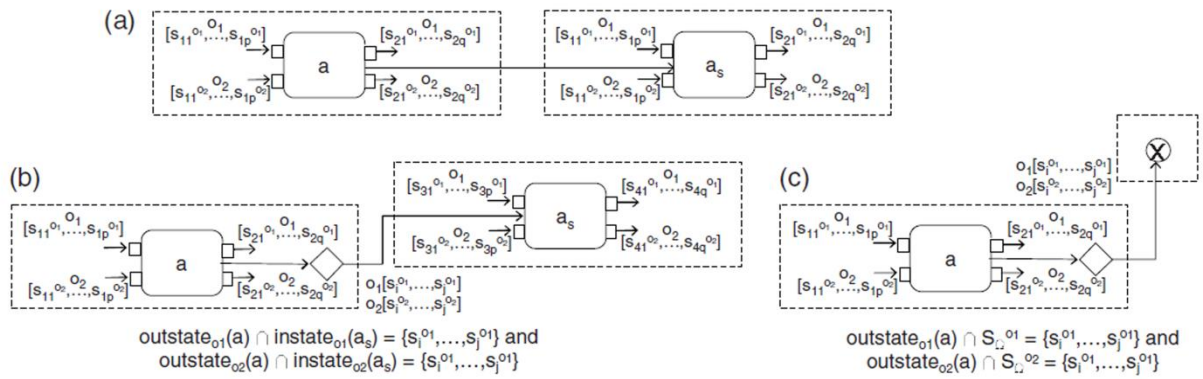


Рисунок 1.9. Соединение фрагментов технологического процесса

Соединение фрагментов происходит следующим образом. Если имеются два фрагмента технологического процесса Фрагмент 1 и Фрагмент 2, содержащие только действия **a** и **a_s**, выполняемые последовательно друг за другом, то эти фрагменты соединяются друг с другом (рисунок 1.9(a)). Если Фрагмент 1 кроме действия **a** содержит узел ветвления, то соединение фрагментов происходит как показано на рисунке 1.9(b). Из рисунка 1.9(b) видно что, действие **a_s** выполняется после **a** если состояния объектов типов **o₁** и **o₂** после выполнения действия **a** принадлежат множествам состояний $outstate_{o_1}(a) \cap instate_{o_1}(a_s)$ и $outstate_{o_2}(a) \cap instate_{o_2}(a_s)$ соответственно. На рисунке 1.9(c) изображен способ соединения фрагментов, содержащих действие с узлом ветвления и завершающий узел, и условия завершения выполнения технологического процесса.

Результат применения алгоритма автоматической генерации технологических процессов к жизненному циклу, изображенному на рисунке 1.6, показан на рисунке 1.10.

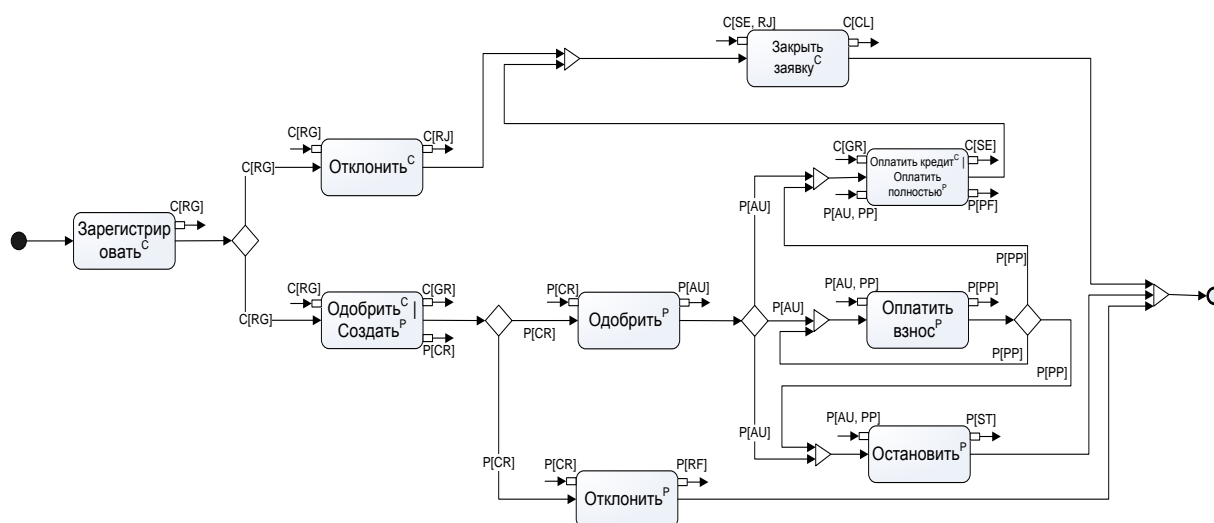


Рисунок 1.10. Автоматически построенная модель технологического процесс по получению и оплате кредита

Однако пока не удалось найти публикаций о результатах применения данного подхода к более сложным технологическим процессам.

К недостаткам этого подхода можно отнести следующее:

- 1) построением жизненных циклов объектов занимаются специально обученные специалисты (инженеры по знаниям), которые обязаны работать с экспертами для получения информации об объектах. Отсюда возникают проблемы приобретения знаний от экспертов;
- 2) если в технологическом процессе задействовано достаточно большое число объектов различных типов, то инженер по знаниям обязан держать их все в голове.

1.3.3. Применение компьютерных технологий для поддержки технологических процессов в психологии

Современное развитие информационных технологий позволяет разрабатывать программные средства, способные брать на себя функции, традиционно считавшиеся прерогативой интеллектуальной деятельности. На

сегодняшний день в психологии выделяют следующие основные типы прикладных интеллектуальных систем [17]:

- 1) интеллектуальные информационно-поисковые системы, которые позволяют обрабатывать и накапливать огромное количество информации из различных источников;
- 2) экспертные системы, которые способны аккумулировать профессиональные знания квалифицированных экспертов и служат полезным инструментом в помощь, как опытному психологу, так и начинающему специалисту;
- 3) обучающие системы, способные давать обоснованные, методически эффективные для обучения знания с адаптивной степенью детализации по рассматриваемым диагностическим решениям.

Наибольшее распространение информационные технологии получили в психодиагностике ввиду того, что многие процессы и методики в данной области формализуются намного проще, чем, например, в психологическом консультировании. Все они направлены на решение следующих ключевых задач в области компьютерной психодиагностики [17]:

- 1) конструирование психодиагностических методик в рамках традиционной психометрической парадигмы и психосемантического подхода на основе субъективной парадигмы анализа данных;
- 2) разработка психодиагностических экспертных систем для прогноза поведения в рамках нетрадиционных подходов, например, опирающихся на внешние критерии с использованием стохастического подхода;
- 3) создание компьютерных психодиагностических методик, использующих компьютер в качестве организатора стимульного материала, т.е. реализация психофизических тестов, систем

адаптивного, игрового, дистанционного и мультимедийного тестирования;

- 4) разработка на основе инженерии знаний компьютерных интерпретаторов результатов тестирований, моделей прогноза;
- 5) разработка гибридных систем для психологического сопровождения конкретных видов деятельности (например, кадрового отбора, обучения и т.д.);
- 6) создание оболочек-конструкторов для компьютеризирования различных компонентов, важных в практической работе психологов (методик, профилей, интерпретаторов и т.д.).

Для решения некоторых из перечисленных задач, стали применяться экспертные системы. В США разрабатывались экспертные системы для выявления и оценки различных нарушений, препятствующих успешному школьному образованию (например, DYSLEXPART [87] и др. [88], [89]). Одним из недавних примеров такого типа систем является система FEARDEX (Fear Diagnostic Expert System), диагностирующая фобии[90]. В нашей стране первая система по психодиагностике АВТАНТЕСТ была создана Т.А. Гавриловой ещё в 1984 году для интерпретации данных по опроснику Р. Кэттелла и выдаче на естественном языке связных непротиворечивых психологических заключений для кадровой службы академий ВМФ СССР [91]. В последнее десятилетие число таких работ растёт, есть примеры удачных решений в области профориентации (например, система ПРОФЭКСПЕРТ [92]), диагностики проблем развития (ЭС индивидуального сопровождения развития "Лонгитюд" [93]).

До недавнего времени развитие в этой области являлось преимущественно экстенсивным. Многие создаваемые в мире системы в области психодиагностики предназначены для вынесения диагноза специалистом, однако сам принцип работы с системой – опрос пользователя и вынесение системой суждения о случае по фиксированным показателям – остаётся неизменным. Примером может служить размещенный на сайте

австралийской рекрутинговой компании PsychPress интерфейс профессиональной ЭС. Предназначенная для сертифицированных специалистов, система позволяет выявлять степень выраженности или отсутствие признаков психических расстройств и личностных нарушений, внесенных в DSM-IV[94]. Всё это скорее напоминает компьютеризированный гипертекстовый учебник, чем процесс совместного, т.е. специалиста и системы, психодиагностического исследования.

В России примером активно развивающейся и внедряемой многофункциональной (гибридной) компьютерной диагностической системы является комплекс Psychometric Expert [95]. Данный комплекс позволяет проводить обследование сотрудников организации и включает в себя следующие системы и модули:

- 1) система планирования и проведения психологической диагностики личности и группы, позволяющая вести учет, психологический отбор, оценку и сопровождение персонала организации, а также различных групп людей (сопровождение клиентов, пациентов, учащихся, осужденных и т.п.);
- 2) библиотека психодиагностических методик с возможностью печати стимульного материала, бланков, накладных ключей и т.д.;
- 3) электронный учебник по экспериментальной психологии, психодиагностике и математической статистике;
- 4) среда разработки баз данных (таблиц, запросов, форм, отчетов, любых документов), направленных на чет различных сведений о респондентах;
- 5) среда разработки и совершенствования экспериментальных методов исследования личности и группы, в частности, психодиагностических тестов;
- 6) среда управления компетенциями, разработки интерпретаторов, моделей, профессиограмм, критериев оценки данных;

- 7) система статистического анализа данных;
- 8) среда подключения других диагностических приложений, имеющих возможность работы внутри комплекса Psychometric Expert, обеспечивая более полное и качественное сопровождение персонала организации;
- 9) среда разработки справочных систем (в том числе – мультимедийных);
- 10) система контроля исполнения обязательств. Данный модуль позволяет учитывать не только психологические параметры личности, но также и другие данные сотрудников, такие как выполненная в срок работа, количество опозданий и т.п.

В настоящий момент разработка системы идет в направлении создания многофункциональной «открытой» прецедентной экспертной системы, которая позволяет решать широкий круг задач психологического сопровождения организаций в различных сферах.

Кроме того, данная система ориентирована на психологическое сопровождение профессиональной деятельности и образовательного процесса, обеспечение проведения психологической экспертизы и консультирования. Комплекс Psychometric Expert может быть использован как в рамках поддержки организации, так и в рамках научно-исследовательской деятельности психологов.

1.4. Выводы.

Анализ существующих систем и методов моделирования и поддержки технологических процессов в области медицины и психологии показал, что они не способны учитывать все особенности технологических процессов в этих областях для того, чтобы ориентировать сами процессы на конкретного пациента (или клиента в психологии). При разработке систем в области медицины за основу берутся клинические руководства, стандарты и планы лечения. Это ориентирует построенные модели на лечение заболеваний, а не

пациентов (клиентов). Далеко не все реальные медицинские процессы совпадают с тем, что написано в клинических руководствах и стандартах. В таблице 1.4.1 указано, какими из основных требований, предъявляемых к системам поддержки технологических процессов в области медицины и психологии, обладают современные системы и система, содержащая методы и алгоритмы, предложенные в настоящей работе.

Таблица 1.4.1 – сравнение свойств систем поддержки МТП

	Системы моделирования и поддержки МТП [3 -14, 18 - 41]	Системы автоматического построения МТП[42-86]	Система поддержки МТП (ИСА РАН)	МИС (Interin)
Формальное описание МТП	+	+	+	-
Средства моделирования МТП	+	+	+	-
Автоматическое построение модели МТП	-	+	+	-
Разбиение МТП на этапы и их поддержка	+	-	+	+
Персонализация МТП	+/-	-	+	-
Средства построения МТП без привлечения инженера по знаниям	-	-	+	-
Средства оценки качества МТП	+	-	+	-
Интерактивность	+	-	+	+
Историчность	+	-	+	+

Из таблицы видно, что система, включающая в себя методы и алгоритмы, предложенные в настоящей работе, удовлетворяет всем основным требованиям, предъявляемым к современным системам поддержки технологических процессов в области медицины и психологии.

Следует отметить то, что большинство разработок в области психологии относится к психодиагностике, в виду того, что многие диагностические методики можно компьютеризировать. Однако другие процессы (психологическое консультирование и др.) не структурированы и внедрение информационных технологий в них сильно затруднено.

В целом можно сказать, что на основе современных информационных технологий в наше время разрабатываются различные инструменты, обеспечивающие профессиональный рост практического психолога. Однако ни один из развиваемых подходов не позволяет решать проблему комплексно, обеспечивая как интерактивность, так и полноту необходимых ему знаний в области диагностики и консультирования. Однако до сих пор не решены следующие задачи:

1. Не разработаны методы автоматической персонализации медицинских технологических процессов.
2. Не учитываются особенности большого числа пациентов.
3. Не разработаны методы динамической адаптации медицинских технологических процессов к особенностям пациентов.

Для повышения уровня и качества систем поддержки технологических процессов в слабоструктурированных областях следует разработать интеллектуальную технологию поддержки ТП, основанную на моделировании не самих процессов, а рассуждений специалистов в каждой конкретной ситуации. В процессе поддержки технологического процесса система, построенная на основе интеллектуальной технологии определяет наиболее рациональные действия специалиста в сложившейся ситуации. Обучение системы происходит на основе работы с экспертами, прецедентами и другими источниками знаний (это могут быть клинические руководства для медицины, последние достижения теоретической медицины, психологии и т.д.).

Для разработки данной технологии необходимо решить следующие задачи:

- 1) построение общей формализованной модели ТП, необходимой для автоматизации обработки информации;
- 2) разработка алгоритмов извлечения персонализированных технологических процессов из медицинских данных для обучения

интеллектуальной системы поддержки ТП (ИСП ТП) на основе прецедентов;

- 3) построение архитектур интеллектуальных систем поддержки ТП медицины и психологии;

Глава 2. Модели технологических процессов

В этой главе рассматривается модель технологического процесса точки зрения операторной теории, предложенной Г.С. Осиповым и Г.И. Назаренко. Сформулировано определение последовательного применения n операторов из этой теории и доказаны утверждения ассоциативности и коммутативности последовательного применения операторов. Описаны типы маршрутов в технологическом процессе. Предложено графическое представление моделей технологических процессов.

2.1. Модель технологического процесса

Технологический процесс (ТП) представляет собой взаимосвязанный набор мероприятий, который потребляет ресурсы и выдает результат.

Медицинский технологический процесс – это система взаимосвязанных необходимых и достаточных научно обоснованных лечебно-диагностических мероприятий, выполнение которых позволяет наиболее рациональным образом провести лечение и обеспечить достижение максимального соответствия научно прогнозируемых результатов реальным при минимизации затрат [96].

Что касается технологических процессов в области практической психологии, то они направлены на обеспечение психологического здоровья личности в условиях социокультурной среды. Психологи не лечат заболевания; они оказывают психологическую помощь конкретному человеку в зависимости от его личностных особенностей и проблемы, во время обращения к специалисту. Таким образом, для оказания квалифицированной помощи психологу необходимо решить следующие задачи:

- выявить личностные особенности клиента;
- определить характер и вид конкретной психологической помощи, адекватной личностным особенностям клиента и соответствующей его реальным условиям жизни.

На основании выше изложенного можно сказать, что технологический процесс в области психологии – это набор взаимосвязанных и научно обоснованных мероприятий, направленных на установление личностных особенностей человека (клиента) и определение характера и вида психологической помощи.

Модель технологического процесса – это формализованное (математическое, графическое, табличное, текстовое, символьное и т.п.) описание ТП.

Моделью технологического процесса является следующая семерка $\mathbf{M} = \langle \mathbf{I}, \mathbf{CON}, \mathbf{IMP}, \mathbf{O}, \mathbf{CON}^I, \mathbf{IMP}^I, \mathbf{O}^I \rangle [1, 104]$, где

- \mathbf{I} – множество с заданным на нём отношением линейного порядка \leq (например, множество натуральных чисел или временная шкала);
- \mathbf{CON} – множество ограничений на значения признаков, характеризующих состояния больных;
- \mathbf{IMP} – множество императивов;
- \mathbf{O} – множество операторов;
- \mathbf{CON}^I – отображение, которое каждому элементу из множества \mathbf{I} ставит в соответствии элемент из множества \mathbf{CON} , т.е. $\mathbf{CON}^I: \mathbf{I} \rightarrow \mathbf{CON}$;
- \mathbf{IMP}^I – отображение, которое каждому элементу из множества \mathbf{I} ставит в соответствии элемент из множества \mathbf{IMP} , т.е. $\mathbf{IMP}^I: \mathbf{I} \rightarrow \mathbf{IMP}$;
- \mathbf{O}^I – отображение, которое декартову произведению $\mathbf{I} \times \mathbf{I}$ ставит в соответствии элемент из множества \mathbf{O} , т.е. $\mathbf{O}^I: \mathbf{Int} \rightarrow \mathbf{O}$, где $\mathbf{Int} \subseteq \mathbf{I} \times \mathbf{I}$ такие, что,
 $\forall t \in \mathbf{I} \mathbf{CON}^I(t) = \mathbf{Con}(t)$, где, $\mathbf{Con}(t)$ – элемент \mathbf{CON} ,
 $\forall t \in \mathbf{I} \mathbf{IMP}^I(t) = \mathbf{Imp}(t)$, где $\mathbf{Imp}(t)$ – элемент \mathbf{IMP} ,

$\forall (t_1, t_2) \in \text{Int } O^I(t_1, t_2) = o(t_1, t_2, T_t, V_t, \text{Loc}_t)$, где $o(t_1, t_2, T_t, V_t, \text{Loc}_t)$ – элемент O .

В медицинских технологических процессах в качестве ограничений выступают требования на продолжительность этапов МТП, наиболее важные показатели, требующие объективного исследования, критериальное подтверждение диагноза (если таковое существует), критерии перевода в другое отделение, критерии смены терапии, критерии выписки из стационара, условиями применения лечебного мероприятия и др.

К императивам относятся указания по сбору данных из всех допустимых источников, о проведении исследований и о выборе специалистов.

Каждый оператор $o \in O$ имеет следующий вид:

$$o = o(t_1, t_2, T, V, Ex),$$

где t_1, t_2 – время начала и окончания выполнения оператора, T – периодичность, Ex – отметка о том, что оператор обязательно должен быть выполнен до конца, или его выполнение можно прервать, V – объем мероприятий. Параметры t_1 и t_2 имеют интервальные значения, которые обозначают минимальное и максимальное время начала выполнения оператора и минимальное и максимальное время окончания выполнения оператора. Параметр Ex говорит о том, что далеко не все лечебные мероприятия обязаны быть завершены. Например, выполнение МРТ может быть прервано в случае, если у пациента выявится скрытая клаустрофобия и любые другие особенности, не позволяющие дальнейшее выполнение данного лечебного мероприятия.

Если ограничения и императивы описывают состояния, то операторы изменяют состояния, воздействуя на них. Эффект воздействия операторов заключается:

- а) в изменении значений некоторых признаков (как количественных, так и качественных);

- б) в появлении новых и исчезновении старых признаков;
- с) в появлении новых и исчезновении старых патологических процессов [103].

Изменения значений признаков производятся с помощью вычисления некоторых функций (присвоение соответствующей переменной нового значения) или, если значения признаков носят неколичественный характер, путем удаления старого признака и добавлении нового. Если параметры состояния являются не признаками, а иными концептуальными элементами картины мира, например, патологическими процессами [1], то надо добавить в новое состояние соответствующий концептуальный элемент – если он возникает в результате применения оператора, либо удалить – если он исчезает.

Таким образом, каждый оператор $o(t_1, t_2, T, V, Ex)$ имеет вид [1]:

$$\left\langle \begin{array}{c} \Phi \\ P_1 \\ P_2 \\ Z_1 \\ Z_2 \end{array} \right\rangle ,$$

где Φ - семейство функций времени из $F = \{f_1, f_2, \dots, f_n\}$, действующих из некоторых декартовых произведений областей $X_{i1}^T, X_{i2}^T, \dots, X_{ik}^T$ в области X_j^T – множества значений признаков; P_1 – множество новых появляющихся признаков, задаваемых экстенциональным образом (множествами значений); P_2 – множество исчезающих признаков, задаваемых экстенциональным образом; Z_1 – множество новых патологических процессов; Z_2 – множество исчезающих патологических процессов. При этом, пересечения множеств P_1 и P_2 и Z_1 и Z_2 пустые.

Определение 2.1 (Г.С. Осипов, Г.И. Назаренко [1]). Оператор o будем называть применимым к состоянию $State(t)$, если его условие применимости C таково, что $C \subseteq State(t)$.

Пусть даны два оператора:

$$o_1 = \left\langle \begin{array}{c} \Phi^1 \\ P_1^1 \\ P_2^1 \\ Z_1^1 \\ Z_2^1 \end{array} \right\rangle, o_2 = \left\langle \begin{array}{c} \Phi^2 \\ P_1^2 \\ P_2^2 \\ Z_1^2 \\ Z_2^2 \end{array} \right\rangle$$

C_1 – условие применимости оператора o_1 , C_2 – условие применимости оператора o_2 .

Определение 2.2.(Г.С. Осипов, Г.И. Назаренко [1]). Последовательным применением операторов o_1 и o_2 называется оператор

$$o_3 = o_1 o_2 = \left\langle \begin{array}{c} \Phi^3 \\ P_1^3 \\ P_2^3 \\ Z_1^3 \\ Z_2^3 \end{array} \right\rangle,$$

такой, что $\Phi^3 = 1 \cup \Phi^1 \cup \Phi^2 \cup \Phi^2 \bullet \Phi^1$, где 1 – единичный оператор, Φ^1 и Φ^2 – компоненты операторов o_1 и o_2 соответственно, $\Phi^2 \bullet \Phi^1$ – композиция этих компонент, $P_1^3 = P_1^1 \cup P_1^2$, $P_2^3 = P_2^1 \cup P_2^2$, $Z_1^3 = Z_1^1 \cup Z_1^2$, $Z_2^3 = Z_2^1 \cup Z_2^2$.

Если к состоянию $State(t_1) = X \cup P \cup Z$ применить оператор o_1 , то получим состояние $State(t_2) = X \cup P \cup Z \cup \Phi^1(X) \cup P_1^1(t_2) \cup Z_1^1(t_2) \setminus (P_2^1(t_2) \cup Z_2^1(t_2))$. Далее, если к полученному состоянию получить оператор o_2 , то получим состояние

$$State(t_3) = X \cup P \cup Z \cup \Phi^1(X) \cup P_1^1(t_2) \cup Z_1^1(t_2) \setminus (P_2^1(t_2) \cup Z_2^1(t_2)) \cup \Phi^2(X \cup \Phi^1(X)) \cup P_1^2(t_3) \cup Z_1^2(t_3) \setminus (P_2^2(t_3) \cup Z_2^2(t_3)).$$

Если в состоянии $State(t_3)$ выделить все множества, относящиеся к компонентам Φ^1 и Φ^2 , то получим следующее выражение $X \cup \Phi^1(X) \cup \Phi^2(X \cup \Phi^1(X))$ или $\Phi^3 = X \cup \Phi^1(X) \cup \Phi^2(X) \cup \Phi^2(\Phi^1(X)) = X \cup \Phi^1(X) \cup \Phi^2(X) \cup \Phi^2 \bullet \Phi^1(X)$.

Пусть задано множество $N = \{1, 2, \dots, n\}$.

Определение 2.3. (А.И. Молодченков). Последовательным применением n операторов $o_1 \dots o_n$, называется оператор

$$\tilde{o} = o_1 o_2 \dots o_n = \left\langle \begin{array}{c} \Phi^{\sim} \\ P_1^{\sim} \\ P_2^{\sim} \\ Z_1^{\sim} \\ Z_2^{\sim} \end{array} \right\rangle$$

такой, что $\Phi^{\sim} = 1 \cup \Phi^1 \cup \Phi^2 \cup \dots \cup \Phi^n \cup \bigcup_{k=2}^n \text{Com}(k)$, где 1 – единичный оператор, $\Phi^1, \Phi^2 \dots \Phi^n$ – функциональные компоненты операторов $o_1, o_2 \dots o_n$ соответственно, $\text{Com}(k) = \bigcup_{i=1}^{C_n^k} \bullet \text{com}_i$, где $\bullet \text{com}_i$ – композиция элементов com_i , com_i – i -о-к-элементное сочетание множества $\{\Phi_n, \Phi_{n-1}, \dots, \Phi_1\}$ такое, что все элементы $\Phi^{i_1}, \Phi^{i_2}, \dots, \Phi^{i_k} \in \text{com}_i$ ($i_1, i_2, \dots, i_k \in N$) упорядочены по убыванию значений i_1, i_2, \dots, i_k , $P_1^{\sim} = P_1^1 \cup P_1^2 \cup \dots \cup P_1^n$, $P_2^{\sim} = P_2^1 \cup P_2^2 \cup \dots \cup P_2^n$, $Z_1^{\sim} = Z_1^1 \cup Z_1^2 \cup \dots \cup Z_1^n$, $Z_2^{\sim} = Z_2^1 \cup Z_2^2 \cup \dots \cup Z_2^n$.

Если рассматривать применимость последовательности $o_1 o_2$ то, для оператора o_1 условие применимости $C_1 \subseteq \text{State}(t_1)$, а для оператора o_2 – $C_2 \subseteq \text{State}(t_2)$. Выразим это последнее условие через оператор $o_1[1]$.

$$\text{State}(t_2) = \text{State}(t_1) \cup \Phi^1(X) \cup P^1_1(t_2) \cup Z^1_1(t_2) \setminus (P^1_2 \cup Z^1_2).$$

Таким образом, для применимости последовательности операторов $o_1 o_2$ к некоторому состоянию $\text{State}(t_1)$ необходимо и достаточно[1]:

$$C_1 \subseteq \text{State}(t_1), C_2 \subseteq \text{State}(t_1) \cup (\Phi^1(X) \cup P^1_1(t_2) \cup Z^1_1(t_2)) \setminus (P^1_2 \cup Z^1_2) \quad (1).$$

Если, обозначить $(\Phi^1(X) = P^1_1(t_2) \cup Z^1_1(t_2))$ через A_1 , $(P^1_2(t_2) \cup Z^1_2(t_2))$ – через D_1 , ..., $(\Phi^n(X) = P^n_1(t_n) \cup Z^n_1(t_n))$ – через A_n , $(P^n_2(t_n) \cup Z^n_2(t_n))$ – через D_n то для последовательности из n операторов получим:

$$C_1 \subseteq \text{State}(t_1), C_2 \subseteq \text{State}(t_1) \cup A_1 \setminus D_1, \dots, C_n \subseteq \text{State}(t_1) \cup A_1 \setminus D_1, \dots, \cup A_n \setminus D_n (2)$$

Условия (1) и (2) называются *условиями применимости последовательности операторов*, а последовательности операторов, удовлетворяющие этим условиям – *применимыми* [1].

Далее докажем леммы о коммутативности и ассоциативности последовательного применения операторов.

Лемма 2.1(Г.С. Осипов, Г.И. Назаренко [1]). Последовательное применение операторов ассоциативно, т.е. $(o_1 o_2) o_3 = o_1 (o_2 o_3)$.

Доказательство (А.И. Молодченков). Ассоциативность операций теоретико–множественного объединения и взятия дополнения очевидна, поэтому следует доказать ассоциативность функциональной компоненты Φ . Вычислим для начала функциональную компоненту левой части доказываемого равенства. $(\Phi^1 \Phi^2) \Phi^3 = (1 \cup \Phi^1 \cup \Phi^2 \cup \Phi^2 \bullet \Phi^1) \Phi^3 =$
 $= 1 \cup (1 \cup \Phi^1 \cup \Phi^2 \cup \Phi^2 \bullet \Phi^1) \cup \Phi^3 \cup \Phi^3 \bullet (1 \cup \Phi^1 \cup \Phi^2 \cup \Phi^2 \bullet \Phi^1) =$
 $= 1 \cup \Phi^1 \cup \Phi^2 \cup \Phi^3 \cup \Phi^2 \bullet \Phi^1 \cup \Phi^3 \bullet \Phi^1 \cup \Phi^3 \bullet \Phi^2 \cup \Phi^3 \bullet \Phi^2 \bullet \Phi^1.$

Теперь вычислим функциональную компоненту правой части доказываемого равенства. $\Phi^1 (\Phi^2 \Phi^3) = \Phi^1 (1 \cup \Phi^2 \cup \Phi^3 \cup \Phi^3 \bullet \Phi^2) =$
 $= 1 \cup \Phi^1 \cup (1 \cup \Phi^2 \cup \Phi^3 \cup \Phi^3 \bullet \Phi^2) \cup (1 \cup \Phi^2 \cup \Phi^3 \cup \Phi^3 \bullet \Phi^2) \bullet \Phi^1 =$
 $= 1 \cup \Phi^1 \cup \Phi^2 \cup \Phi^3 \cup \Phi^3 \bullet \Phi^2 \cup \Phi^2 \bullet \Phi^1 \cup \Phi^3 \bullet \Phi^1 \cup \Phi^3 \bullet \Phi^2 \bullet \Phi^1.$

Видно, что выражения левой и правой частей равны друг другу, что доказывает лемму.

Лемма 2.2(Г.С. Осипов, Г.И. Назаренко [1]). Последовательное применение операторов не коммутативно, т.е. $o_1 o_2 \neq o_2 o_1$.

Доказательство (А.И. Молодченков). Операции теоретико–множественного объединения коммутативны, однако функциональная компонента Φ не коммутативна. Для левой части доказываемого неравенства функциональная компонента $\Phi = 1 \cup \Phi^1 \cup \Phi^2 \cup \Phi^1 \bullet \Phi^2$. Для правой части -

$\Phi = 1 \cup \Phi^1 \cup \Phi^2 \cup \Phi^2 \bullet \Phi^1$. Функциональные компоненты левой и правой частей не равны в виду того, что $\Phi^1 \bullet \Phi^2 \neq \Phi^2 \bullet \Phi^1$. Лемма доказана.

В тех случаях, когда необходимо выполнять перестановки операторов или выполнять их параллельное выполнение в [1] вводится операция соединения операторов, которая позволяет избавиться от члена $\Phi^2 \bullet \Phi^1$.

2.2. Маршруты в технологических процессах

В технологическом процессе можно выделить основные конструкты и основные элементы. К основным конструктам технологического процесса относятся ограничения, императивы и операторы [1], а к основным элементам – мероприятия, проводимые в процессе выполнения технологического процесса (например, лечебные мероприятия). В самом общем случае элементы включают в себя ограничения, императивы и операторы. Обозначим множество элементов через $E = \{e_1, e_2, \dots, e_n\}$, где каждый $e_i = \langle \text{Con}(t_i), \text{Imp}(t_i), \text{O}(t_i, t_{i+1}, T_i, V_i, \text{Loc}_i, Ex_i) \rangle$. Каждому этапу технологического процесса соответствует несколько элементов, т.е. k-й этап представляет собой подмножество E' декартова произведения множеств $\text{CON}(t_k)$, $\text{IMP}(t_k)$ и $\text{O}(t_k, t_{k+1}, T_k, V_k, \text{Loc}_k, Ex_k)$:

$$E' \subseteq \text{Con}(t_k) \times \text{Imp}(t_k) \times \text{O}(t_k, t_{k+1}, T_k, V_k, \text{Loc}_k, Ex_k).$$

Однако при такой записи не ясно, в какой последовательности выполняются операторы из множества O . Следует отметить, что с операторами из множества O могут быть связаны условия их выполнения, что приводит к появлению точек ветвления технологического процесса. Элементы из множества E образуют маршруты технологического процесса, а процесс построения этих маршрутов называется маршрутизацией. Для снятия неопределенности выделяют четыре вида маршрутизации [105]: последовательная, параллельная, условная и итеративная. Последовательная маршрутизация соответствует временному отношению строгого линейного порядка на множестве элементов. Это означает, что мероприятие $e(t_2) \in E$

может быть выполнено *только* после того, как будет выполнено $e(t_1) \in E$. При параллельной маршрутизации порядок выполнения мероприятий не имеет значения. При условной маршрутизации производится выбор между двумя или более маршрутами. Итеративная маршрутизация соответствует повторению некоторого маршрута, пока не будут выполнены критерии выхода из итерации.

Используем следующие обозначения [1]:

- символом « \rightarrow » будем обозначать последовательную маршрутизацию;
- « \leftrightarrow » – параллельную;
- « $\uparrow\downarrow$ » – условную;
- (...) Kt – итеративную, где в скобках – повторяемая последовательность операций, а Kt – критерий окончания повторения.

Набор перечисленных типов маршрутов фактически покрывает все многообразие возможных путей лечения различных заболеваний. Графы, построенные на их основе, с различной степенью детализации, помогают врачу видеть весь процесс лечения и принимать оперативные решения. Состав лечебно-диагностических мероприятий определяется возможностями лечебного учреждения, ресурсами, особенностями пациента. Более того, для каждой нозологической формы существуют различные наборы лечебно-диагностических мероприятий и, соответственно, различные пути реализации всего процесса.

Таким образом, в общем виде технологический процесс представляет собой последовательность этапов $E_1 \rightarrow E_2 \rightarrow \dots \rightarrow E_n$, где E_i – этапы технологического процесса ($i = 1, 2, \dots, n$). Каждый этап представляет собой структурированное множество элементов. Например, для этапа E_1 оно может иметь вид:

$$(e_{11} \rightarrow e_{12} \rightarrow (e_{13} \leftrightarrow e_{14}) \rightarrow (e_{15} \uparrow \downarrow e_{16})).$$

Здесь у каждого элемента первый нижний индекс означает номер этапа – второй – номер в этапе. Эта запись означает, что элементы e_{11} и e_{12} выполняются строго последовательно, затем, в любой последовательности выполняются e_{13} и e_{14} и после выполнения этих мероприятий выполняется один из элементов e_{15} или e_{16} , в зависимости от выполнения условий, входящих в ограничения элемента e_{15} и в ограничения мероприятия e_{16} .

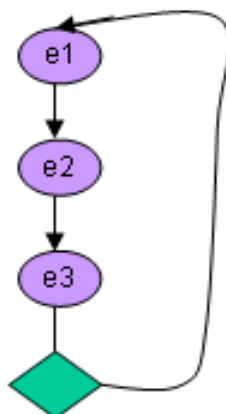
2.3. Графическое представление технологического процесса

Для графического представления технологических процессов будем использовать следующие соглашения [1, 106]: графическое изображение технологических процессов представляет собой набор активных графических объектов (прямоугольников и эллипсов) и условий, изображаемых ромбами. В прямоугольниках указано место лечения больного, а в эллипсы – лечебные операторы и манипуляции, выполняемые в рамках технологического процесса. Последовательное расположение стрелок указывает на строго последовательные лечебные мероприятия, две стрелки, исходящие из одного прямоугольника или эллипса свидетельствуют о параллельной маршрутизации, две стрелки исходящие из ромба означают условную маршрутизацию, а стрелка, исходящая из ромба и входящая в некоторую вершину, предшествующую исходящей, обозначает итеративную маршрутизацию. На рисунке 2.1 показано графическое представление различных типов маршрутов в потоках работ.

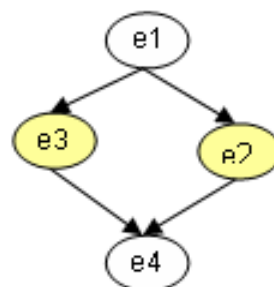
а) последовательный



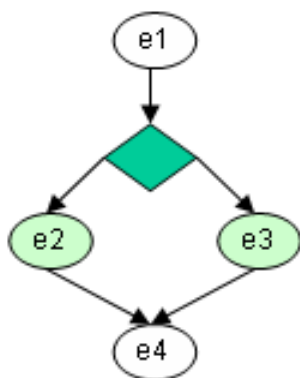
b) итеративный



с) параллельный



d) условный



е) конкурентный

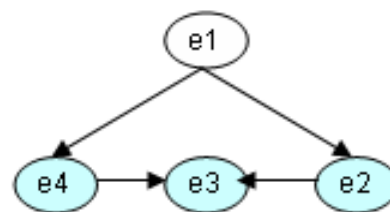


Рисунок 2.1. Типы маршрутов в потоках работ

На рисунке 2.2 показаны примеры упрощенного представления медицинских технологических процессов бронхиальной астмы и железодефицитной анемии. В этих примерах присутствуют практически все элементы из модели технологических процессов.

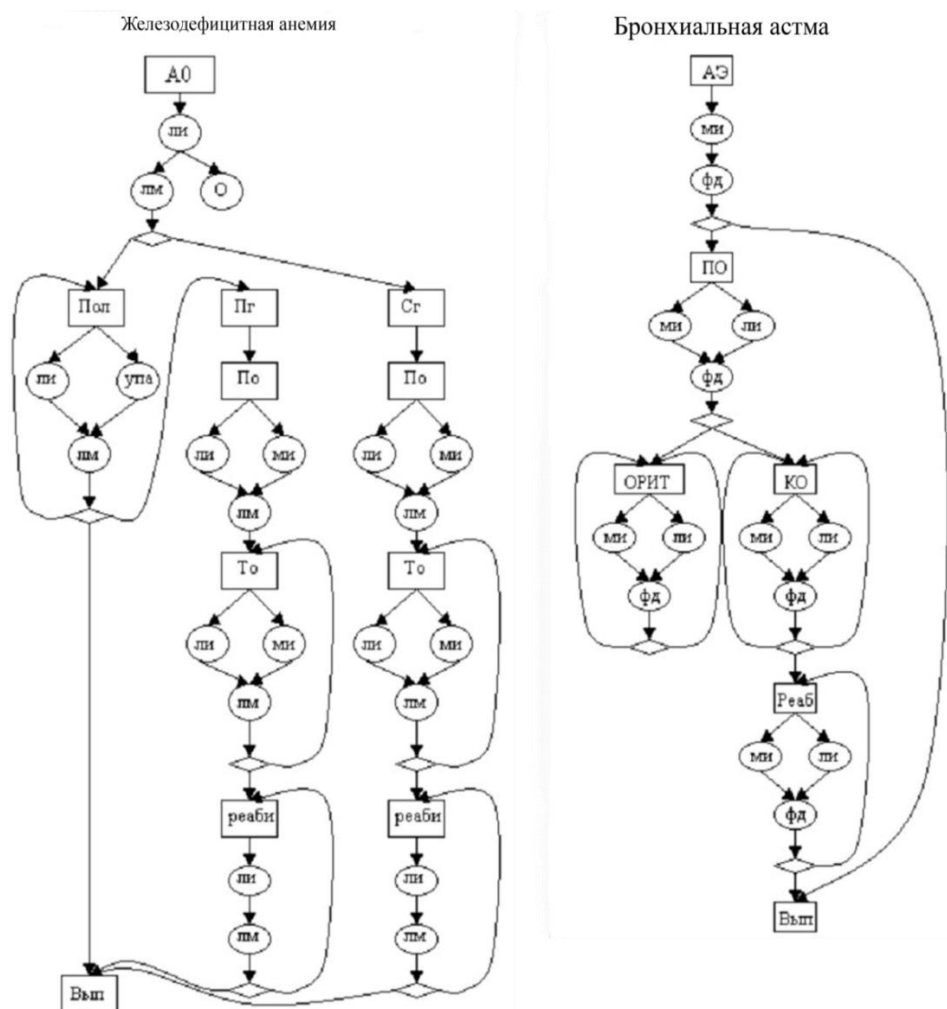


Рисунок 2.2. Упрощенные примеры МТП Бронхиальной астмы и железодефицитной анемии, АЭ – амбулаторный этап, лм – манипуляции и исследования, ли – лабораторные исследования, фд – физикальные данные, ПО – приемное отделение, ОРИТ – отделение реанимации, КО – клиническое отделение, реаб – реабилитация, вып – выписка, лм – лечебные мероприятия, о – оценка тяжести обострения, упа – установление причин анемии, пол – поликлиника, пг – плановая госпитализация, сг – срочная госпитализация, ТО – терапевтическое отделение

Медицинский технологический процесс должен соответствовать стандарту лечения нозологии или протоколу ведения больных (Российский аналог клинических руководств). Для выполнения автоматической проверки этого соответствия стандарт или протокол ведения больных должны быть

представлены в формальном виде. Для этого формального описания предлагается использовать онтологию на основе семантических сетей.

2.4. Онтологический подход для описания стандартов лечения и протоколов ведения больных

Онтология – это формальное описание медицинских знаний, касающихся медицинских технологических процессов. В качестве способа представления знаний используется семантическая сеть. Узлы сети соответствуют классам, экземплярам классов и другим элементам медицинского технологического процесса. Ребра семантической сети соответствуют отношениям на них.

Для описания онтологии используется язык представления онтологий OWL (*Ontology Web Language*) в диалекте DL (*Description Logic*) и редактор онтологий Protégé (<http://protege.stanford.edu/>). Protégé имеет открытую, легко расширяемую архитектуру, поддерживает все наиболее распространенные языки представления знаний (SHOE, XOL, DAML+OIL, RDF/RDFS, OWL) и модули расширения функциональности (plug-in). Protégé основан на модели представления знаний ОКВС (*Open Knowledge Base Connectivity*). Основными элементами онтологии являются классы, их экземпляры, слоты (представляющие свойства классов и экземпляров) и фасеты, задающие дополнительную информацию о слотах. Классы обладают двумя типами свойств: свойства-данные и свойства-отношения. Свойства-данные соответствуют характеристикам, которым можно присвоить некоторое значение (числовое, строковое и т.п.). Например, имя, идентификатор, возраст. Свойства-отношения представляют собой связи с классами, экземплярами классов и другими элементами онтологии.

Класс «*Медицинский_технологический_процесс*» содержит множество лечебно-диагностических мероприятий (действий), применимых к пациенту, и множество этапов, из которых состоит МТП. Обязательно присутствует начальное действие (start_action), которое является точкой входа МТП. Класс «*Этап_ТП*» является подклассом класса

«Медицинский_технологический_процесс» ввиду того, что каждый этап медицинского технологического процесса – это его подпроцесс. Этап МТП может быть разбит на множество более мелких этапов и т.д.

Действия в МТП носят лечебно-диагностический (например, «сбор анамнеза и жалоб при заболеваниях легких и бронхов») или административный (например, «перевод в клиническое отделение») характер, связаны с временными характеристиками, местом проведения, исполнителем и т.д. Действия могут быть простыми и составными. Примером простого действия является «Пульсоксиметрия». Составное действие состоит из набора простых действий и может включать в себя составные мероприятия. Например, «Диагностические исследования в приемном отделении» при обострении бронхиальной астмы включает в себя следующие действия: «Биохимический анализ крови», «Общий анализ крови», «Общий анализ мочи», «Пульсоксиметрия» и др.

Класс времени делится на три группы. Первая группа касается временных периодов, таких как «Время пребывания в стационаре», «Первый день после проведения хирургического лечения» и т.д. Вторая группа относится к временным моментам и связана с моментами начала и завершения лечебного мероприятия. Третья группа касается временных отношений между действиями. Эта группа может быть разделена на две категории. Первая категория называется относительным временным отношением, которое описывает положение выполнения двух вмешательств относительно друг друга. Например, действие В выполняется через 4 часа после исполнения действия А. Вторая категория называется отношением временного интервала, которое сопоставляет некоторый известный период времени с интервалом между началом и окончанием выполнения двух действий. Эта категория служит для выявления отклонений в процессе исполнения МТП. В качестве известного интервала выступает временной интервал первой категории между двумя действиями, указанный в онтологии. Далее происходит вычисление времени между началом и

окончанием выполнения этих действий и сопоставление. В связи с этим для действий допускается указывать по два момента времени допустимого начала и окончания выполнения.

С каждым действием, кроме времени, связаны «Роль», «Место_проведения» и «Ресурсы». Эти классы соответствуют различным механизмам поддержки выполнения лечебно-диагностических мероприятий. Класс «Роль» имеет два подкласса: «Пациент» и «Персонал». «Персонал» делится на следующие подклассы: «Врач», «Медсестра», «Обслуживающий_персонал» и «Руководство». Классу «Место_проведения» соответствуют подразделения медицинского учреждения, в которых выполняются соответствующие действия. В качестве ресурсов выступают оборудование, медикаменты и т.п.

Каждому из типов маршрутов в онтологии соответствуют свои классы и отношения. Для обозначения того, что два действия e_1 и e_2 выполняются последовательно, было введено отношение $next(e_1, e_2)$, которое означает, что действие e_2 выполняется только после выполнения e_1 . Для остальных типов маршрутов были введены одноименные классы. Для тех случаев, когда время начала выполнения $e_2 \leq$ времени окончания выполнения e_1 , было введено понятие конкурентного маршрута. Класс «Параллельный_маршрут» является подклассом класса «Маршрутизация» и обладает свойством-отношением «*action*», которое связывает экземпляр класса «Параллельный_маршрут» с экземплярами класса «Элемент_техпроцесса». Это отношения типа «один ко многим», т.к. в одном параллельном маршруте может быть исполнено несколько элементов технологического процесса. Класс «Условный_маршрут» также является подклассом класса «Маршрутизация» и обладает свойством-данным «*условие*» и двумя свойствами-отношениями *true_action* и *false_action*. Свойство-данное принимает строковое значение, в котором в определенном формате описывается условие дальнейшего движения по технологическому процессу. На основании описанных условий формируется система правил, позволяющая управлять МТП в процессе

исполнения. Свойства *true_action* и *false_action* аналогичны свойству «*action*» класса «*Параллельный_маршрут*» и соответствуют дальнейшему движению по технологическому процессу, если условие истинно или ложно. Класс «*Итеративный_маршрут*» является подклассом «*Маршрутизация*» и обладает свойством-данным «*условие*» и свойством-отношением «*action*». В свойстве «*условие*» прописывается условие выхода из итерации. Свойство «*action*» аналогично соответствующему свойству в классе «*Параллельный_маршрут*». Кроме классов, соответствующих различным типам маршрутов, к элементам технологического процесса относятся лечебно-диагностические и организационные мероприятия. Соответствующие классы введены как подклассы класса «*Элемент_техпроцесса*». Далее лечебно-диагностические и организационные мероприятия делятся на более мелкие классы и т.д. Пример онтологии лечения пациента с обострением бронхиальной астмы (БА) изображен на рисунке 2.3.

Весь процесс лечения БА можно разбить на несколько этапов, каждом из которых проводятся соответствующие лечебно-диагностические мероприятия: приемное отделение (ПО), отделение реанимации и интенсивной терапии (ОРИТ) и коечное отделение (КО). Таким образом, в созданном экземпляре технологического процесса созданы следующие экземпляры класса «*Этап_ТП*»: «*Этап_ПО*», «*Этап_ОРИТ*» и «*Этап_КО*». На рисунке изображен этап приемного отделения (ПО). Целями этапа выступают «Подтверждение диагноза БА» и «Оценка степени тяжести заболевания». Стартовым мероприятием этапа (точка входа) является оценка жизненно важных функций пациента, по результатам которой он может быть сразу направлен в ОРИТ при наличии жизнеугрожающего состояния, либо обследован в ПО. Этой ситуации соответствует условный маршрут. В условии устанавливается проверка состояния пациента: жизнеугрожающее или нет. В *true_actions* устанавливается экземпляр класса

«Организационные мероприятия» «Перевод в ОРИТ», в *false_actions* записываются мероприятия, проводимые далее в ПО.

Обследование в ПО включает ряд последовательных мероприятий, выполняемых терапевтом («осмотр», «сбор анамнеза», «получение информированного согласия»), а также комплексное мероприятие «Диагностическое исследование в ПО», состоящее из ряда действий, выполняемых параллельно.

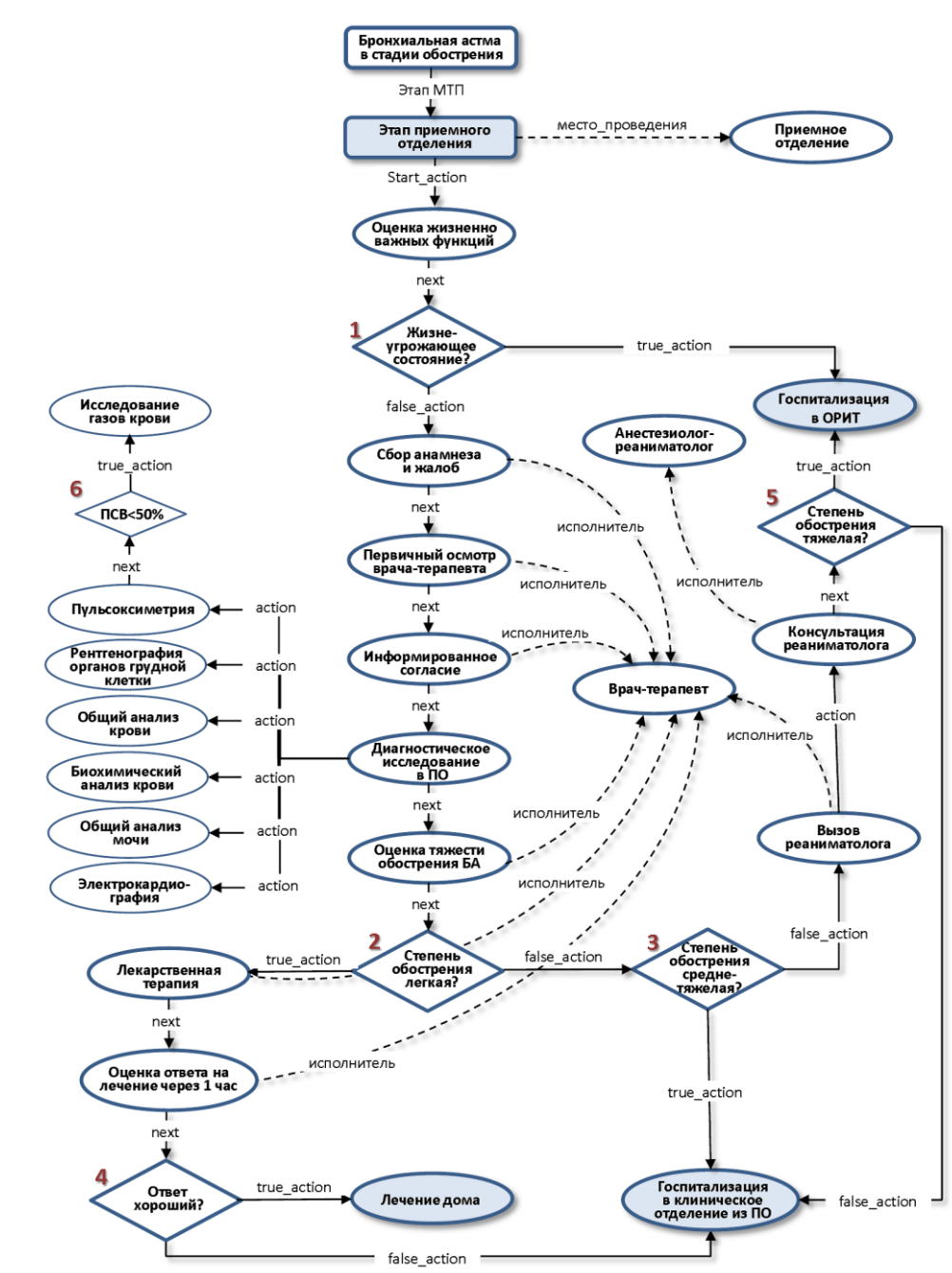


Рисунок 2.3. Пример онтологии лечения пациента с обострением бронхиальной астмы

Действия относятся к основному классу «Элемент_техпроцесса», имеют свойства-данные «код_услуги», «длительность_ЛДМ», «приоритетность» и связаны между собой связью «next» либо «action».

Окончанием комплексного мероприятия может быть выполнение как всех, так и любого числа входящих в него действий. В случае «Диагностического исследования в ПО» для перехода к следующему лечебно-диагностическому мероприятию (ЛДМ) – «Оценке тяжести обострения БА» – необходимо, чтобы были завершены «Пульсоксиметрия» и «Исследование газов крови».

После выполнения вышеперечисленных мероприятий производится оценка обострения бронхиальной астмы, которое имеет 4 степени тяжести: лёгкая, среднетяжёлая, тяжёлая и жизнеугрожающая. При легкой степени тяжести пациенту назначают лекарственные препараты, оценивают реакцию на них через 1 час и при наличии хорошего ответа на лечение отпускают домой. В противном случае пациента госпитализируют в коечное отделение. При среднетяжелой степени обострения БА пациента госпитализируют в коечное отделение, а при тяжелой или жизнеугрожающей (в период пребывания в ПО) врач приемного отделения вызывает реаниматолога и после его консультации пациента госпитализируют в ОРИТ. Таким образом, образуется три условных маршрута движения пациента (рис.1). Условиями, определяющими принятие решения о выборе маршрута, в первом и пятом случаях являются критерии перевода в ОРИТ, во втором и третьем – критерии степени тяжести обострения БА, в четвертом – критерии хорошего ответа на лечение, в шестом – величина пиковой скорости выдоха.

Для регистрации отклонений и оценки качества выполнения МТП в онтологию был введен класс «Контрольные точки». Индикаторами качества МТП могут быть факт выполнения ЛДМ, относящегося к классу контрольных точек, либо его характеристики (например, дата-время исполнения). В описываемом примере для этапа ПО индикаторами качества

явились длительность пребывания пациента в приемном отделении и регистрация степени тяжести БА.

Применение онтологического подхода для разработки систем поддержки медицинских технологических процессов активно развивается.

2.5. Выводы

В главе определены понятия модели технологического процесса в области медицины и психологии. Описаны составляющие технологического процесса, которые включают в себя индикаторы, ограничения, императивы, операторы, элементы и т.д. Предложено определение последовательного применения n операторов. Доказаны леммы об ассоциативности и коммутативности последовательного применения операторов. Определены понятия состояния и маршрутов технологического процесса.

В конце описан онтологический подход для описания медицинских технологических процессов. Приведен пример онтологии МТП на примере процесса лечения бронхиальной астмы.

Перейдем теперь к описанию алгоритма автоматического синтеза персонализированных медицинских технологических процессов.

Глава 3. Построение общего описания технологического процесса на основе прецедентной информации

Рассматривается алгоритм автоматического синтеза общего описания технологического процесса на основе прецедентной информации. Описан метод адаптации медицинских технологических процессов к индивидуальным особенностям пациентов. Описаны результаты экспериментальных исследований предложенного алгоритма автоматического построения технологических процессов.

3.1. Алгоритм автоматического синтеза общего описания технологического процесса на основе прецедентной информации

Через $\Omega(O)$ обозначим семейство последовательностей вида $w = \langle o_i, o_j, \dots, o_k \rangle$, где w – упорядоченное множество операторов, $i < j < \dots < k$ – элементы множества натуральных чисел N . На множестве $\Omega(O)$ задано некоторое отношение эквивалентности ρ , порождающее фактор-множество Ω_ρ и разбивающее множество всех примеров на классы эквивалентности (например, по нозологическим формам). Ближайшей целью является построение полного описания всех классов эквивалентности. Другими словами, задача состоит в том, чтобы опираясь на множество примеров, входящих в классы эквивалентности $\{w\}$, построить такие описания $G(\{w\})$, что каждый пример, входящий в $\{w\}$, удовлетворял бы его описанию $G(\{w\})$.

Описание классов эквивалентности будем строить в виде графа, пример которого изображен на рисунке 3.1.

То обстоятельство, что все элементы в $\{w\}$ удовлетворяют описанию, означает, что для каждого примера w из $\{w\}$ в графе $G(\{w\})$ должен найтись эквивалентный ему подграф.

Все графы $G(\{w\})$ будем представлять посредством их матриц смежности. В [107] была введена операция покомпонентного сложения матриц смежности:

если $A=[a_{ij}]$ и $B=[b_{ij}]$ –матрицы смежности, то $A + B = [a_{ij}] + [b_{ij}] = \max([a_{ij}], [b_{ij}])$.

Для последовательности o_1, o_2, \dots, o_k матрица смежности будет иметь следующий вид:

$$M\{o_1, o_2, \dots, o_k\} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix},$$

где $a_{12} = a_{23} = \dots = a_{k-1,k} = 1$, все остальные элементы равны 0. Матрица $M\{o_1, o_2, \dots, o_k\}$ соответствует последовательному маршруту в технологическом процессе.

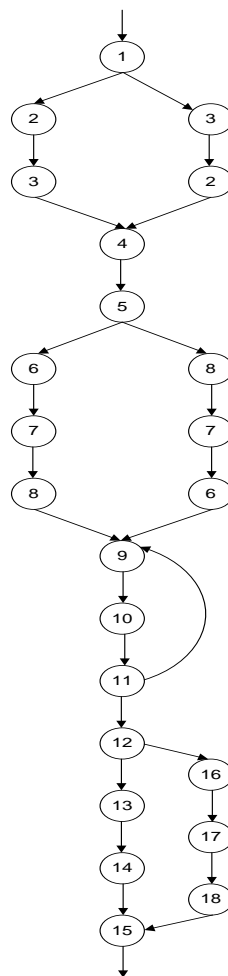


Рисунок 3.1. Пример описания класса эквивалентности

Как было написано выше, технологический процесс содержит четыре типа маршрутов: последовательный, параллельный, условный и итеративный. Кроме перечисленных маршрутов в работе будет рассмотрен еще один тип маршрутизации – конкурентный. Конкурентный маршрут возникает при синтезе следующих экземпляров технологических процессов: $(e_{11} \rightarrow e_{13} \rightarrow e_{12})$ и $(e_{12} \rightarrow e_{13} \rightarrow e_{11})$.

В [107] доказаны корректность и полнота описаний классов эквивалентности с помощью матриц смежности. Таким образом, матрица смежности, содержащая описание обобщенной схемы технологического процесса, содержит в себе все типы маршрутов.

Опишем алгоритм автоматического построения обобщенной схемы технологического процесса на основе прецедентной информации для медицинских технологических процессов [108, 109].

Алгоритм 3.1 (общий алгоритм автоматического построения ТП)

- 1) построение матриц смежности, описывающих экземпляры технологических процессов;
- 2) разбиение экземпляров технологических процессов на классы эквивалентности и их кластеризация;
- 3) выбор опорного примера в каждом классе эквивалентности;
- 4) синтез обобщенной схемы технологического процесса для каждого класса эквивалентности путем объединения его экземпляров.

Алгоритм 3.2 (алгоритм построения матриц смежности, описывающих экземпляры технологических процессов):

- 1) извлечение действий (операторов);
- 2) сортировка списков действия по времени их выполнения (построение $\Omega(O)$);
- 3) построение матриц смежности для каждого примера, входящего в $\{w\}$.

Все действия, которые были выполнены в частном ТП, извлекаются из лог файла, базы данных и других источников, и записываются в массив w_i ,

где i – индекс примера технологического процесса. После извлечения всех действий, они сортируются по возрастанию относительно времени их выполнения. В результате получится множество экземпляров $\Omega(O)$.

Следующим шагом является построение описаний в виде матриц смежности каждого экземпляра в $\Omega(O)$. С номером каждой строки и столбца матрицы смежности связан некоторый оператор (лечебно-диагностическое мероприятие). С одним и тем же оператором может быть связано несколько строк, т.к. при последовательной маршрутизации одно и то же лечебное мероприятие могут делать несколько раз и отсутствует итеративный маршрут.

Пусть $M^k = [m_{ij}^k]$ – матрица смежности для k -го примера технологического процесса, h – размерность M^k , (перед началом работы алгоритма построения матрицы смежности для экземпляра технологического процесса $h = 0$).

Алгоритм 3.3 (алгоритм построения матрицы смежности для экземпляра технологического процесса):

- 1) считывается оператор o_i из экземпляра $x \in \Omega(O)$, где $i = 0, \dots, n$ – индекс оператора в экземпляре технологического процесса;
- 2) если этот оператор ранее не выполнялся, то он добавляется в матрицу M^k ; $M^k[h, h+1] = 1$, во всех остальных пустых ячейках устанавливаются 0, h и i увеличиваются на 1, переход к шагу 1;
- 3) если оператор ранее выполнялся и находится в j -ом столбце ($j < i$), то запускается процедура поиска итерации;
- 4) если удалось обнаружить итерацию, то $M^k[h, j] = 1$, запуск процедуры объединения операторов, расположенных между строками j и h матрицы M^k и после i -го оператора в x , i увеличивается на количество элементов в итерации, переход к шагу 1;

5) если итерации нет, то оператор добавляется в матрицу M^k , $M^k[h, h+1] = 1$, во всех остальных пустых ячейках устанавливаются 0, h и i увеличиваются на 1, переход к шагу 1.

Алгоритм проиллюстрирован на рисунке 3.2 и завершит работу, когда будут обработаны все операторы из экземпляра $x \in \Omega(O)$.

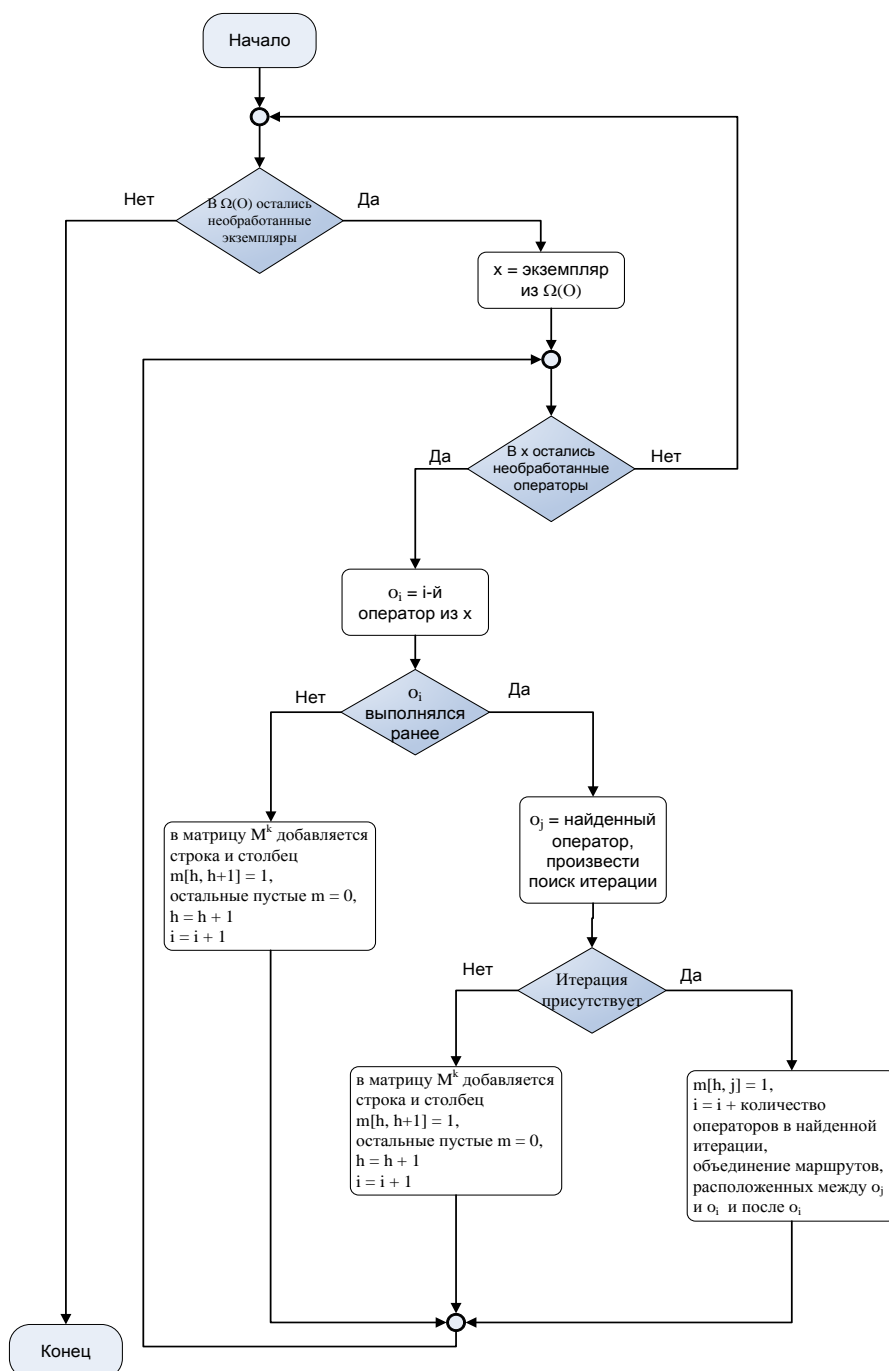


Рисунок 3.2. Алгоритм построения матрицы смежности экземпляра ТП

Итеративные маршруты возникают в том случае, если при анализе примера технологического процесса обнаружен оператор, выполненный

ранее. В этом случае проверяется, совпадают ли множества операторов, выполненных вслед за предыдущим и текущим выполнением обнаруженного оператора. Итеративный маршрут порождается, если совпадение обнаружено.

Опишем процедуру поиска итерации. Для определения совпадения используется следующая функция релевантности [107]:

$$rel(\pi_1, \pi_2) = \varphi(\rho_1(e_1, \pi_2), \dots, \rho_n(e_n, \pi_2)),$$

где π_1, π_2 – множества операторов между предыдущим и текущим положением оператора и следующим за текущим, e_1, \dots, e_n – элементы π_1 . Положим

$$\rho(e_i, \pi_2) = \begin{cases} 1, & \text{если } e_i \in \pi_2 \\ 0, & \text{если } e_i \notin \pi_2 \end{cases},$$

а φ – линейная комбинация ρ_i , где $i \in \{1, \dots, n\}$, нормированная на 1, т.е.

$$rel(\pi_1, \pi_2) = \frac{1}{n} \sum_{i=1}^n \rho_i.$$

При вычислении релевантности принимается во внимание близость множеств элементов (лечебных мероприятий) в сравниваемых экземплярах и не принимается во внимание порядок их следования.

Если значение релевантности больше некоторого коэффициента k , то будем считать, что итерация обнаружена, иначе нет. Значение k определяется в зависимости от решаемой задачи. Для медицинских технологических процессов в [107] k полагается равным 0,8.

Объединение маршрутов, образующих итерацию, заключается в построении матрицы смежности M^k для последовательности мероприятий, проводимых после текущего выполнения исследуемого оператора. После чего производится объединение фрагмента матрицы M^k , в котором находятся операторы итерации, и M^k .

Следующей задачей является разбиение множества экземпляров $\Omega(O)$ на классы эквивалентности. Для выполнения данной операции на множестве

$\Omega(O)$ вводится отношение эквивалентности ρ . Для медицинских технологических процессов таким отношением является выбор примеров по общей нозологии. В один класс эквивалентности записываются экземпляры, направленные на лечении одного заболевания.

Для построения классов эквивалентности введем следующее отношение:

$$f : \Omega(O) \rightarrow U,$$

где U – множество нозологических форм.

Два экземпляра $x \in \Omega(O)$ и $y \in \Omega(O)$ принадлежат одному классу эквивалентности тогда и только тогда, когда $f(x) = f(y)$. Алгоритм разбиения множества $\Omega(O)$ на классы эквивалентности проиллюстрирован на рисунке 3.3.

Для каждого экземпляра $y \in \Omega(O)$ производится проверка на равенство $f(x) = f(y)$ со всеми x – примерами, входящими в классы эквивалентности $\{w\}$. Если удалось найти такой пример x , такой что $f(x) = f(y)$, то записываем y в класс эквивалентности, которому принадлежит x . Если не удалось найти ни одного примера x , что $f(x) = f(y)$, то добавляем во множество $\{w\}$ новый класс эквивалентности и записываем в него y .

После разбиения частных технологических процессов на классы эквивалентности производится синтез обобщенной схемы ТП.

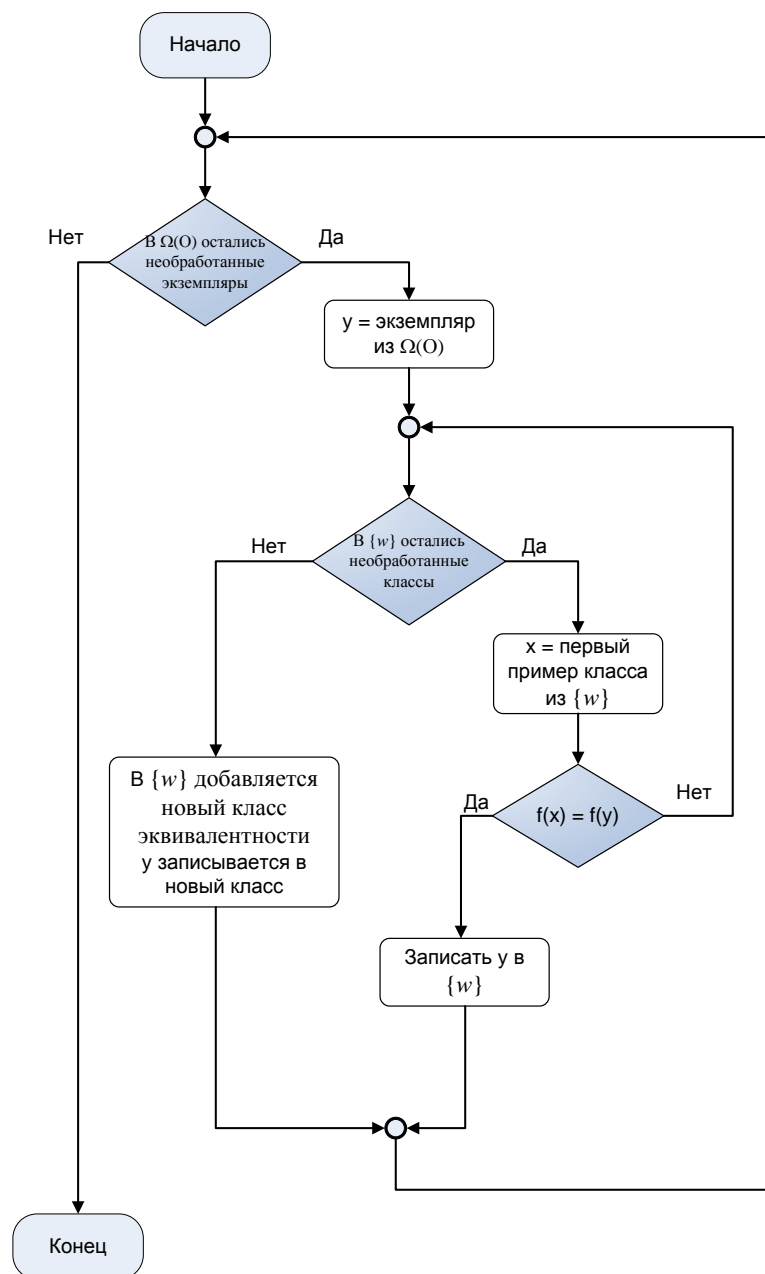


Рисунок 3.3. Алгоритм разбиения множества $\Omega(O)$ на классы эквивалентности

Пусть $\{M^k\}$ - множество матриц смежности, описывающих частные технологические процессы для k-го класса эквивалентности.

Алгоритм 3.4 (алгоритм автоматического синтеза обобщенной схемы технологических процессов):

- 1) приведение всех матриц смежности из $\{M^k\}$ к одной размерности таким образом, чтобы их строки и столбцы указывали на одни и те же операторы;

- 2) построить матрицу M – являющуюся объединением всех матриц из $\{M^k\}$;
- 3) убрать шумы в получившемся описании обобщенной схемы технологического процесса;
- 4) выявить все маршруты в описании обобщенной схемы технологического процесса;
- 5) построить условия в точках ветвления технологического процесса.

Пусть MCh –список обработанных алгоритмом матриц смежности, Op – массив, который хранит в себе перечень лечебно-диагностических мероприятий, проводимых в классе эквивалентности k .

Алгоритм 3.5 (алгоритм построения матрицы смежности для описания обобщенной схемы МТП на конкретном этапе):

- 1) из множества $\{M^k\}$ выбирается матрица M^k самой большой размерности;
- 2) в Op записываются все лечебно-диагностические мероприятия, связанные со строками матрицы M^k ;
- 3) матрица M^k добавляется в MCh ;
- 4) из множества $\{M^k\}$ выбирается не обработанная матрица M_j^k ;
- 5) h = количество строк в матрице M_j^k , $i = 0..h$;
- 6) в Op записывается оператор, связанный с i -ой строкой матрицы M_j^k ;
- 7) если o совпадает с $Op[i]$, то i увеличивается на 1, переход к шагу 6;
- 8) если o не совпадает с $Op[i]$, то

8.1. в Op , начиная с $(i+1)$ -ой позиции, производится поиск позиции, в которой совпадут операторы в Op и в i -ой строке M_j^k ;

- 8.2. если позицию найти удалось и ее номер равен m , то в матрице M_j^k оператор перемещается на строку и столбец с номером m , переход к шагу 6;
- 8.3. если позицию найти не удалось, то в Op добавляется оператор o , в матрице M_j^k оператора перемещается на строку и столбец с номером m , равным номеру позиции оператора o в множестве Op , во все матрицы из MCh добавляется строка и столбец, заполненные нулями, переход к шагу 6;
- 9) если в Op остались еще не обработанные операторы при обработке матрицы M_j^k , то они добавляются в соответствующие позиции матрицы M_j^k в виде строк и столбцов, заполненных нулями;
- 10) переход к шагу 3.

Алгоритм заканчивает свою работу в тот момент, когда все матрицы в $\{M^k\}$ будут обработаны. В результате работы алгоритма в MCh находятся все матрицы из множества $\{M^k\}$, одной размерности и их строки и столбцы указывают на одни и те же операторы.

На рисунке 3.4 проиллюстрирован алгоритм преобразования матриц из множества $\{M^k\}$, и формирующих множество MCh .

После формирования множества MCh , необходимо построить матрицу смежности, описывающую обобщенную схему МТП. Эту матрицу обозначим через M . Размерность матрицы M равна h — числу элементов множества Op , каждая i -я строка матрицы ($i = 1 \dots h$), связана с оператором (лечебно-диагностическим мероприятием), расположенным в i -ой позиции множества Op . Матрица M строится путем объединения всех матриц смежности в множестве MCh .

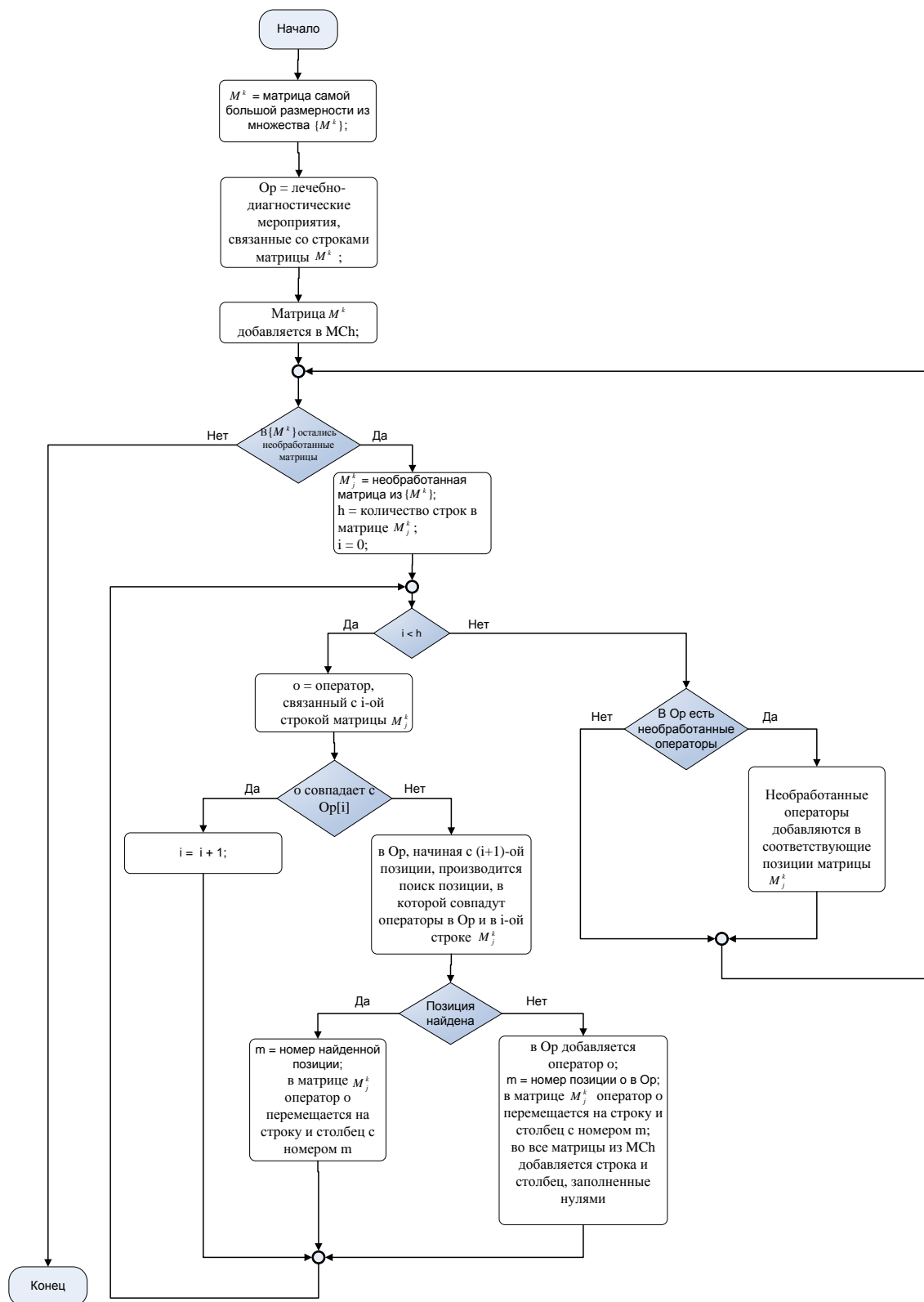


Рисунок 3.4. Алгоритм преобразования матриц из множества $\{M^k\}$.

Следующая задача: учет влияния шумов в описании технологического процесса. Через $a \rightarrow b$ обозначим тот факт, что оператор b выполняется после выполнения оператора a . Для каждой пары операторов (a, b) в матрице M ,

которые находятся в отношении $a \rightarrow b$, рассчитывается значение следующей метрики [110]:

$$p = \frac{e^{LR}}{1 + e^{LR}},$$

где значение LR, рассчитывается по следующей формуле [110]:

$$LR = -8.280 + 6.376 * LM + 4.324 * GM + 8.654 * CM$$

LM, GM, CM являются отдельными метриками для удаления шумов [46, 110-113]. Обозначим через $(a \rightarrow b)$ количество частных технологических процессов, в которых имеет место отношение $a \rightarrow b$, #A, #B – количество частных технологических процессов, в которых присутствуют операторы a и b соответственно, #L – общее количество частных технологических процессов в классе эквивалентности.

Метрика LM (local metric) [111, 113]:

$$LM = P - 1.96 \sqrt{\frac{P(1-P)}{N+1}},$$

где $P = \frac{(a \rightarrow b)}{N+1}$, $N = (a \rightarrow b) + (b \rightarrow a)$.

Метрика GM (global metric)[111, 113]:

$$GM = ((a \rightarrow b) - (b \rightarrow a)) \frac{\#L}{\#A * \#B}$$

Метрика CM (causal metric) [110]:

$$CM = \frac{C(a \rightarrow b)}{\#A},$$

где значение $C(a \rightarrow b)$ рассчитывается следующим образом; для каждого прецедента $C(a \rightarrow b)$ увеличивается на величину δ^n . В результате проведенных экспериментов авторами этой метрики было установлено значение δ равным 0.8. n – количество элементов между a и b. В случае если элемент b следует сразу после a (т.е. $n = 0$), то $C(a \rightarrow b)$ увеличивается на 1.

Пороговое значение σ для метрики p было получено экспериментальным путем и равно 0.8 [110]. Это означает, что $a \rightarrow b$ в

построенной модели технологического процесса не является шумом в том случае если значение p больше 0.8. Авторы этого подхода экспериментально показали, он в 95% случаев корректно оставляет отношение следования между элементами a и b . В остальные 5% случаев, чаще всего, входят ситуации, когда действия a и b выполняются достаточно редко. В этом случае значения LM, GM и CM достаточно низкие. Несмотря на этот недостаток эта метрика вполне пригодна для устранения шумов из модели технологического процесса.

Алгоритм 3.6 (алгоритм отсеечения шумов) состоит из следующих шагов:

- 1) выбирается пара элементов a и b , такие, что $a \rightarrow b$;
- 2) для этой пары элементов рассчитываются критерии LM, GM, CM, LR и p ;
- 3) если значение p не меньше 0.8, то отношение $a \rightarrow b$ сохраняется;
- 4) иначе, в матрице смежности в соответствующей ячейке ставится 0;
- 5) переход к шагу 1.

Алгоритм заканчивает свою работу, когда все пары элементов a и b , такие, что $a \rightarrow b$, обработаны.

После работы алгоритма отсеечения шумов из матрицы смежности удаляются все элементы, не имеющие входных связей и не являющиеся начальными, и элементы, не имеющие выходных связей и не являющиеся конечными.

Завершающим этапом построения обобщенной схемы технологических процессов является извлечение различных типов маршрутов (потоков работ) на основе извлеченной прецедентной информации. Ребро, соединяющее операторы e_i и e_j , будем обозначать через $\text{edge}(e_i, e_j)$.

Извлечение различных типов маршрутов в ТП производится путем применения к матрице M правил.

Система правил

Правило 3.1: *ЕСЛИ* у элемента e_i в M нет ни одного исходящего ребра, *ТО* данный элемент является последним в технологическом процессе.

Правило 3.2: *ЕСЛИ* у элемента e_i в M нет ни одного входящего ребра, *ТО* данный элемент является первым в частном технологическом процессе.

Правило 3.3: *ЕСЛИ* количество исходящих ребер у элемента e_i в M равно 1, *ТО* элементы e_i и e_{i+1} образуют последовательный маршрут.

Правило 3.4: *ЕСЛИ* количество исходящих ребер у элемента e_i в M больше 1 *И* существует минимум один e_k такой, что $k \leq i$ и имеет место $\text{edge}(e_i, e_k)$ и существует путь от e_k к e_i , состоящий из $\{e_j\}$ ($k \leq j \leq i$), *ТО* со всеми e_k от e_i строятся итеративные маршруты, включающие в себя элементы e_k, \dots, e_i ; к остальным элементам применяются правила 3.3 – 3.6.

Приведем пример случая, для которого применяется правило 3.4. Матрица смежности выглядит следующим образом:

$$M = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Графически эта ситуация изображена на рисунке 3.5.

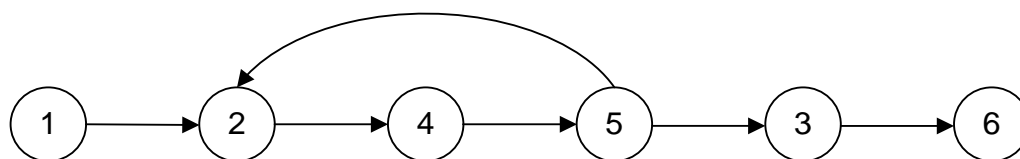


Рисунок 3.5. Пример для правила 3.4

В описанном примере номера в кружках означают номер элемента в технологическом процессе. Итерация строится между элементами с номерами 2, 4, 5. Переход к элементу с номером 3 происходит при выполнении условий выхода из итерации.

Правило 3.5: *ЕСЛИ* количество исходящих ребер у элемента e_i в M больше 1 *И* для элементов e_n, e_m, e_k имеет место $\text{edge}(e_i, e_m), \text{edge}(e_i, e_n)$,

$\text{edge}(e_m, e_n), \text{edge}(e_n, e_m), \text{edge}(e_n, e_k), \text{edge}(e_m, e_k)$, **ТО** для элементов e_n, e_m строится параллельный маршрут.

Правило 3.6: *ЕСЛИ* количество исходящих ребер у элемента e_i в M больше 1 **И** для элементов e_n, e_m, e_p, e_k имеет место $\text{edge}(e_i, e_n), \text{edge}(e_i, e_p), \text{edge}(e_n, e_m), \text{edge}(e_p, e_m), \text{edge}(e_m, e_p), \text{edge}(e_m, e_n), \text{edge}(e_p, e_k), \text{edge}(e_n, e_k)$, **ТО** для элементов e_i, e_n, e_m, e_p строится конкурентный маршрут.

Правило 3.7: *ЕСЛИ* количество исходящих ребер у элемента e_i больше 1 **И** для элементов e_j, \dots, e_k , следующих за e_i в ЧТП не удалось применить правила П4 – П6, **ТО** для элементов e_j, \dots, e_k строится условный маршрут.

Далее каждый путь в условном маршруте строится путем применения к нему правил 3.2-3.7 до их точки слияния. Точка слияния находится путем применения правила 3.8.

Правило 3.8: *ЕСЛИ* у элемента e_i в описании технологического процесса число входящих ребер больше 1, **ТО** e_i является точкой слияния маршрутов, завершающихся этими ребрами.

Правила применяются в соответствии с выбранной стратегией управления, которая, в общем виде, описана выше.

В результате применения правил будет построена обобщенная схема медицинского технологического процесса. После синтеза МТП необходимо решить вопрос его персонализации.

3.2. Персонализация медицинских технологических процессов

Задача персонализации заключается в следующем [108]:

- на основе схем лечения K пациентов строится общая схема лечения конкретной нозологической формы всех больных с близкими состояниями;
- для вновь поступившего пациента Z , взяв за основу за основу построенную обобщенную схему и используя его индивидуальные особенности, построить схему лечения для пациента Z .

Продemonстрируем предложенный алгоритм на примере диагностики и лечения пациентов с болью в спине. Пусть имеются схемы лечения пациентов X и Y, изображенные на рисунке 3.6.

Особенности состояния пациента X:

- боль в поясничном отделе позвоночника;
- наличие дискордикулярного конфликта;

и особенности состояния пациента Y:

- боль в поясничном отделе позвоночника;
- возникновение осевой боли при нагрузках;
- дегенеративное поражение межпозвонкового диска.

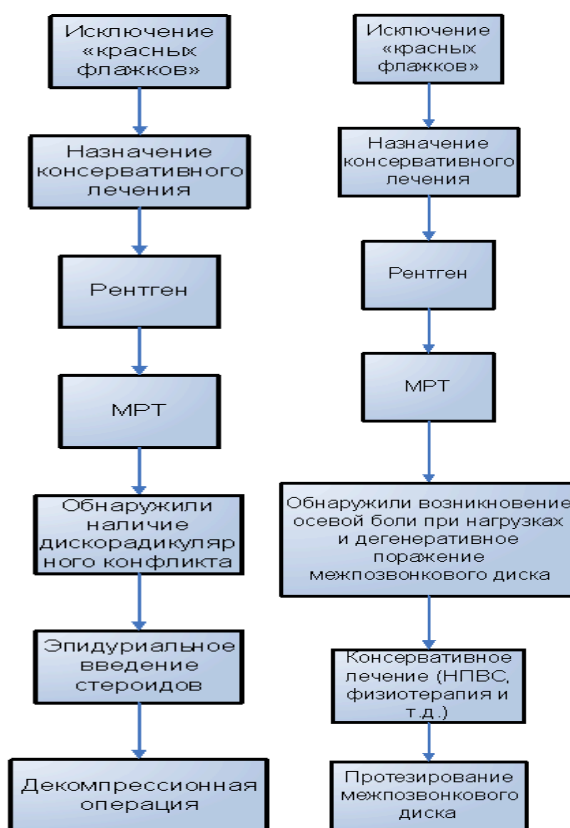


Рисунок 3.6. Схемы лечения пациентов X и Y.

Выполнив действие абстрагирования этих схем лечения, получим следующую общую схему диагностики и лечения пациентов с болью в спине группы пациентов с близкими состояниями, изображенную на рис. 3.7.

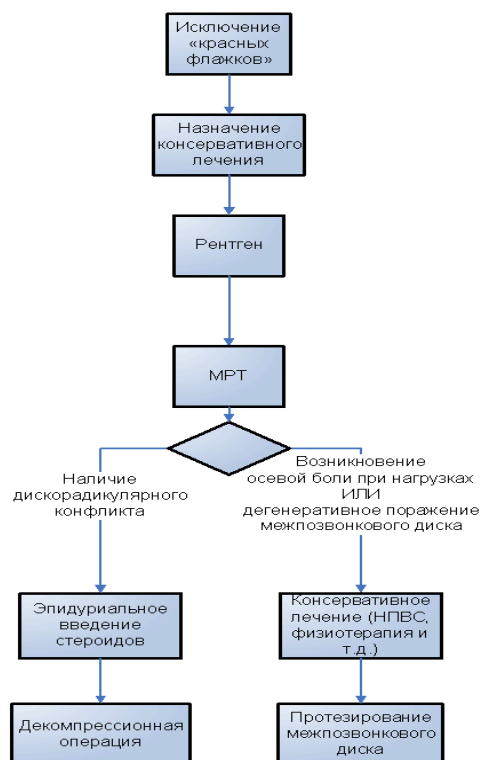


Рисунок 3.7. Общая схема диагностики и лечения пациентов с болью в спине группы пациентов с близкими состояниями

Взяв теперь за основу эту схему и учитывая персональные особенности пациента Z, а именно:

- боль в поясничном отделе позвоночника;
- наличие дискордикулярного конфликта;
- боль более шести недель;
- затруднения при мочеиспускании;

получим следующую схему лечения пациента Z, изображенную на рис. 3.8.

Для выполнения адаптации обобщенной схемы медицинского технологического процесса под индивидуальные особенности пациентов необходимо построить условия выбора маршрута в точках ветвления, которые на рис. 3.13 изображены ромбиком.

Эту задачу можно сформулировать следующим образом: даны обобщенная схема технологического процесса и множество примеров частных технологических процессов с описаниями состояний пациентов в каждый момент времени исполнения ТП, необходимо в точках ветвления (в условных и итеративных маршрутах) определить условия выбора маршрута в

ТП. Выбор маршрута можно записать в виде правил, а решение задачи построения этих правил сводится к решению задачи классификации, где каждому классу соответствует конкретный маршрут после точки ветвления.

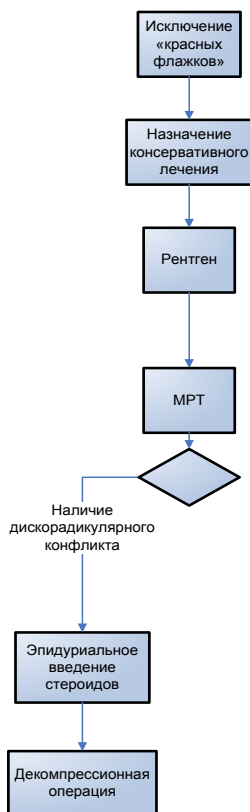


Рисунок 3.8. Схема лечения пациента Z

Для решения подобного рода задач, как в области интеллектуального анализа данных, так и в искусственном интеллекте разработано достаточно много алгоритмов классификации [71, 72, 114–121]. Эти алгоритмы используют множества правил, байесовские и нейронные сети, деревья решений и др. В этом разделе описывается применение алгоритма последовательного порождения правил AQ[122-124] для решения задачи классификации и проводится оценка результатов его работы. AQ-алгоритм является достаточно простым, не использует статистические функции, не зависит от типов входных данных и дает полное и непротиворечивое описание рассматриваемой ситуации. Недостатком его является сильная зависимость от примеров, с которыми он работает, и, в некоторых случаях, получение довольно громоздких правил.

AQ-обучение основывается на идее постепенного покрытия обучающих данных с помощью последовательно порождаемых правил и используется целым семейством алгоритмов. Суть метода заключается в поиске множества правил (конъюнкций пар атрибут-значение или, в общем случае, произвольных предикатов), которое покрывает все положительные (+) примеры и ни одного отрицательного (–) примера. Вместо разбиения множества примеров AQ-алгоритм шаг за шагом обобщает описания выбранных положительных примеров, называемых опорными примерами.

Собственно алгоритм имеет следующий вид [122, 124]:

- 1) разделить все примеры на подмножества *PE* положительных примеров и *NE* отрицательных примеров;
- 2) выбрать из *PE* случайным образом или по каким-то соображениям один пример, который будет считаться опорным примером;
- 3) найти множество максимально общих правил, характеризующих опорный пример. Предел обобщения определяется множеством *NE*: обобщенное описание опорного примера не должно удовлетворять ни одному объекту из *NE*. Полученное таким образом множество правил называется опорным множеством;
- 4) используя некоторый *критерий предпочтения*, выбрать лучшее правило в опорном множестве;
- 5) если это правило, вместе с ранее порожденными таким образом правилами, покрывает все объекты из *PE*, то конец. Иначе — найти другой опорный пример среди неохваченных примеров в *PE* и перейти к 3.

Шаг 3 выполняется специальной процедурой порождения опорного множества правил. Критерий предпочтения для выбора правил на шаге 4 определяется характером решаемой задачи. В качестве такого критерия может выступать комбинация различных элементарных критериев, таких как требование максимального числа положительных примеров, описываемых правилом, минимального числа используемых атрибутов, максимальной

оценки степени общности, т.е. отношения числа положительных примеров, покрываемых правилом, к числу всех примеров, минимальных затрат на измерение значений атрибутов и т.п. Кроме того, можно использовать критерии выбора атрибутов, как в обучении деревьям решений, — такие как энтропия, информационная функция полезности и т.д. Алгоритм позволяет также строить множество правил с различными отношениями между отдельными правилами.

В обучающиеся системы, построенные на основе алгоритма AQ, можно интегрировать любые базовые знания, т.к. они часто могут быть представлены в виде системы правил. Более того, AQ не накладывает ограничения на типы знаний. Они могут быть числовыми, интервальным, символьными и др.

Проиллюстрируем работу алгоритма на примере определения степени обострения бронхиальной астмы. Разумеется, пример будет носить модельный характер и предназначен исключительно для иллюстрации процесса обнаружения правил.

В качестве обучающей выборки возьмем примеры степеней обострения бронхиальной астмы: в таблице 3.1 примеры 1-3 соответствуют тяжёлому обострению, пример 4 – обострению средней степени и пример 5 – обострению легкой степени. Каждый пример задан набором признаков и их значений.

Для простоты возьмём четыре признака (хотя в реальности их должно быть не менее девяти): пульс, чдд, сознание и наличие или отсутствие свистящих хрипов.

Таблица 3.1. Примеры степеней обострения бронхиальной астмы.

Признаки Примеры	Пульс	ЧДД	Сознание	Свистящие хрипы
П1	125	30	Возбужд	Да
П2	130	25	Возбужд	Да

П3	120	30	Возбужд	Да
П4	120	25	Возбужд	Да
П5	110	18	N	Да

Каждая строка соответствует одному вектору значений признаков. Предположим, критерий предпочтения рекомендует выбирать правила, охватывающие максимально возможное число положительных примеров, а условия правил могут пересекаться друг с другом. Программа, ведущая к выявлению правил из приведенных примеров, будет состоять из следующих шагов:

- 1) выбрать первый опорный пример: П1;
- 2) для построения опорного множества примера П1 (т.е. множества максимально общих описаний П1), начать с построения множества всех описаний П1, которым не удовлетворяют отрицательные примеры.

Для этого обобщим какой-либо признак опорного примера, например, *Пульс* получив $\text{Пульс} = 120 \vee 125 \vee 130$, т.е. правило

R1 таково: ЕСЛИ: $\text{Пульс} = (120 \vee 125 \vee 130) \& (\text{ЧДД} = 30) \& (\text{Сознание} = \text{Возбужден}) \& (\text{Свистящие хрипы} = \text{Да})$,

ТО Обострение тяжелое

Далее займёмся вторым атрибутом и попытаемся его обобщить.

Однако, попытки обобщения, т.е. расширения его области значений, приводят к правилу

$R1'$: $\text{Пульс} = (120 \vee 125 \vee 130) \& (\text{ЧДД} = 25 \vee 30) \& (\text{Сознание} = \text{Возбужден}) \& (\text{Свистящие хрипы} = \text{Да})$, которое покрывает пример 4, который не относится к тяжелому обострению бронхиальной астмы. Поэтому правило $R1'$ мы вынуждены отбросить. Однако, оставшееся правило $R1$ покрывает примеры 1 и 3. Остался «неохваченным» пример 2.

Возьмем П2 в качестве следующего опорного примера и попытаемся его обобщить.

Попытка обобщения первого признака примера П2 приводит к тому, что полученное правило покрывает четвертый пример, чего нельзя допустить, поэтому займемся вторым признаком и получим правило

R2: (Пульс = 130), (ЧДД = $25 \vee 30$) & (Сознание = Возбужден) & (Свистящие хрипы = Да) или, в иной записи:

ЕСЛИ (Пульс = 130) & (ЧДД = $25 \vee 30$) & (Сознание = Возбужден) & (Свистящие хрипы = Да),

ТО Обострение тяжелое.

Итак, поскольку неохваченных примеров больше не осталось, то два правила

R1: ЕСЛИ: Пульс = ($120 \vee 125 \vee 130$) & (ЧДД=30) & (Сознание = Возбужден) & (Свистящие хрипы = Да),

ТО Степень обострения = Тяжелое и

R2: ЕСЛИ (Пульс = 130) & (ЧДД = $25 \vee 30$) & (Сознание = Возбужден) & (Свистящие хрипы = Да),

ТО Степень обострения = Тяжелое

дают полное и непротиворечивое описание представленной в таблице 3 ситуации.

Для выявления условий в точках ветвления AQ-алгоритм используется следующим образом. Пусть даны P – список точек ветвления в условных и итеративных маршрутах, $Rules$ – множество правил, определяющих направления движения в каждой точке ветвления, отображение $fr:P \rightarrow Rules$, $rule^p$ –список правил, определяющих выбор пути движения в технологическом процессе для точки p , $PosStates$ –список состояний, характеризующих положительные примеры, $NegStates$ –список состояний, характеризующих отрицательные примеры. При построении правил выбора маршрута в точках ветвления AQ-алгоритм использует дополнительные знания из стандартов лечения и протоколов ведения больных, описанные в онтологической форме.

Алгоритм 3.7 (Алгоритм решения задачи определения условий выбора направления движения в условных и итеративных маршрутах):

- 1) для каждой точки ветвления $p \in P$ массив $paths$ записываются все пути, ведущие из p до точки их слияния, $states$ – все состояния пациентов, в момент нахождения процесса лечения пациента в точкер, $rule^p = 0$;
- 2) если в $paths$ нет необработанных маршрутов, то в $Rules$ добавляется $rule^p$, переход к шагу 1;
- 3) иначе $path =$ необработанный маршрут из $paths$, $PPaths$ – все частные ТП из $\{M^k\}$, включающие в себя $path$, в $PosStates$ записываются все состояния из $states$, связанные с частными ТП из $PPaths$, в $NegStates$ записываются остальные состояния из $states$;
- 4) в $rule^p$ добавляется результат работы алгоритма AQ, для которого $PosStates$ – положительные примеры, $NegStates$ – отрицательные, переход к шагу 2;

Алгоритм завершит свою работу, когда все точки ветвления $p \in P$ будут обработаны. В результате работы алгоритма во всех точках ветвления в условных и итеративных маршрутах будут определены правила, определяющие условия выбора пути движения по маршруту технологического процесса. Блок-схема алгоритма представлена на рисунке 3.9.

Спецификой медицинских задач является то, что медицинские данные не только разнородны, но и, как правило, не имеют представительной выборки. При решении задачи автоматического построения условий выбора направления движения в точках ветвления маршрутов, основным требованием является то, что построенные правила должны быть очень хорошего качества даже в случаях, когда выборки очень малы. Проведем оценку результатов работы AQ-алгоритма в сравнении с алгоритмами

классификации BayesNet (Байесовский классификатор) [125], J48graft (алгоритм C4.5, использующий деревья решений) [126] и J48rip (JRipper) [127].

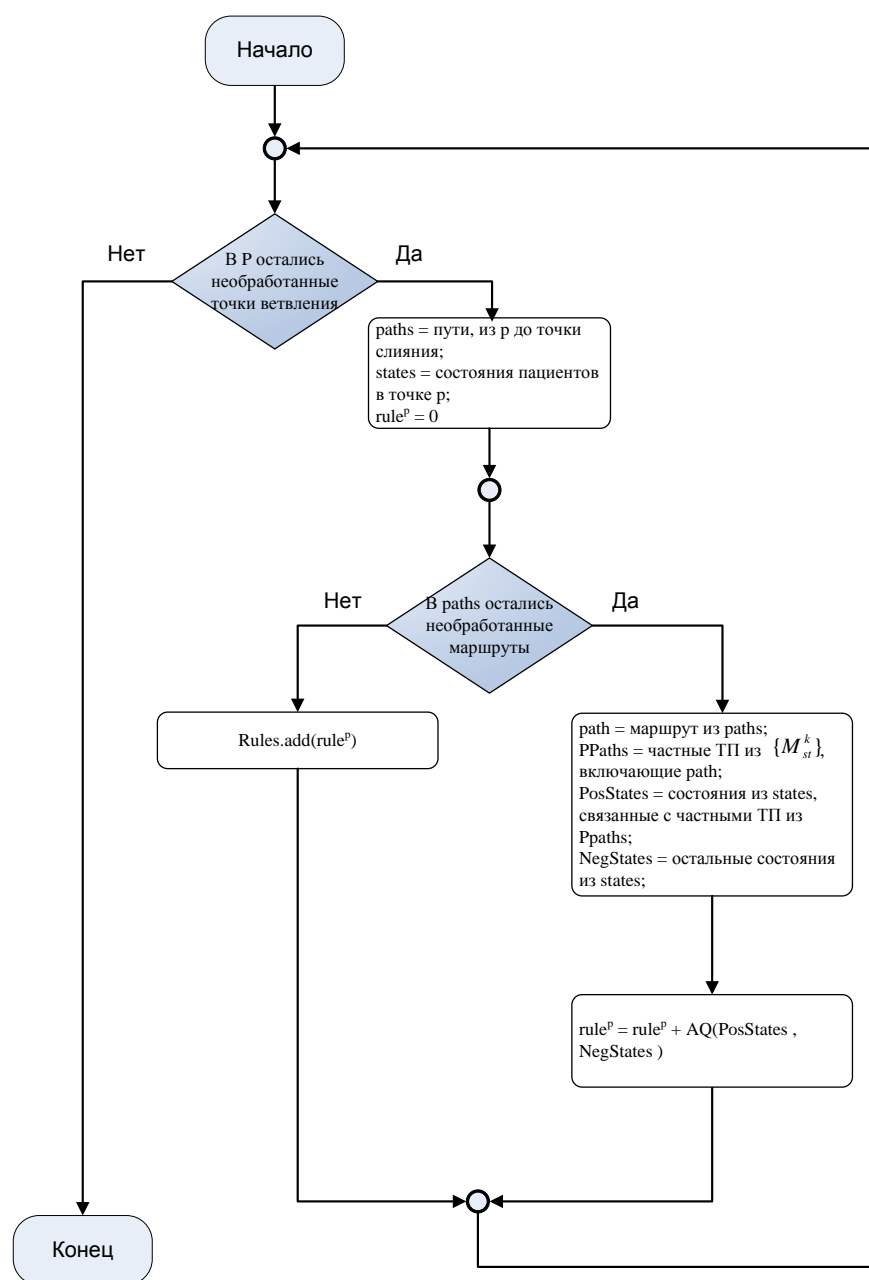


Рисунок 3.9. Блок-схема алгоритма решения задачи определения условий выбора направления движения в условных и итеративных маршрутах

Оценка результатов проводилась путем расчета следующих метрик: чувствительность (Sensitivity) [128], специфичность (Specificity) [128], F-мера [129-130], точность (Precision) [129], достоверность (Accuracy) [128, 131, 132].

Чувствительность рассчитывается по следующей формуле:

$$Sensitivity_{a_i} = \frac{na_i}{Na_i},$$

где na_i – количество объектов из тестовой выборки, верно отнесенных к классу a_i , Na_i – общее число объектов из тестовой выборки, принадлежащих классу a_i .

Специфичность рассчитывается по следующей формуле:

$$Sensitivity = \frac{a}{a + c},$$

где a – число объектов правильно отнесенных к классу (например, пациент болен и тест дал положительный результат), c – число объектов неверно не отнесенных к классу (например, пациент болен, а тест дал отрицательный результат).

F-мерарассчитывается по следующей формуле:

$$F = 2 \times \frac{Precision \times Sensitivity}{Precision + Sensitivity}$$

где точность рассчитывается по следующей формуле:

$$Precision_{a_i} = \frac{n}{m},$$

где n – число примеров были отнесены к некоторому классу $a_i \in A$ верно, а m – общее число примеров, отнесенных к классу a_i .

Достоверность рассчитывается по следующей формуле:

$$Accuracy = \frac{\sum_{i=1}^M m_i}{N},$$

где N – общее число примеров, M – число классов $a_i \in A$, m_i – число объектов, правильно отнесенных к классу a_i .

Оценка результатов работы AQ-алгоритма проводилась на трех наборах данных. Два набора были взяты из базы данных UCI [133], предназначенной для оценки работы методов машинного обучения. Третий набор данных был предоставлен Медицинским центром Центрального банка

Российской Федерации. Для проведения экспериментов была взята реализация AQ-алгоритма AQ21 [134]. Сравнение проводилось с тремя алгоритмами классификации, реализованными в библиотеке WEKA [135-137]: Байесовский классификатор BayesNet, J48graft (алгоритм C4.5, использующий деревья решений, восьмая версия), JRip (JRipper). Обучение сети Байеса осуществлялось без отсутствующих значений атрибутов, поиск узлов сети проводился с помощью жадного алгоритма K2 [138, 139]. J48graft и JRip запускался с настройками, установленными в WEKA по умолчанию.

Первый набор данных предназначен для решения задачи диагностики сахарного диабета по восьми атрибутам. В качестве атрибутов выступают персональная информация о пациенте (например, возраст, пол и др.) и результаты медицинских исследований (например, кровяное давление, индекс массы тела, результаты теста толерантности к глюкозе и др.). Все атрибуты числовые и интервальные, нет атрибутов с отсутствующими значениями. Пациенты разбиты на два класса: класс 1 – больные диабетом и класс 2 – нет. Всего примеров 768 из них 500 относятся к классу 2 и 268 – к классу 1. Выборка была разбита на два множества: обучающее и тестовое. Мощность обучающего множества равна 508, тестового – 260. Положительными примерами для AQ-алгоритма являются примеры из класса 1, отрицательными – примеры из класса 2. В результате работы AQ-алгоритма было сформировано 35 правил, максимальное число условий в правиле 8, минимальное 4, в среднем в правиле 6 условий. Максимальное число положительных примеров, покрываемых правилом равно 27, минимальное – 2. Оценка результатов работы AQ-алгоритма и в сравнении с результатами работы других классификаторов представлены в таблице 3.2.

Таблица 3.2. Оценка и сравнение результатов работы AQ-алгоритма (пример 1)

Алгоритм	Результат	Класс примера	Чувствительность /	Точность	F-мера	Достоверность
----------	-----------	---------------	--------------------	----------	--------	---------------

	работы алгори тма	Клас с 1	Класс 2	Специфичнос ть			ть
AQ	Класс 1	49	28	0,583	0,636	0,608	0,758
	Класс 2	35	148	0,841	0,809	0,825	
BayesNet	Класс 1	59	28	0,702	0,678	0,69	0,796
	Класс 2	25	148	0,841	0,855	0,848	
J48graft	Класс 1	70	42	0,833	0,625	0,714	0,785
	Класс 2	14	134	0,761	0,905	0,827	
JRip	Класс 1	53	23	0,631	0,697	0,663	0,792
	Класс 2	31	153	0,869	0,832	0,85	

В этом примере AQ-алгоритм незначительно уступает другим классификаторам. Но, причем по показателю специфичности он уступил только одному классификатору.

Второй пример, на котором проводилось тестирование, относится к функционированию щитовидной железы. Задача заключается в том, чтобы выявить есть ли у пациента, обратившегося в клинику, признаки гипотиреоза. Данные были взяты из базы данных UCI [133]. Авторы, собравшие эти данные, разделили данные на обучающую и тестовую выборки. Размер обучающей выборки 3772 примера, тестовой – 3428. Примеры разделены на три класса: класс 1 – пациенты без признаков гипотиреоза (nothyroid), класс 2 – гиперфункция (hyperfunction), класс 3 – субнормальное функционирование (subnormal functioning). Количество атрибутов – 21, из них 15 атрибутов бинарные, остальные интервальные; все атрибуты числовые, нет атрибутов с отсутствующими значениями. Особым требованием к алгоритмам классификации является то, что они должны правильно классифицировать значительно больше 92% всех примеров из тестовой выборки. В результате работы AQ-алгоритма было сформировано в общей сложности 6 правил, максимальное число условий в правиле 7,

минимальное 2, в среднем в правиле 5 условий. Из двадцати одного атрибута AQ-алгоритм для принятия решения в правилах оставил только девять. В условиях правил присутствуют все интервальные атрибуты, бинарные атрибуты почти все не были задействованы. В таблице 3.3 представлены результаты работы AQ и других алгоритмов на втором примере и рассчитаны критерии качества работы этих алгоритмов.

Таблица 3.3. Оценка и сравнение результатов работы AQ-алгоритма (пример 2)

Алгоритм	Результат работы алгоритма	Класс примера			Чувствительность	Точность	F-мера	Достоверность
		Класс 1	Класс 2	Класс 3				
AQ	Класс 1	3159	1	2	0,994	0,999	0,996	0,994
	Класс 2	12	176	0	0,994	0,936	0,964	
	Класс 3	7	0	71	0,972	0,91	0,94	
BayesNet	Класс 1	3140	42	0	0,988	0,987	0,987	0,977
	Класс 2	9	135	0	0,763	0,938	0,841	
	Класс 3	29	0	73	1	0,716	0,834	
J48graft	Класс 1	3161	1	2	0,995	0,999	0,997	0,994
	Класс 2	8	176	0	0,994	0,951	0,972	
	Класс 3	9	0	71	0,973	0,899	0,934	
JRip	Класс 1	3156	1	0	0,993	1	0,996	0,993
	Класс 2	11	176	0	0,994	0,941	0,967	
	Класс 3	11	0	73	1	0,869	0,93	

Расчеты показали, что результаты работы AQ алгоритма удовлетворяют не только требованиям качества, но и почти по всем параметрам не хуже, а по многим наиболее точным критериям лучше, результатов работы других алгоритмов.

Следующий пример призван оценить работу AQ-алгоритма на малых выборках. Речь идет об оценке состояния пациентов с раком предстательной железы. Данные были предоставлены Медицинским центром Центрального банка России. Размер выборки составил 97 пациентов. Пациенты были разбиты на три класса, для оценки состояния использовались 28 атрибутов. Все значения атрибутов числовые. Есть атрибуты бинарные, интервальные и принимающие значения из некоторого множества. Множество примеров было разбито на два подмножества: обучающее, размер которого составил 65 примеров, и тестовое, размер которого 32 примера. В результате работы AQ-алгоритма было сформировано в общей сложности 7 правил, максимальное число условий в правиле 7, минимальное 2, в среднем в правиле 4 условия. Из двадцати восьми атрибутов AQ-алгоритм в правилах использовал только восемь. В условиях правил присутствуют все интервальные атрибуты, бинарные атрибуты почти все не были задействованы, часть атрибутов, значения которых берутся из некоторого множества, также не были задействованы. Оценка результатов работы AQ-алгоритма и их сравнение с результатами работы других классификаторов представлены в таблице 3.4.

Таблица 3.4. Оценка и сравнение результатов работы AQ-алгоритма (пример 3)

Алгоритм	Результат работы алгоритма	Класс примера			Чувствительность	Точность	F-мера	Достоверность
		Класс 1	Класс 2	Класс 3				
AQ	Класс 1	28	1	0	1	0,966	0,983	0,938
	Класс 2	0	1	0	0,333	1	0,5	
	Класс 3	0	1	1	1	0,5	0,666	
BayesNet	Класс 1	27	3	0	0,964	0,9	0,931	0,875
	Класс 2	1	0	0	0	0	0	
	Класс 3	0	0	1	1	1	1	

J48graft	Класс 1	28	3	1	1	0,875	0,997	0,875
	Класс 2	0	0	0	0	0	0	
	Класс 3	0	0	0	0	0	0	
JRip	Класс 1	28	3	1	1	0,875	0,997	0,875
	Класс 2	0	0	0	0	0	0	
	Класс 3	0	0	0	0	0	0	

В третьем примере AQ-алгоритм превзошел другие алгоритмы по многим показателям, включая очень важный показатель достоверности. Стоит отметить, что алгоритмы BayesNet, J48graft и JRip имеют одинаковую достоверность, но алгоритмы J48graft и JRip смогут правильно классифицировать только объекты первого класса и выделить объекты второго и третьего класса с их помощью невозможно. Алгоритм BayesNet все объекты второго класса относит к первому.

Эксперименты показали, что правила, построенные AQ-алгоритмом, по качеству сопоставимы или превосходят результаты указанных выше алгоритмов.

Для проверки эффективности предложенных в данном разделе алгоритмов были проведены эксперименты.

3.3. Экспериментальные исследования алгоритма синтеза обобщенной схемы МТП

В отличие от анализа результатов алгоритмов классификации метод k-fold cross validation [140] в чистом виде не подходит для анализа результатов алгоритма автоматического синтеза технологических процессов. Это связано с тем, что прецеденты не содержат положительных и отрицательных примеров. Каждый прецедент описывает поведение технологического процесса в конкретном случае и всегда будет положительным. Некоторые исследователи предлагают искусственно вводить «неправильные» примеры. Но этот способ не подходит при оценке реальных процессов.

Для области анализа процессов для оценки результатов используется следующая метрика [46]:

$$f = \frac{1}{2} \left(1 - \frac{m}{c} \right) + \frac{1}{2} \left(1 - \frac{r}{p} \right),$$

где m – количество пропущенных в ходе проверки примера позиций, c – количество позиций, в которые переходила метка, r – количество позиций в которых метка осталась после полной проверки примера, p – количество позиций, в которые метка должна перейти. При $m=r=0$ фитнес функция равна 1.

Физический смысл этой фитнес функции заключается в том, что она оценивает наличие маршрута, в примере из тестовой выборки, в построенной модели технологического процесса. Если ее значение равно 1, это означает, что искомый маршрут полностью имеется в модели технологического процесса. Если значение фитнес функции меньше 1, то не все части искомого маршрута имеются в построенной модели. Если значение ее значение равно 0, то искомый маршрут отсутствует в построенной модели.

Пусть N – число примеров в тестовой выборке, σ_i – i -й пример из тестовой выборки. Тогда фитнес функция для оценки результатов алгоритма автоматического синтеза обобщенной схемы технологического процесса рассчитывается по следующей формуле [46]:

$$F = \frac{1}{2} \left(1 - \frac{\sum_{\sigma_i} m_{\sigma_i}}{\sum_{\sigma_i} c_{\sigma_i}} \right) + \frac{1}{2} \left(1 - \frac{\sum_{\sigma_i} r_{\sigma_i}}{\sum_{\sigma_i} p_{\sigma_i}} \right)$$

При проведении экспериментов результаты алгоритма, предложенного в этой работе и реализованного на языке C#, сравнивались с результатами α -алгоритма, реализованного в пакете ProM версии 6.3 [46]. Эксперименты проводились на трех выборках логов событий. Все эти выборки были разделены на десять подмножеств примеров. Девять подмножество использовались для построения обобщенной модели технологических

процессов; десятое подмножество использовалось для оценки результатов работы алгоритмов.

Первая выборка входит в пакет ProM6.3 и применяется для проверки алгоритмов автоматического синтеза технологических процессов на основе прецедентной информации. Выборка в этом примере очень хорошего качества, создана искусственно и без шумов. Выборка содержит 1104 частных процессов, содержащих более 7000 событий.

Вторая выборка была собрана сотрудниками Технологического университета Эйндховена (Нидерланды) в рамках выполнения проекта CoSeLoG [141]. Выборка содержит реальные процессы по подаче заявки на строительство в муниципалитет. В файле содержится 1434 частных процессов, содержащих 8577 событий. Выборка содержит шумы.

Третья выборка была собрана сотрудниками Технологического университета Эйндховена [142]. В этой выборке содержится 274 процесса лечения злокачественных новообразований в области шейки матки, содержащих, в общей сложности, более 1000 событий. Выборка была предоставлена Голландским академическим госпиталем и содержит реальные лечебно-диагностические процессы. Стоит отметить, что эта выборка содержит много шума.

Оценка результатов алгоритмов проводилась путем применения фитнес функции, осуществляющей поиск маршрута, содержащегося в примере из тестовой выборки, в построенной модели технологического процесса. В таблицах 3.5 – 3.7 представлены результаты оценки работы алгоритмов.

Таблица 3.5. Результаты оценки на первой выборке

Алгоритм	Количество примеров	Размер обучающей выборки	Размер тестовой выборки	Шумы	Значение фитнес функции
Использующий матрицы смежности	1104	994	110	Нет	0,93
α -алгоритм	1104	994	110	Нет	0,93

Таблица 3.6. Результаты оценки на второй выборке

Алгоритм	Количество примеров	Размер обучающей выборки	Размер тестовой выборки	Шумы	Значение фитнес функции
Использующий матрицы смежности	1434	1291	143	Есть	0,92
α -алгоритм	1434	1291	143	Есть	0,33

Таблица 3.7. Результаты оценки на третьей выборке

Алгоритм	Количество примеров	Размер обучающей выборки	Размер тестовой выборки	Шумы	Значение фитнес функции
Использующий матрицы смежности	274	247	27	Сильно зашумлен	0,77
α -алгоритм	274	247	27	Сильно зашумлен	0,015

Как показали результаты экспериментальных исследований, алгоритм предложенный в данной работе, работает не хуже α -алгоритма на хорошей выборке и превосходит по качеству работы на зашумленных выборках. При проведении экспериментальных исследований необходимо было показать, что предложенный в работе алгоритм на зашумленных данных не только работает лучше α -алгоритма, но значение фитнес функции, при оценке результатов, превышает 0.6. В области анализа процессов считается, что алгоритм автоматического построения технологического процесса на основе прецедентной информации работает очень хорошо, если оценка результатов его работы превышает пороговое значение, равное 0.6. Эксперименты показали, что результаты работы предложенного в данной работе алгоритма не только превосходят результаты работы α -алгоритма, но и значение фитнес функции предложенного алгоритма значительно превышает 0.6.

Описание основных функций программной реализации алгоритмов синтеза технологических процессов представлено в приложении В.

3.4 Выводы

В главе предложен алгоритм синтеза персонализированных медицинских технологических процессов на основе прецедентной информации. В этом алгоритме отсутствует проблема петель. Решена проблема дублирования задач, т.к. при синтезе модели, технологический процесс оценивается не в локальной его точке, а в целом. Проблема принудительного выбора решается путем применения AQ-алгоритма построения условий выбора направления движения в точках ветвления технологического процесса. Кроме вышеперечисленных проблем, в предложенном алгоритме заложена функция устранения тупиков и шумов в построенном технологическом процессе. Проведенные эксперименты показали эффективность описанного алгоритма автоматического построения обобщенной схемы технологического процесса.

используется для управления технологическими процессами и их персонализацией.

Любой процесс лечения строится на основе клинических рекомендаций и медицинских стандартов. Однако клинические рекомендации и стандарты представляют собой текст, написанный на естественном языке, который необходимо перевести в некоторую формальную модель и записать в базу знаний. Это необходимо для того, чтобы в дальнейшем этим процессом можно было управлять и анализировать его. Формализацией МТП занимается подсистема работы с базой знаний (БЗ) системы поддержки технологических процессов (СПТП). Стоит отметить, что в задачу подсистемы работы с БЗ входит не только формализация технологических процессов, но и адаптация их к конкретному медицинскому учреждению. После того, как технологический процесс лечения заболевания был построен, его необходимо протестировать прежде, чем он будет применяться для лечения больных. Этой задачей занимается подсистема тестирования технологических процессов.

Если технологический процесс лечения заболевания удачно прошёл тестирование, то он с помощью подсистемы синхронизации и редактирования добавляется в базу процессов лечения заболеваний. Иначе, этот процесс отправляется на доработку.

Кроме медицинской информационной системы в современных клиниках установлены другие компьютерные системы, предназначенные для решения различных задач. Это могут быть системы лекарственных назначений, поддержки принятия решений, радиочастотной идентификации, справочники лекарственных препаратов, медицинские регистры и т.п. Очень часто эти системы никак не связаны между собой и врачу, чтобы с ними работать, приходится загружать каждую систему отдельно. Кроме того, что это очень неудобно, практикующему врачу иногда приходится дублировать информацию о пациенте в этих системах. Задача системы, осуществляющей поддержку медицинских технологических процессов, наладить

взаимодействие системы поддержки МТП с этими системами. Решением этой задачи занимается подсистема синхронизации со сторонними приложениями. Основной задачей подсистемы синхронизации является реализация программных коннекторов к другим системам, используемым в медицинских технологических процессах. После того, как коннектор реализован и проверен, он внедряется в подсистему управления МТП через подсистему конфигурирования.

Подсистема управления медицинскими технологическими процессами является ключевым элементом системы поддержки МТП. В ее задачи входят поиск в базе ТП подходящего технологического процесса лечения заболевания в зависимости от заболевания больного, запуск найденного процесса, контроль его выполнения, анализ результатов его выполнения и многое другое.

Рассмотрим простой пример. В лечебном учреждении (а именно, стационаре) имеется МИС, система назначений (лекарственных и нелекарственных), база технологических процессов лечения заболеваний в виде медицинских технологических карт и др. Каждый процесс лечения разбит на этапы, каждый из которых длится в течение некоторого времени. Задача системы поддержки технологических процессов осуществить сбор информации со всех установленных в медицинском учреждении систем, подбирать подходящие ТП, проверять качество их исполнения. Стоит отметить, что предлагаемый пример рассматривается в очень упрощенном виде. В рассматриваемом случае алгоритм работы подсистемы управления технологическими процессами состоит из следующих шагов:

- 1) зафиксировать поступление нового пациента;
- 2) проверить, прошел ли пациент консультацию с врачом приемного отделения;
- 3) получить из МИС диагноз пациента;
- 4) подобрать подходящий технологический процесс;
- 5) уведомить лечащего врача о составлении плана лечения;

- 6) после завершения текущего этапа технологического процесса проверить соответствие проведенного лечения плану;
- 7) если лечение не соответствует, то уведомить об этом лечащего врача;
- 8) сохранить результаты анализа проведенного этапа технологического процесса;
- 9) получить из МИС текущий диагноз пациента;
- 10) если диагноз изменился, то переход к шагу 4, иначе переход к шагу 5.

Работа подсистемы управления с текущим пациентом заканчивается при завершении его лечения в стационаре.

Реализация алгоритма управления медицинскими технологическими процессами осуществляется с помощью подсистемы редактирования и проверки процесса управления лечением.

Но кроме онтологического подхода для описания знаний в области медицины и психологии очень хорошо себя зарекомендовали неоднородные семантические сети [71].

4.3. Архитектура системы поддержки технологических процессов в области практической психологии

На основе анализа процессов оказания помощи клиентам в области психологии была разработана следующая архитектура системы поддержки этих процессов (рисунок 4.2).

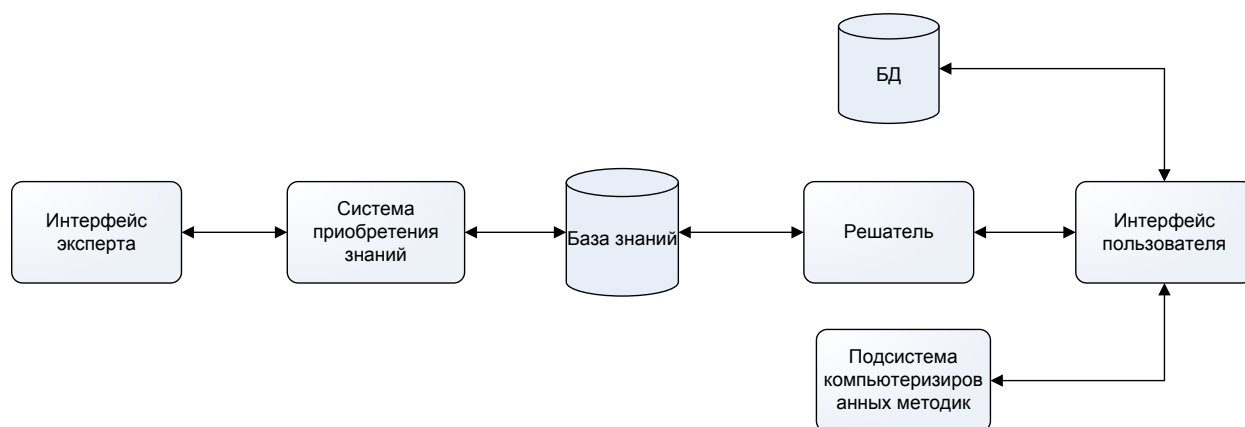


Рисунок 4.2. Архитектура системы поддержки процессов оказания помощи клиентам в области практической психологии

С помощью интерфейса эксперта и системы приобретения знаний эксперты строят базу знаний предметной области [143, 144]. Интерфейс эксперта может быть реализован как Windows или Web приложение. Web-интерфейс эксперта используется в тех случаях, когда эксперты территориально находятся в разных городах или странах. При удаленном построении базы знаний разными экспертами в систему приобретения знаний добавляется возможность объединения баз знаний, построенных разными экспертами. Эксперты могут строить базу знаний как с привлечением инженера по знаниям, так и без него. Это зависит от реализованного алгоритма приобретения знаний от экспертов. Решатель обрабатывает введенную пользователем информацию в соответствии с логикой базы знаний. Интерфейс пользователя служит для ввода данных и отображения результатов работы решателя. С помощью интерфейса эксперта можно работать и с компьютеризированными методиками. Эти методики могут быть реализованы как в самой системе поддержки процессов оказания помощи клиентам, так и просто подключены к ней. Интерфейс пользователя может быть реализован в виде Windows или Web приложения.

Ключевым элементом в системе поддержки технологических процессов в области психологии является база знаний. База знаний содержит формальное представление знаний о предметной области. Существует несколько способов представления знаний: правила, фреймы, семантические

сети, неоднородные семантические сети и др. Ниже предложен способ представления знаний в области практической психологии с помощью неоднородных семантических сетей. Отличительной особенностью предлагаемого подхода является использование новых отношений на множестве концептов.

4.4. Неоднородные семантические сети как средство описания знаний в области практической психологии

Неоднородная семантическая сеть (НСС)[71] – семейство графов, имеющих общее множество вершин; вершинам сопоставлены объекты моделируемой действительности, ребрам – элементы некоторых бинарных отношений на множестве вершин; ребрам же сопоставлены процедуры, предназначенные для проверки корректности сети и порождения различного рода гипотез, повышающих эффективность процесса построения сети.

НСС предназначены для описания плохо структурированных областей, в которых заранее не известен полный набор свойств их индивидов, не полностью известна структура самих индивидов, а знания об индивидах и их взаимосвязях и зависимостях не имеют «готового», завершенного вида, такого, например, который описывается с помощью правил. При этом предполагается, что те связи и зависимости, которые удастся установить, носят локальный характер.

В работах [72, 91, 92] описаны формы отношений, используемые в медицине и других областях.

Для психологии формы отношений, предложенные в [72, 91, 92], дополняются формами отношений, представленными в таблице 4.1.

Таблица 4.1. Дополнительные формы отношений для психологии

N формы	Имя формы
Φ_{13}	А является частью В
Φ_{14}	А включает в себя В

Φ ₁₅	А является показателем В
Φ ₁₆	А является методикой, показателем которой является В
Φ ₁₇	А является свойством методики В
Φ ₁₈	А является методикой, свойством которой является В
Φ ₁₉	А является описанием клиента для В
Φ ₂₀	А является методикой, применимой к клиенту с описанием В
Φ ₂₁	А является объектом диагностики для В
Φ ₂₂	А является методикой, измеряющей объект диагностики В
Φ ₂₃	А является сферой диагностики для В
Φ ₂₄	А является методикой, применимой для сферы диагностики В

По аналогии с предыдущими формами отношений были выделены типы отношений, представленные в таблице 4.3.

Таблица 4.2. Типы отношений для новых форм.

N формы	Имя формы
T ₁₇	А является частью В, В включает в себя А
T ₁₈	А является показателем В, В является методикой, показателем которой является А
T ₁₉	А является свойством методики В, В является методикой, свойством которой является А
T ₂₀	А является описанием клиента для В, В является методикой, применимой к клиенту с описанием А
T ₂₁	А является объектом диагностики для В, В является методикой, измеряющей объект диагностики А
T ₂₂	А является сферой диагностики для В, В является методикой, применимой для сферы диагностики А

Тип T_{17} описывает иерархичность на множестве измеряемых объектов диагностики (ОД) и используется для построения дерева ОД. Например, если специалист хочет узнать что-либо о характере клиента, то понятие «характер» в себя может включать акцентуации характера, черты характера. В качестве черты характера могут выступать агрессивность, застреваемость, нерешительность и др. Стоит отметить, что понятие агрессивности может относиться не только к черте характера, но и к эмоциональной сфере клиента. Там агрессивность может проявляться в виде раздражительности и гневливости. Отношение этого типа транзитивно, не симметрично, не рефлексивно.

Отношение типа T_{18} связывает методики, проводимые с клиентом, с показателями, измеряемыми с помощью выбранных методик. Значения полученных показателей используются для интерпретации результатов проведения методики. Например, с помощью методики Т.В. Чередниковой «Вкус и запах» [93] можно получить значения следующих показателей: соответствие возрастным нормативам, называние простых вкусов, определение интенсивности вкуса, называние сложных вкусов, определение интенсивности запаха, отнесение запахов к обобщенной категории, определение объекта, которому принадлежит предъявляемый запах. Отношение этого типа не транзитивно, не симметрично, не рефлексивно.

Отношения типа T_{19} используются для связи методик с их свойствами. Свойства используются для формирования батареи методик в зависимости от возможностей специалиста. Например, методика Т.В. Чередниковой «Вкус и запах» [93] обладает следующими свойствами: необходимо провести клиническую пробу, при проведении методики используется стимульный материал и нормативы, время проведения методики занимает до 15 минут. Отношение этого типа не транзитивно, не симметрично, не рефлексивно.

Отношения типа T_{20} используются для связи методик с некоторыми параметрами клиента. В качестве таких параметров выступают возраст, форма обследования (индивидуальная, групповая), пол и др. Эти параметры

для формирования батареи методик, применимых для клиента с определенными свойствами. Например, методика Т.В. Чередниковой «Вкус и запах» [93] проводится индивидуально для детей 6-7 лет. Отношение этого типа не транзитивно, не симметрично, не рефлексивно.

Отношения типа T_{21} используются для связи методик с объектами диагностики, которые с помощью них измеряются. В качестве объектов диагностики выступают интеллект, девиации, формы девиаций, семейные отношения, эмоциональная сфера, личностные особенности и др. Например, с помощью методики Т.В. Чередниковой «Вкус и запах» [93] можно измерить такие объекты диагностики как «восприятие вкусовое» и «восприятие обонятельное». Отношение этого типа не транзитивно, не симметрично, не рефлексивно.

Отношения типа T_{22} используются для связи методик со сферами диагностики, для которых они применяются. В качестве сфер диагностики выступают: обследование людей с ограниченными возможностями, общая психодиагностика, патопсихологическая диагностика, психодиагностика девиаций, психодиагностика дошкольников, школьная диагностика и др. Они используются для формирования батареи методик, необходимых специалисту для проведения необходимых испытаний в указанных сферах диагностики. Отношение этого типа не транзитивно, не симметрично, не рефлексивно.

База знаний при использовании НСС представляет собой следующую структуру $KB = \langle S, CL, O, C, CT, P, H \rangle$, где S – множество разделов; CL – множество классов; O – множество объектов; C – множество связей; CT – множество типов связей; P – множество процедур; H – множество состояний базы знаний.

Класс cl – это произвольная пара $\langle N, Ch \rangle$, где N – имя класса, Ch – его характеристика, $Ch = \langle P \rangle$ – множество свойств класса cl , N представляет собой одно или несколько слов в некотором алфавите.

Основными составляющим базы знаний являются объекты, представляющие собой различные понятия предметной области (утверждения, события, факты, результаты наблюдений и т.п.). Объекты являются вершинами семантической сети. Объект представляет собой следующую структуру $o = \langle oN, oCh \rangle$, где oN – имя объекта, $oCh = \{A_1:v_1, \dots, A_n:v_n\}$ – его характеристика. Характеристика объекта задается непустым множеством атрибутов и их значений. Каждому атрибуту A_i (независимо от рассматриваемого объекта) сопоставлен домен (шкала) $D_i = D(A_i)$ – непустая область возможных значений атрибута A_i , значение v_i должно принадлежать D_i . В качестве характеристики объекта могут выступать либо другие объекты, либо атрибуты. Атрибутом называется свойство объекта, имеющее область значений и способное изменяться в зависимости от ситуации в пределах этой области. Большинство объектов характеризуется множеством атрибутов и множествами значений каждого атрибута. Если свойством объекта объявляется другой объект, то к свойствам первого объекта присоединяются все свойства второго. Для каждой конкретной области в базе знаний используется свой набор объектов, их атрибутов, доменов и др.

Объекты в базе знаний сгруппированы в разделы. Объекты могут попадать в разные разделы в зависимости от их функциональной роли. Например, при поддержке процесса психодиагностики объекты делятся на разделы «Клиент», «Объекты диагностики», «Свойства методик» и др. Имя раздела, как правило, играет большую информативную роль. Имя раздела не должно совпадать с именами объектов. Количество разделов, как и объектов не ограничено.

Для того чтобы наличие одних объектов в предметной области могло повлечь порождение либо исключение других, между объектами в базе знаний устанавливаются связи. Связь можно установить между любыми двумя объектами, хотя наиболее важны связи между признаками и гипотезами.

Для системы поддержки технологических процессов в области психодиагностики используются связи между объектами, которым соответствуют следующие высказывания:

- 1) объект 1 входит в понятие объекта 2. Эта связь используется для формирования древовидной иерархии между объектами 1 и 2. Объект 1 находится на уровень ниже объекта 2;
- 2) в понятие объекта 1 входит объект 2. Эта связь используется для формирования древовидной иерархии между объектами 1 и 2. Объект 1 находится на уровень ниже объекта 2;
- 3) объект 1 является показателем для объекта 2. Эта связь соединяет методики и ее показателя;
- 4) объект 1 является свойством объекта 2. Эта связь соединяет методики с ее свойствами;
- 5) объект 1 является описанием клиента для объекта 2. Эта связь соединяет методики с объектами, описывающими характеристики клиента;
- 6) объект 1 является объектом диагностики для объекта 2. Эта связь соединяет методики с объектами диагностики;
- 7) объект 1 является сферой диагностики для объекта 2. Эта связь соединяет методики со сферами диагностики;
- 8) объект 1 является методикой для объекта 2. Эта связь соединяет методики с объектами-признаками.

Для однозначного установления типа отношения связи в базе знаний должны быть установлены как между объектами 1 и 2, так и между объектами 2 и 1.

Исполнение связей происходит путем вызова соответствующих процедур обработки связей.

На рисунке 4.4 изображён фрагмент НСС базы знаний системы «Психодиагностика», входящей в комплекс экспертных систем «Психология» [145, 146, 147]. Система «Психодиагностика» предназначена

для поддержки всего процесса психодиагностики, начиная с подбора диагностических методик и заканчивая интерпретацией результатов использованных методик с рекомендациями по составлению окончательного диагноза.

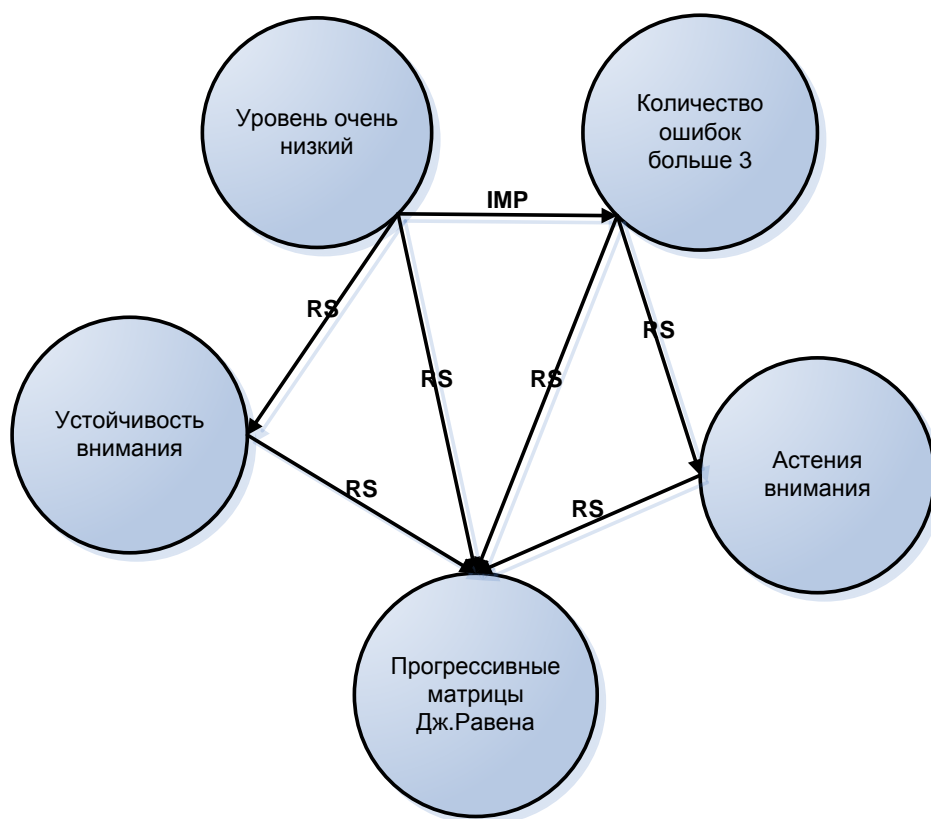


Рисунок 4.4. Фрагмент НСС системы «Психодиагностика»

4.5. Основные структуры данных

Общая структурная схема системы поддержки технологических процессов в области медицины и психологии имеет трехзвенный характер и представлена на рисунке 4.5.

Класс Knowledgeпредназначен для выгрузки в память компьютера базы знаний и содержит функционал для работы с БЗ. Он представляет собой кроссплатформенную библиотеку.

Класс MainForm является основным классом для взаимодействия с интерфейсом пользователя и выступает в роли клиента трехзвенной архитектуры.

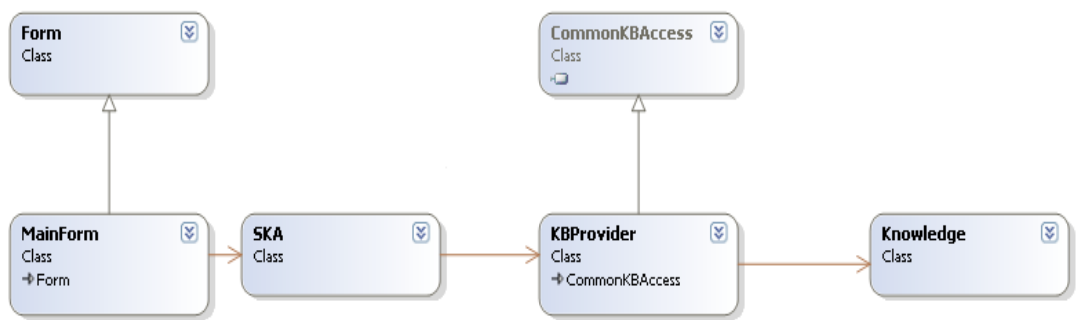


Рисунок 4.5.Общая структурная схема

Классы SKA и KBProvider играют роль сервера приложений, предоставляя классу MainForm и получая от него высокоуровневые объекты для отображения и редактирования. KBProvider реализует взаимодействие с Knowledge. Например, если Knowledge реализован в виде C++ библиотеки, а интерфейс пользователя с помощью .NET, то KBProvider преобразует типы C++ к типам платформы .Net. SKA реализует логику работы системы, осуществляющей манипуляции со знаниями. Например, системы приобретения знаний, поддержки процесса психодиагностики или построения персонализированного плана лечения для пациента.

На рисунке 4.6 представлены основные классы и отношения ассоциации и наследования между ними.

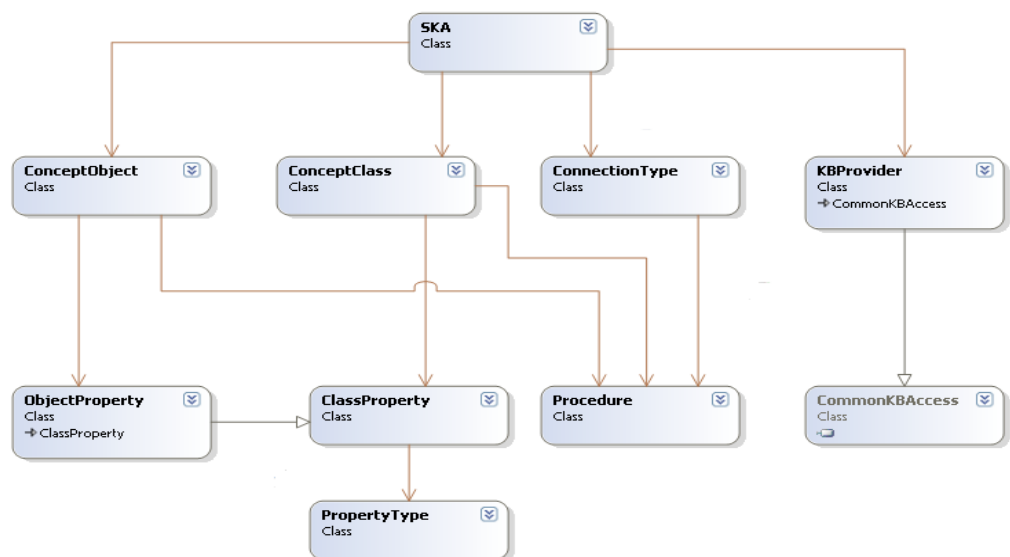


Рисунок 4.6.Основные классы

Поясним каждый из приведенных классов с помощью таблицы 4.3.

Таблица 4.3. Пояснения к основным классам

Имя класса	Пояснение
CommonKBAccess	Предоставляет общие для внешних систем и прикладных решателей функции работы с базой знаний, реализуется в отдельной DLL.
KBProvider	Реализует взаимодействие с Knowledge
ConceptObject	Класс для представления объекта предметной области. Создается и инициализируется классом SKA.
ConceptClass	Служит для представления класса сущностей предметной области.
ConnectionType	Служит для представления типа связи.
ObjectProprety	Класс для представления свойства объекта предметной области. Наследуется от ClassProperty.
ClassProperty	Служит для представления свойства класса сущности предметной области.
PropertyType	Служит для представления типа свойства.
Procedure	Класс для представления процедур.

Классы, предназначенные для работы с пользовательским интерфейсом, приведены на рисунке 4.7.

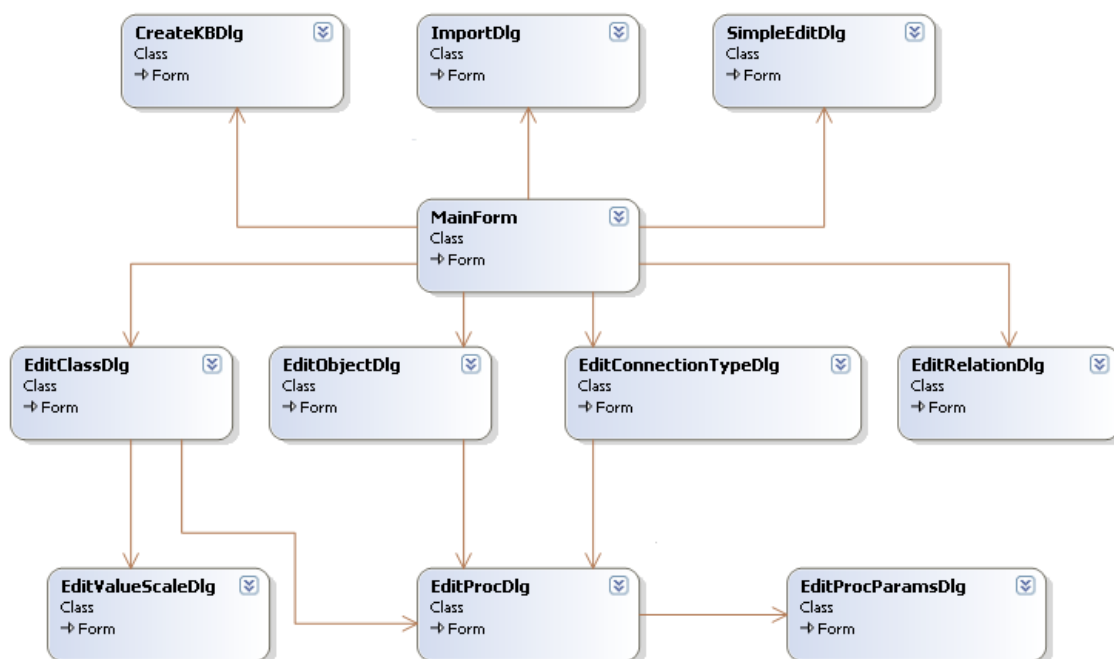


Рисунок 4.7. Классы для работы с пользовательским интерфейсом

Поясним каждый из приведенных классов с помощью таблицы 4.4.

Таблица 4.4. Пояснения к основным классам пользовательского интерфейса

Имя класса	Пояснение
MainForm	Класс главного окна пользовательского интерфейса.
CreateKBDlg	Класс для создания БЗ.
ImportDlg	Класс для импорта БЗ.
SimpleEditDlg	Класс для редактирования названий разделов БЗ
EditClassDlg	Класс для создания и редактирования классов предметной области.
EditObjectDlg	Класс для создания и редактирования объектов предметной области.
EditConnectionTypeDlg	Класс для создания и редактирования типов связей в предметной области.
EditRelationDlg	Класс для создания и редактирования связей в предметной области.

EditValueScaleDlg	Класс для создания и редактирования области значений свойства класса сущности предметной области.
EditProcDlg	Класс для создания и редактирования процедур.
ProcedureOwner	Перечисление возможных обладателей процедур (объект, класс, тип связи).
EditProcParamDlg	Класс для создания и редактирования параметров процедур.

4.6. Выводы

В главе предложен способ представления знаний в области практической психологии с помощью неоднородных семантических сетей. Описаны архитектуры и основные структуры данных программных средств поддержки технологических процессов в области медицины и психологии.

Программные средства поддержки технологических процессов, разработанные на основе предложенных архитектур, могут быть использованы как для поддержки технологических процессов в целом, так и их части. Использование алгоритма автоматического синтеза обобщенной схемы технологических процессов в подсистеме работы с базой знаний позволяет сократить время построения персонализированных технологических процессов лечения различных заболеваний. Применение этого алгоритма в подсистеме анализа технологических процессов позволяет решить следующие задачи:

- оценить качество оказанной медицинской помощи;
- оценить соответствия проведенного лечения клиническим рекомендациям;
- оценить соответствие построенной модели процесса лечения заболевания реальным медицинским технологическим процессам, протекающим в лечебном учреждении;

- строить персонализированные планы лечения пациентов.

Предложенные архитектуры программных средств поддержки технологических процессов в области медицины и психологии использовались при разработке комплекса экспертных систем «Психология» и системы поддержки процесса принятия решения о проведении коронарного вмешательства.

Заключение

Диссертационная работа посвящена изучению методов и алгоритмов поддержки технологических процессов в области медицины и психологии. При описании модели технологического процесса использовалась операторная теория медицинских технологических процессов.

Выполнен анализ методов поддержки медицинских технологических процессов путем моделирования лечебно-диагностических процессов, построения планов лечения и формализации клинических руководств. Выполнен обзор методов поддержки технологических процессов в области психологии путем разработки экспертных систем и компьютеризации психологических методик. Описаны модели технологических процессов в области медицины и психологии. Доказаны утверждения об ассоциативности и не коммутативности последовательного применения операторов. Предложен онтологический подход для описания медицинских технологических процессов. Описан новый алгоритм автоматического синтеза обобщенной схемы технологических процессов на основе прецедентной информации. Проведена экспериментальная оценка разработанного алгоритма.

Основные результаты работы, выносимые на защиту:

- впервые предложено матричное представление медицинских технологических процессов;
- доказаны утверждения об ассоциативности и некоммутативности последовательного применения произвольного числа операторов МТП;
- введена операция покомпонентного сложения матриц смежности экземпляров медицинских технологических процессов в задаче автоматического синтеза обобщенной схемы технологических процессов;

- разработан новый алгоритм построения обобщенной схемы технологического процесса на основе прецедентной информации;
- предложен метод выявления точек ветвления условного маршрута на основе классификации экземпляров технологических процессов;
- разработаны архитектуры и реализованы программные средства автоматического синтеза обобщенных технологических процессов и методы их персонализации.

Предложенные в диссертационной работе методы и алгоритмы были применены для разработки комплекса экспертных систем «Психология» и ряда систем поддержки различных этапов лечебно-диагностических процессов в Медицинском центре Банка России.

Полученные результаты позволяют перейти к решению следующих задач:

- улучшение методов устранения шумов при автоматическом синтезе технологических процессов;
- разработка методов построения медицинских технологических процессов путем разработки методов приобретения знаний от экспертов и использования методов извлечения знаний из текстов;
- разработка методов построения системы управления медицинскими технологическими процессами.

Список литературы

1. Назаренко Г.И., Осипов Г.С. Основы теории медицинских технологических процессов. Ч. 1. – М.: ФИЗМАТЛИТ, 2005. – 144 с.
2. Ю.М. Забродин, В.Э. Пахальян Психодиагностика; под общ.ред. Ю.М. Забродина. – М.: Эксмо, 2010. – 448 с. – (Новейший справочник психолога).
3. Е. Н. Shortliffe. Computer-Based Medical Consultations: MYCIN. Elsevier/North Holland, New York NY, 1976.
4. CASNET [Электронный ресурс] <http://www.casnet.com>
5. DXplain [Электронный ресурс] <http://dxplain.org/dxp/dxp.pl>
6. Doherty J, Noirot LA, Mayfield J, Ramiah S, Huang C, Dunagan WC, Bailey TC. Implementing GermWatcher, an enterprise infection control application. AMIA AnnuSymp Proc.2006:209-13.
7. Glenn Edwards, Paul Compton, Ron Malor, Ashwin Srinivasan, Leslie Lazarus. Peirs: A pathologist-maintained expert system for the interpretation of chemical pathology reports. Pathology. 1993, Vol. 25, No. 1, Pages 27-34
8. Gardner RM, Pryor TA, Warner HR. The HELP hospital information system: update 1998. Int J MedInf. 1999 Jun;54(3):169-82.
9. Darmoni SJ, Massari P, Droy JM, Mahe N, Blanc T, Moiror E, Leroy J. SETH: an expert system for the management on acute drug poisoning in adults. Computer Methods Programs Biomed. 1994 Jun;43(3-4):171-6
10. Кобринский Б.А. Автоматизированные диагностические и информационно-аналитические системы в педиатрии//Русский медицинский журнал. – 1999. т. 7. - №4. с. 35-42.
11. Бураковский В.И., Бокерия Л.А., Газизова Д.Ш., Лищук В.А. и др. Компьютерная технология интенсивного лечения: контроль, анализ, диагностика, лечение, обучение. – М.: НЦ ССХ РАМН, 1995.
12. Silvia Miksch, Yuval Shahar, Werner Horn, Christian Popow, Franz Paky, and Peter Johnson. Time-oriented skeletal plans: Support to design and

- execution. In *Fourth European Conference on Planning (ECP'97)*. Springer, September 24–26 1997
13. Silvia Miksch, Yuval Shahar, and Peter Johnson. Asbru: A task-specific, intention-based, and time-oriented language for representing skeletal plans. In *Proceedings of the 7th Workshop on Knowledge Engineering: Methods & Languages (KEML-97)*. Milton Keynes, UK, Open University, 1997.
 14. Andreas Seyfang, Robert Kosara, and Silvia Miksch. Asbru's reference manual, Asbru version 7.2. Technical Report Asgaard-TR-2000-3, Vienna University of Technology, Institute of Software Technology, 2000.
 15. XMLSchema [Электронный ресурс] <http://www.w3.org/XML/Schema/>
 16. Robert Kosara, Klaus Hammermüller, and Silvia Miksch. Codesigning XML-based languages and classes with pontifex. Technical Report Asgaard-TR-2000-1, Vienna University of Technology, Institute of Software Technology, Vienna, Austria, 2000.
 17. Y. Lee, C. Patel, S. A. Chun, and J. Geller. Compositional knowledge management for medical services on semantic web. In *WWW2004*, May 17–22, 2004, New York, New York, USA, May 2004.
 18. S. Jablonski. Mobile: A modular workflow model and architecture. In *Procs. Fourth International Working Conference on Dynamic Modelling and Information Systems*. Noordwijkerhout, The Netherlands, September 1994.
 19. J. Miller, A. Sheth, K. Kochut, and X. Wang. Corba-based run-time architectures for workflow management systems. *Journal of Database Management, Special Issue on Multidatabases*, vol. 7(1):pp. 16–27, winter 1996.
 20. S. Quaglini, M. Stefanelli, A. Cavallini, G. Micieli, C. Fassino, and C. Mossa. Guideline-based care flow systems. *Artificial Intelligence Medicine*, vol. 20(1):pp. 5–22, August 2000.
 21. S. Quaglini, M. Stefanelli, G. Lanzola, V. Caporusso, and S. Panzarasa. Flexible guideline-based patient careflow systems. *Artificial Intelligence in Medicine*, vol. 22(1):pp. 65–80, 2001.

22. Dogac, E. Gokkoca, S. Arpinar, P. Koksall, I. Cingil, B. Arpinar, N. Tatbul, P. Karagoz, U. Halici, and M. Altinel. Design and implementation of a distributed workflow management system: METUFlow. In A. Dogac, L. Kalinichenko, T. Ozsu, and A. Sheth, editors, NATO-ASI on Workow Management Systems and Interoperability, pp. 60-90. August 1997.
23. J. Warren, G. Beliakov, J. Noone, and H. Frankel. Chronic disease coordinated care planning: flexible, task-centered decision support. In Proceedings of the 32nd Hawaii International Conference on System Sciences (HICSS-32), vol. 4. Maui, Hawaii, January 1999.
24. James R Warren, Joseph T Noone, Brian J Smith, Richard Ruffin, Peter Frith, Berend J van der Zwaag, Gleb V Beliakov, Heath K Frankel and Heather J McElroy. Automated attention flags in chronic disease care planning. MJA. 2001, Vol. 175, Pages 308-312.
25. M. Reichert and P. Dadam. ADEPT flex -supporting dynamic changes of workflows without losing control. Journal of Intelligent Information Systems, vol. 10(2):pp. 93-129, 1998.
26. R. Muller and E. Rahm. Rule-based dynamic modification of workflows in a medical domain. In A. Buchmann, editor, Proceedings of BTW99, pp. 429-448. Springer, Berlin, Freiburg imBreisgau, March 1999.
27. R. Muller. Event-Oriented Dynamic Adaptation of Workows: Model, Architecture, and Implementation. Ph.D. thesis, Fakultat fur Mathematik und Informatik der Universitat Leipzig, 2002.
28. Данс А., Данс Л., Сильвестер М. Безболезненное освоение доказательной медицины. – Wiley, 2008. – 140с.
29. Field, M.J., Lohr, K.H.: Clinical Practice Guidelines: Directions for a New Program. NationalAcademyPress (1990).
30. Martini, P., Kaiser, K., Miksch, S., Easing the Formalization of Clinical Guidelines with a User-tailored, Extensible Agile Model Driven Development (AMDD), *21st IEEE International Symposium on Computer-Based Medical Systems*, 2008, pp.120-125.

- 31.R. Kosara and S. Miksch. Metaphors of Movement: A Visualization and User Interface for Time-Oriented, Skeletal Plans. *Artificial Intelligence in Medicine, Special Issue: Information Visualization in Medicine*, 22(2):111–131, May 2001.
- 32.Kosara, R., Miksch, S., Metaphors of Movement: A Visualization and User Interface for Time-Oriented, Skeletal Plans, *Artificial Intelligence in Medicine, Special Issue: Information Visualization in Medicine*, Vol.22, No.2, 2001, pp. 111–131.
- 33.J. H. Gennari, M. A. Musen, R.W. Ferguson, W. E. Grosso, M. Crub'ezzy, H. Eriksson, N. F. Noy, and S.W. Tu. The Evolution of Prot'eg'e: An Environment for Knowledge-based Systems Development. *International Journal of Human Computer Studies*, 58(1):89–123, 2003.
- 34.Arezzo [Электронный ресурс]
http://www.infermed.com/index.php/arezzo/arezzo_technology
- 35.Peleg, M., Tu, S., Bury, J., Ciccarese, P., Fox, J., Greenes, R., Hall, R., Johnson, P., Jones, N., Kumar, A., Miksch, S., Quaglini, S., Seyfang, A., Shortliffe, E., Stefanelli, M., Comparing Computer-Interpretable Guideline Models: A Case-Study Approach, *Journal of the American Medical Informatics Association* No.10, 2003, pp. 52–68.
- 36.R.N. Siffman, B.T. Karras, A. Agrawal, R. Chen, L. Marengo and S. Math Gem: A proposal for a more comprehensive guideline document model using xml. *Journal of the American Medical Informatics Association*. 7(5), 488-398, 2000.
- 37.V. Svatek and M. Razicka. Step-by-step mark-up of medical guideline documents. *International Journal of Medical Informatics*, 70(2-3), 329-335, 2003.
- 38.Ruzicka, M., Svatek V., Step-by-Step Mark-Up of Medical Guideline Documents, *International Journal of Medical Informatics*, Vol.70, No.2-3, 2003, pp. 329-335.

39. Votruba, P., Miksch, S., Kosara, R.: Facilitating Knowledge Maintenance of Clinical Guidelines and Protocols, in *11th World Congress of Medical Informatics (MedInfo 2004)*.
40. Votruba, P., Miksch, S., Kosara, R.: Tracing the Formalization Steps of Textual Guidelines, in Kaiser, K.; Miksch, S. and Tu, S. (eds.): *Computer-based Support for Clinical Guidelines and Protocols. Proceedings of the Symposium on Computerized Guidelines and Protocols (CGP 2004)*, volume 101 Studies in Health Technology and Informatics. IOS Press, 172-176, 2004.
41. Y. Shahr, O. Young, E. Shalom, A. Mayaffit, R. Moskovitch, A. Hessian, and M. Galperin. DEGEL: A hybrid, multiple-ontology framework for specification and retrieval of clinical guidelines. In M. Dojat, E. Keravnou, and P. Barahona, editors, Proceedings of the 9th Conference on Artificial Intelligence in Medicine in Europe, AIME 2003, volume 2780 of LNAI, pages 122–131, Protaras, Cyprus, 2003. Springer Verlag.
42. R. Agrawal, D. Gunopulos, F. Leymann, Mining process models from workflow logs, in: H. Schek, F. Saltor, I. Ramos, G. Alonso (Eds.), Proceedings of the 6th International Conference on Extending Database Technology (EDBT'98), Lecture Notes in Computer Science, vol. 1377, Springer, 1998, pp. 469–483.
43. J. Cook, A. Wolf, Discovering models of software processes from event-based data, ACM Transactions on Software Engineering and Methodology 7 (1998) 215–249.
44. A. Datta, Automating the discovery of AS-IS business process models: probabilistic and algorithmic approaches, Information Systems Research 9 (1998) 275–301.
45. W.M.P. van der Aalst, 'The application of Petri nets to workflow management', Journal of Circuits, Systems and Computers, 8(1), 21-66, (1998).

46. Wil M. P. van der Aalst, *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, Springer Publishing Company, Incorporated, 2011, 352 p.
47. Барсегян, А. А. Анализ данных и процессов: учеб. пособие / А. А. Барсегян, М. С. Куприянов, И. И. Холод, М. Д. Тесс, С. И. Елизаров. — 3-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2009. — 512 с.: ил.
48. W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.
49. W.M.P. van der Aalst, A.J.M.M. Weijters, L. Maruster, Workflow mining: discovering process models from event logs, *IEEE Transactions on Knowledge and Data Engineering* 16 (2004) 1128–1142.
50. R. Agrawal, D. Gunopulos, F. Leymann, Mining process models from workflow logs, in: H. Schek, F. Saltor, I. Ramos, G. Alonso (Eds.), *Proceedings of the 6th International Conference on Extending Database Technology (EDBT'98)*, Lecture Notes in Computer Science, vol. 1377, Springer, 1998, pp. 469–483.
51. A. Datta, Automating the discovery of AS-IS business process models: probabilistic and algorithmic approaches, *Information Systems Research* 9 (1998) 275–301.
52. J. Cook, A. Wolf, Discovering models of software processes from event-based data, *ACM Transactions on Software Engineering and Methodology* 7 (1998) 215–249.
53. H. Mannila, C. Meek, Global partial orders from sequential data, in: *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '00)*, ACM, New York, NY, USA, 2000, pp. 161–168.
54. G. Schimm, Processminer—a tool for mining process schemes from event-based data, in: S. Flesca, S. Greco, N. Leone, G. Ianni (Eds.), *JELIA, Lecture Notes in Computer Science*, vol. 2424, Springer, 2002, pp. 525–528.

- 55.G. Schimm, Mining exact models of concurrent workflows, *Computers in Industry* 53 (2004) 265–281.
- 56.A.K. Alves de Medeiros, B.F. van Dongen, W.M.P. van der Aalst, A.J.M.M. Weijters, *Process Mining: Extending the Alpha-Algorithm to Mine Short Loops*, BETA Working Paper Series 113, TU Eindhoven, 2004.
- 57.J. Herbst, D. Karagiannis, Workflow mining with InWoLvE, *Computers in Industry* 53 (2004) 245–264.
- 58.B.F. van Dongen, W.M.P. van der Aalst, Multi-phase process mining: aggregating instance graphs into EPCs and Petri nets, in: *Proceedings of the 2nd International Workshop on Applications of Petri Nets to Coordination, Workflow and Business Process Management (PNCWB)*.
- 59.W.Gaaloul, K.Bauna, C. Godart, Towards mining structural workflow patterns, in: K.V.Andersen, J.K.Debenham, R.Wagner (Eds.), *DEXA, Lecture Notes in Computer Science*, vol.3588, Springer, 2005, pp.24–33.
- 60.A.J.M.M. Weijters, W.M.P. van der Aalst, Rediscovering workflow models from event-based data using little thumb, *Integrated Computer-Aided Engineering* 10 (2003) 151–162.
- 61.A.J.M.M. Weijters, W.M.P. van der Aalst, A.K. Alves de Medeiros, *Process Mining with the Heuristics Miner algorithm*, BETA Working Paper Series 166, TU Eindhoven, 2006.
- 62.G.Greco, A.Guzzo, L.Pontieri, D.Sacca, Discovering expressive process models by clustering log traces, *IEEE Transactions on Knowledge and Data Engineering* 18(2006)1010–1027.
- 63.L. Maruster, A.J.M.M. Weijters, W.M.P. van der Aalst, A. van den Bosch, A rule-based approach for process discovery: dealing with noise and imbalance in process logs, *Data Mining and Knowledge Discovery* 13 (2006) 67–87.
- 64.H. Ferreira, D. Ferreira, An integrated life cycle for workflow management based on learning and planning, *International Journal of Cooperative Information Systems* 15 (2006) 485–505.

65. A.K. Alves de Medeiros, A.J.M.M. Weijters, W.M.P. van der Aalst, Genetic process mining: an experimental evaluation, *Data Mining and Knowledge Discovery* 14 (2007) 245–304.
66. A.K. Alves de Medeiros, Genetic Process Mining, Ph.D. Thesis, TU Eindhoven, 2006.
67. C.W. Gunther, W.M.P. van der Aalst, Fuzzy mining—adaptive process simplification based on multi-perspective metrics, in: G. Alonso, P. Dadam, M. Rosemann (Eds.), *Proceedings of the 5th International Conference on Business Process Management, BPM 2007, Lecture Notes in Computer Science*, vol. 4714, Brisbane, Australia, Springer, September 24–28, 2007, pp. 328–343.
68. L. Wen, W.M.P. van der Aalst, J. Wang, J. Sun, Mining process models with non-free-choice constructs, *Data Mining and Knowledge Discovery* 15 (2007) 145–180.
69. E. Lamma, P. Mello, M. Montali, F. Riguzzi, S. Storari, Inducing declarative logic-based models from labeled traces, in: G. Alonso, P. Dadam, M. Rosemann (Eds.), *Proceedings of the 5th International Conference on Business Process Management, BPM 2007, Lecture Notes in Computer Science*, vol. 4714, Brisbane, Australia, Springer, September 24–28, 2007, pp. 344–359.
70. G. Greco, A. Guzzo, L. Pontieri, Mining taxonomies of process models, *Data & Knowledge Engineering* 67 (2008) 74–102.
71. Г.С.Осипов. Методы искусственного интеллекта. М.: Наука, Физматлит, 2011, 295 стр.
72. S. Goedertier, D. Martens, J. Vanthienen, B. Baesens, Robust process discovery with artificial negative events, *Journal of Machine Learning Research* 10 (2009) 1305–1340.
73. L. Wen, J. Wang, W.M.P. van der Aalst, B. Huang, J. Sun, A novel approach for process mining based on event types, *Journal of Intelligent Information Systems* 32 (2009) 163–190.

74. F. Folino, G. Greco, A. Guzzo, L. Pontieri, Discovering expressive process models from noisy log data, in: B. C. Desai, D. Sacca, S. Greco (Eds.), IDEAS, ACM International Conference Proceeding Series, ACM, 2009, pp. 162–172.
75. D. R. Ferreira, D. Gillblad, Discovering process models from unlabelled event logs, in: U. Dayal, J. Eder, J. Koehler, H. A. Reijers (Eds.), Proceedings of the 7th International Conference on Business Process Management, BPM 2009, Lecture Notes in Computer Science, vol. 5701, Ulm, Germany, Springer, September 8–10, 2009, pp. 143–158.
76. J. M. E. M. van der Werf, B. F. van Dongen, C. A. J. Hurkens, A. Serebrenik, Process discovery using integer linear programming, *Fundamenta Informaticae* 94 (2009) 387–412.
77. W. M. P. van der Aalst, V. Rubin, B. F. van Dongen, E. Kindler, C. W. Gunther, Process Mining: A Two-Step Approach using Transition Systems and Regions, BPM-06-30, BPM Center Report, 2006.
78. W. M. P. van der Aalst, V. Rubin, H. M. W. Verbeek, B. F. van Dongen, E. Kindler, C. W. Gunther, Process mining: a two-step approach to balance between underfitting and overfitting, *Software and System Modeling* 9 (2010) 87–111.
79. J. Carmona, J. Cortadella, M. Kishinevsky, New region-based algorithms for deriving bounded Petri nets, *IEEE Transactions on Computers* 59 (2010) 371–384.
80. A. K. Alves de Medeiros, A. J. M. M. Weijters, W. M. P. van der Aalst, Genetic process mining: an experimental evaluation, *Data Mining and Knowledge Discovery* 14 (2007) 245–304.
81. Ebert, J., Engels, G.: Specialization of Object Life Cycle Definitions. *Fachberichte Informatik* 19/95, University of Koblenz-Landau (1997).
82. Kappel, G., Schrefl, M.: Object/Behavior Diagrams. In: Proceedings of the 7th International Conference on Data Engineering, pp. 530–539. IEEE Computer Society, Los Alamitos (1991).

- 83.Schrefl, M., Stumptner, M.: Behavior-Consistent Specialization of Object Life Cycles. *ACM Transactions on Software Engineering and Methodology* 11(1), 92–148(2002).
- 84.Stumptner, M., Schrefl, M.: Behavior Consistent Inheritance in UML. In: Laender, A.H.F., Liddle, S.W., Storey, V.C. (eds.) *Conceptual Modeling - ER 2000*. LNCS, vol. 1920, pp. 527–542. Springer, Heidelberg (2000).
- 85.van der Aalst, W.M.P., Basten, T.: Identifying Commonalities and Differences in Object Life Cycles using Behavioral Inheritance. In: Colom, J.-M., Koutny, M.(eds.) *ICATPN 2001*. LNCS, vol. 2075, pp. 32–52. Springer, Heidelberg (2001).
- 86.J.M. Kuster, K. Ryndina, H. Gall. Generation of Business Process Models for Object Life Cycle Compliance. In *Proceedings of the 5th International Conference on Business Process Management*, 2007, pp. 165-181.
- 87.Blonk, A. M., Bercken, J. H. L. van den, & De Bruyn, E. E. J. (1996). Evaluation of DYSLEXPERT: A comparison of a knowledge-based system with experienced clinicians in the diagnosis of dyslexia // *Computers in Human Behavior*, 12 (4), 567-586.
- 88.Aarle, E. J. M. van, & Bercken, J. H. L. van den (1992). Development of a knowledge-based system for supporting the diagnosis of reading and spelling problems // *Computers in Human Behavior*, 8 (2/3), 183-201.
- 89.Aarle, E. J. M. van, & Bercken, J. H. L. van den (1999). The development of a knowledge-based system supporting the diagnosis of reading and spelling problems (II) // *Computers in Human Behavior*, 15, 693-712.
- 90.Amosig, J.M., Escara, E.J., Martinez, R. & Paculanang, E. (2008). Feardex: Fear Diagnostic Expert System // <http://www.shvoong.com>
- 91.Гаврилова Т.А. Исследования по искусственному интеллекту в Ленинграде. // *Новости искусственного интеллекта*. М.: АИИ, 1991. N2.
- 92.Интерактивная профориентационная экспертная система ПРОФЭКСПЕРТ [Электронный ресурс]<http://www.teletesting.ru/modules/tests/?pg=5>.

93. Программный комплекс Лонгитюд-ЭДК [Электронный ресурс] <http://lnd.pu.ru/>
94. First, M.B., Spitzer, R.L., Gibbon, M., & Williams, J.B.W. (2000). Computer Assisted SCID II Expert System (CAS-II ES) // <http://www.psychpress.com.au>
95. Васищев А.А. PsychometricExpert. Многофункциональная среда разработки психодиагностических баз данных и экспертных систем. Руководство пользователя. – Ярославль, 2004.
96. Назаренко Г.И., Полубенцева Е.И. – Медицинский технологический процесс. – М.: Медицина XXI, 2003.
97. Munson, J.: Software Engineering Measurement. Auerbach Publications (2003)
98. McGarry, J., Card, D., Jones, C., et al.: Practical Software Measurement: Objective Information for Decision Makers. Addison-Wesley, Reading (2001)
99. Garcia, F., Bertoa, M., Calero, C., et al.: Towards a Consistent Terminology for Software Measurement. Information and Software Technology 48(8), 631–644 (2006).
100. Mayer – Oakes S.A., Barnes C. Developing indicators for the Medical Quality Indicator System (MQIS): challenges and lesson learning//Jt. Comm.J. Qual. Improv. – 1997. –Vol. 23. – P. 381-390.
101. Frank, U., Heise, D., Kattenstroth, H., Schauer, H.: Designing and Utilizing Business Indicator Systems within Enterprise Models - Outline of a Method. In: Loos, P., Nüttgens, M., Turowski, K., Werth, D. (eds.) Modellierung Betrieblicher Informationssysteme (MobIS 2008) - Modellierung zwischen SOA und Compliance Management. LNI, vol. 141, pp. 89–105. Saarbrücken, Germany (2008)
102. Popova, V., Treur, J.: A Specification Language for Organizational Performance Indicators. Applied Intelligence Journal 27(3), 291–301 (2007).

103. Г.И.Назаренко, Е.И.Полубенцева. Проектирование медицинских технологических процессов. М.: Медицинский центр Банка России, 2001.
104. Молодченков А.И. Формализация описания лечебно-диагностических процессов. Нечеткие системы и мягкие вычисления: сб. ст. Третьей Всероссийской научной конференции: В 2 т. Т I , Волгоград, 2009, с. 103-109.
105. Calligaro K.D., Dougherty M.J. Raviola C.A et al. Impact of clinical pathways on hospital costs and early outcome after major vascular surgery//J.Vasc.Surg.-1995.-Vol. 22-P.649.- 657.
106. Малышевский А.А., Молодченков А.И., Хачумов В.М. Визуализация медицинских технологических процессов и многомерных данных для поддержки принятия решений // Информационные и телекоммуникационные системы: информационные технологии в научных и образовательных процессах. Материалы республиканской научно-практической конференции. Махачкала: ДНЦ РАН. 2009, С. 164-176.
107. Назаренко Г.И., Осипов Г.С. Основы теории медицинских технологических процессов. Ч. 2. Исследование медицинских технологических процессов на основе интеллектуального анализа данных. – М.: ФИЗМАТЛИТ, 2006. – 144 с.
108. Г.И. Назаренко, Г.С. Осипов, А.Г. Назаренко, А.И. Молодченков. Интеллектуальные системы в клинической медицине. Синтез плана лечения на основе прецедентов // Информационные технологии и вычислительные системы. - 2010, №1, С24-35.
109. Молодченков А.И. Автоматический синтез типовых схем лечения на основе медицинских данных. X Ежегодная специализированная конференция и выставка «Информационные технологии в медицине» / Официальный каталог. Материалы конференции./ - М.: «Консэф», 2009, с. 148-156.

110. T. Weijters, W.M.P. van der Aalst. Process Mining: Discovering Workflow Models from Event-Based Data. In Kröse, B. et. al, (eds.): Proceedings 13th Belgium-Netherlands Conference on Artificial Intelligence (BNAIC'01), 25-26 October 2001, Amsterdam, The Netherlands, pp. 283-290.
111. Mărușter, L., Weijters, A.J.M.M., Aalst, W.M.P., and Bosch, A. 2002. Process mining: Discovering direct successors in process logs. In S. Lange, K. Satoh, and C.H. Smith (Eds.), Proceedings of the 5th International Conference on Discovery Science (Discovery Science 2002), Berlin: Springer-Verlag, vol. 2534: pp. 364–373.
112. Mărușter, L., Weijters, A., Aalst, W., and Bosch, A. 2002. Process mining: Discovering direct successors in process logs. In S. Lange, K. Satoh, and C.H. Smith (Eds.), Proceedings of the 5th International Conference on Discovery Science (Discovery Science 2002), Berlin: Springer-Verlag, vol. 2534: pp. 364–373.
113. Mărușter L. et al. A rule-based approach for process discovery: Dealing with noise and imbalance in process logs //Data mining and knowledge discovery. – 2006. – T. 13. – №. 1. – C. 67-87.
114. Bo Pang, Lillian Lee, ShivakumarVaithyanathan. Thumbs up? Sentiment Classification using Machine Learning Techniques // Proceeding EMNLP '02 Proceedings of the ACL-02 conference on Empirical methods in natural language processing, 2002 - Volume 10 Pages 79-86.
115. Michalski, R. S., Chilausky, R. L. (1980), Learning by being told and learning from examples: an experimental comparison of the two methods of knowledge acquisition in the context of developing and expert system for soybean disease diagnosis. Policy Analysis and Information Systems, 4(2).
116. Mitchell, T. (1997). Machine Learning. McGraw Hill.
117. Nilsson, N.J. (1965). Learning machines. New York: McGraw-Hill.
118. Adriaans, P., and D. Zantige. 1996. *Data mining*. Harlow, England: Addison-Wesley.

119. Holmes, G., B. Pfahringer, R. Kirkby, E. Frank, and M. Hall. 2002. Multiclass alternating decision trees. In T. Elomaa, H. Mannila, and H. Toivonen, editors, *Proceedings of the Thirteenth European Conference on Machine Learning*, Helsinki, Finland. Berlin: Springer-Verlag, pp. 161–172.
120. Abe, N., Zadrozny, B., & Langford, J. (2006). Outlier detection by active learning. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 767–772). New York: ACM Press.
121. Foulds, J., & Frank, E. (2008). Revisiting multiple-instance learning via embedded instance selection. In *Proceedings of the Australasian Joint Conference on Artificial Intelligence* (pp. 300–310). Auckland. Berlin: Springer-Verlag.
122. Молодченков А.И. Применение AQ-алгоритма для персонификации лечебно-диагностических процессов //Теория и практика системного анализа: Труды I Всероссийской научной конференции молодых ученых. – Т. I. – Рыбинск: РГАТА им.П.А. Соловьева, 2010, с.79-84.
123. Michalski, R. S., AQVAL/1--Computer Implementation of a Variable-Valued Logic System VL1 and Examples of its Application to Pattern Recognition // *Proceedings of the First International Joint Conference on Pattern Recognition*, Washington, DC, pp. 3-17, October 30 - November 1, 1973.
124. А.И. Молодченков Анализ результатов работы алгоритма автоматического построения условий, влияющих на выбор метода лечения // Искусственный интеллект и принятие решений. – 2014, № 4, - С. 52-58
125. G. H. John and P. Langley, “Estimating Continuous Dis-tributions in Bayesian Classifiers,” *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, San Francisco, 1995, pp. 338-345.

126. J. Quinlan, "C4.5: Programs for Machine Learning", Morgan Kaufmann, San Mateo, 1993.
127. I. H. Witten and E. Frank, "Data Mining: Practical Machine Learning Tools and Techniques," 2nd Edition, Morgan Kaufmann, San Francisco, 2005.
128. C.J. van Rijsbergen, Information Retrieval, Butterworth, 1979.
129. David M W Powers "Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness& Correlation". Journal of Machine Learning Technologies2(1): 37–63.
130. Krzysztof Dembczynski, ArkadiuszJachnik, WojciechKotlowski, Willem Waegeman, EykeHuellermeier. Optimizing the F-Measure in Multi-Label Classification: Plug-in Rule Approach versus Structured Loss Minimization. Proceedings of the 30th International Conference on Machine Learning (ICML-13), 2013, pp. 1130-1138.
131. Olson, David L., Delen, Dursun (2008);*Advanced Data Mining Techniques*, Springer, 1st edition (February 1, 2008), page 138.
132. K.H. Brodersen, C.S. Ong, K.E. Stephan, J.M. Buhmann. The balanced accuracy and its posterior distribution. Proceedings of the 20th International Conference on Pattern Recognition, 2010, pp. 3121-3124.
133. UCI Machine Learning Repository. <http://archive.ics.uci.edu/ml/>
134. JanuszWojtusiak, Ryszard S. Michalski, Kenneth A. Kaufman, JaroslawPietrzykowski The AQ21 Natural Induction Program for Pattern Discovery: Initial Version and its Novel Features / International Conference on Tools with Artificial Intelligence - ICTAI, pp. 523-526, 2006.
135. Ian H. Witten, Eibe Frank, Len Trigg, Mark Hall, Geoffrey Holmes, and Sally Jo Cunningham (1999). Weka: Practical machine learning tools and techniques with Java implementations.
136. Ian H. Witten, Eibe Frank (2005). Data Mining: Practical Machine Learning Tools and Techniques (Second Edition, 2005). San Francisco: Morgan Kaufmann

137. Ian H. Witten, Eibe Frank, Mark A. Hall Data Mining Practical Machine Learning Tools and Techniques (Third Edition, 2011). Elsevier, USA, 630 p.
138. G.F. Cooper, E. Herskovits (1990). A Bayesian method for constructing Bayesian belief networks from databases.
139. G. Cooper, E. Herskovits (1992). A Bayesian method for the induction of probabilistic networks from data. Machine Learning. 9(4):309-347.
140. M. Stone. Cross-validatory choice and assessment of statistical predictions. Journal of the Royal Statistical Society, B, 36(1):111–147, 1974.
141. Buijs, J.C.A.M. (2014) Receipt phase of an environmental permit application process ('WABO'), CoSeLoG project. Eindhoven University of Technology. Dataset. <http://dx.doi.org/10.4121/uuid:a07386a5-7be3-4367-9535-70bc9e77dbe6>
142. vanDongen, B.F. (2011) Real-life event logs - Hospital log. Eindhoven University of Technology. Dataset. <http://dx.doi.org/10.4121/uuid:d9769f3d-0ab0-4fb8-803b-0d1120ffcf54>
143. Макаров Д.А., Молодченков А.И. Создание систем приобретения знаний для построения медицинских экспертных систем на основе ядра программных инструментальных средств «MedExp». Информационные технологии в медицине. 2009-2010. Тематический научный сборник под ред. Г.С. Лебедева, О.В. Симакова и Ю.Ю. Мухина. - М.: Радиотехника, 2010. с. 130-136.
144. Молодченков А.И. Применение программных средств PsyExp для создания экспертных систем в области практической психологии. //Труды VIII международной конференции Интеллектуальный анализ информации ИАИ-2008, Киев: Просвита, 2008. - С. 298-307.
145. И.А Тихомиров, А.И. Молодченков Неоднородная семантическая сеть как средство представления знаний в предметной области «практическая психология» / Теория и практика системного анализа:

- Труды I Всероссийской научной конференции молодых ученых. – Т. I.
– Рыбинск: РГАТА имени П.А. Соловьева, 2010, стр. 45-49
146. Молодченков А.И. Экспертная система «Психология»./Труды
одиннадцатой национальной конференции с международным участием
КИИ-2008, Том 3, Дубна, ЛЕНАНД, 2008, С. 353-359
147. Молодченков А.И. Экспертные системы в области
психодиагностики./Труды XLIV Всероссийской конференции по
проблемам математики, информатики, физики и химии, Москва, РУДН,
2008, С. 30-39.

Приложение А

Общее описание архитектуры комплекса экспертных систем (ЭС) «Психология», внедренного в Московском городском психолого-педагогическом университете и включающего экспертные системы «Психодиагностика» и «Психологическое консультирование».

ЭС «Психодиагностика» состоит из вводного раздела «Запрос» и двух подсистем, работающих на одной БЗ независимо друг от друга.

В первой из них – «Планирование обследования» - пользователь получает рекомендации по выбору методик в соответствии с учётом возраста клиента, выбранными объектами диагностики и ограничениями, накладываемыми требованиями к батарее методик (принципы экономичности, перепроверки, дополнительности проективных и опросниковых методов) и организационными условиями (время, отводимое на обследование, наличие необходимого стимульного и расходного материалов). В разделе «Итоги планирования» пользователю предлагаются сформированные системой варианты батареи методик и предоставляются (по запросу) объяснения психодиагностической гипотезы, обусловивших выбор той или иной методики.

Вторая подсистема - «Интерпретация результатов обследования» - даёт возможность пользователю после проведения обследования определить значение полученных данных по использованным методикам в терминах объектов диагностики, т.е. получить интерпретацию результатов по отдельным показателям методик (используются интерпретации авторов методик, а также наиболее авторитетных в отечественной психологии интерпретаторов ряда известных методик). В разделе «Итоги обследования» пользователю предоставляется в компактной форме собранная во вводном разделе «Запрос» информация о заказчике, клиенте и запросе, информация об использованных в проведенном обследовании методиках и полученных результатах, рекомендации по формированию психологического диагноза (в

форме указаний на последовательности интеллектуальных операций в структуре психодиагностической деятельности, необходимых для продуктивного выдвижения гипотез и корректного составления диагноза), рекомендации по составлению заключения и организации обратной связи с заказчиком и клиентом (в соответствии с информацией о типе заказчика и возраста клиента).

Блок-схема ЭС «Психодиагностика» приводится на Рисунке 1:

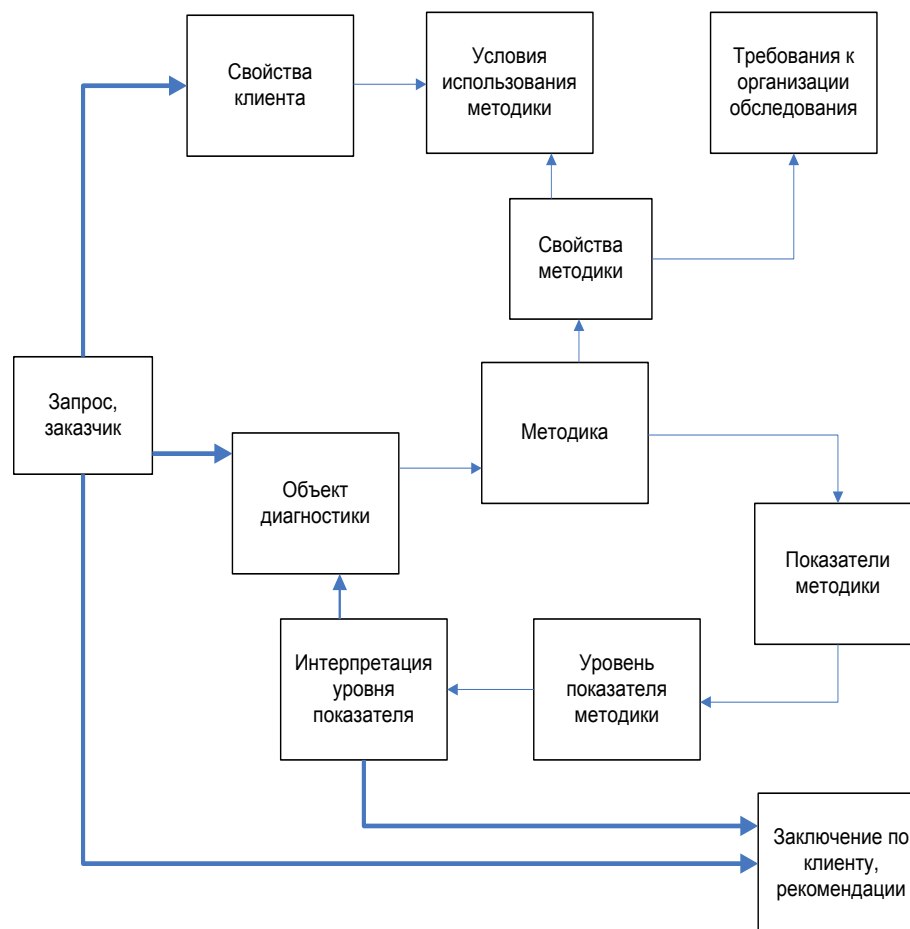


Рисунок 1. Схема работы блока выбора методики диагностики ЭС

Блок-схема ЭС «Психологическое консультирование» приведена на Рисунке2.

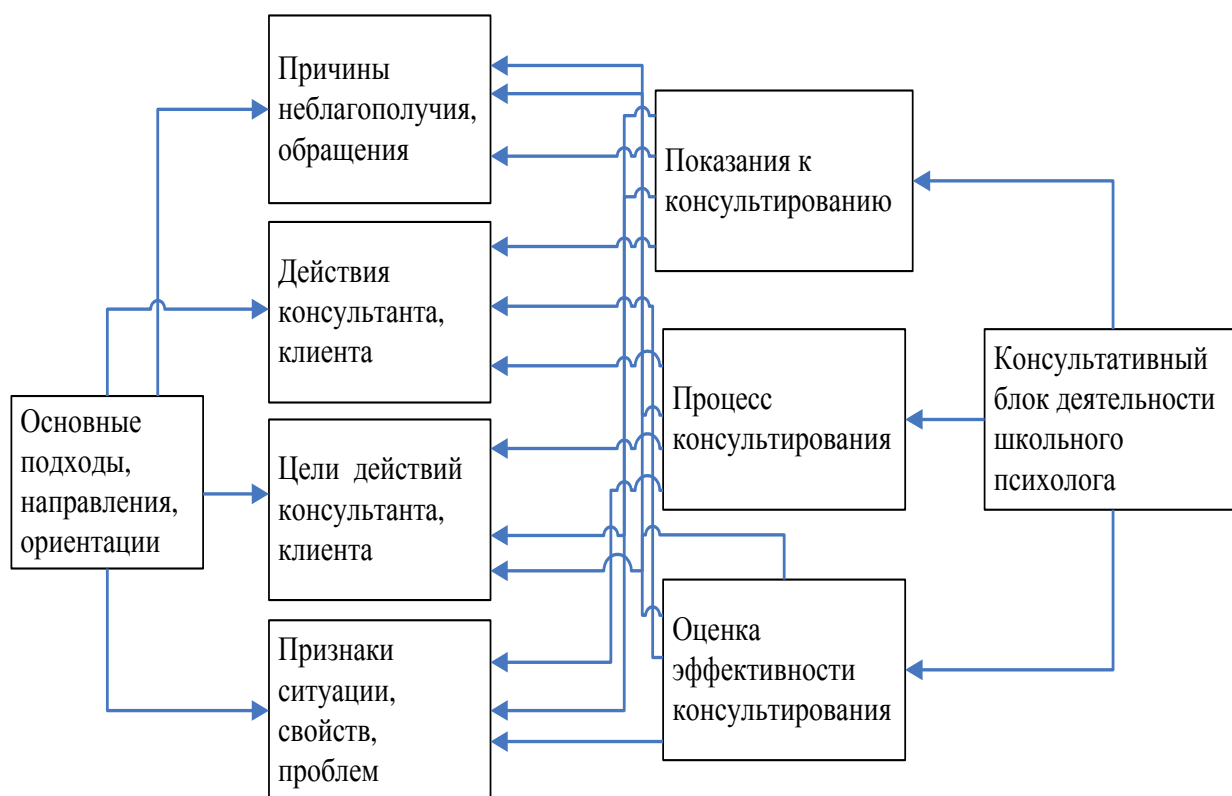


Рисунок 2. Блок-схема ЭС «Психологическое консультирование»

Функциональные блоки ЭС «Психологическое консультирование»

Блок показаний к консультированию

База знаний ЭС содержит описание проблем, с которыми к психологу-консультанту могут обращаться клиенты:

- проблемы отношений: партнерских, супружеских, детско-родительских;
- трудности во взаимодействии с миром, людьми;
- трудности с самим собой;
- проблемы учебы, работы;
- проблемы болезней, зависимостей;
- этнические, мультикультуральные проблемы;
- проблемы самореализации и самоактуализации.

Проблемы представлены типичными жалобами, с которыми в БЗ установлены связи с разделом Причины и разделом Тактика работы.

Блок выбора направления и формы консультативной работы

База знаний ЭС содержит описание существующих в современной психологии направлений или подходов. Раздел Подходы связан в БЗ с разделами Тактика работы и Причины. Выбрав определённый Подход пользователь получает рекомендации по интерпретации Причин и по предпочитаемым в данном подходе Тактикам работы, что создаёт для пользователя операциональный контекст работы в рамках выбранного подхода. Выбрав определённую Причину пользователь получает информацию о Подходе, которому адекватна такая интерпретация Причин, что создаёт для пользователя концептуальный контекст работы с выбранной интерпретацией.

Блок схемы работы в рамках выбранного направления: оценка необходимых навыков, этапов работы, критериев эффективности, источников трудностей

Пользователь имеет возможность в рамках данной ЭС оценить наличие у себя необходимых коммуникативных и личностных свойств, а также сформированных навыков для работы в выбранном направлении по имеющейся проблеме, определить для себя критерии эффективности консультативного взаимодействия, и, наконец, подготовиться к работе на отдельных этапах консультативного процесса:

- 1) контакт, включая способ установления контакта и нарушения контакта;
- 2) сбор информации, включая анализ речи, анализ невербальных компонентов, анализ жалобы, психодиагностические пробы;
- 3) формулировка проблемы;
- 4) влияние, включая приёмы, условия и цели;
- 5) поддержка включая приёмы и условия.

Блок консультационной деятельности школьного психолога

Пользователь – школьный психолог имеет возможность работать только с теми видами Проблем, которые актуальны для его практики, например, запросами родителей школьников по поводу плохой успеваемости,

индивидуально-личностных качеств, особенностей межличностных отношений ребенка со сверстниками и взрослыми и трудностями личных взаимоотношений родителя с ребенком. В базе знаний ЭС установлены связи с соответствующим подразделом Тактик работы, содержащим развёрнутые программы действий с возможностью перехода к работе с ЭС «Психодиагностика».

Подсистема работы с навыками консультанта

Пользователь получает информацию о профессиональных навыках ведения консультации: установление контакта, сбор и анализ анамнеза, анализ жалобы, формулировка проблемы, влияние, поддержка клиента в принятии решения, поддержка клиента в реализации решения, оценка эффективности консультативной деятельности. По каждому из перечисленных процессов разворачивается набор признаков, в зависимости от которых находятся возможные интерпретации либо способы деятельности. Система в вариативном режиме дает решения, опирающиеся на данные связи (признак ситуации - возможная интерпретация или способ деятельности).

Подсистема работы с критерием эффективности консультанта

Пользователь получает информацию о содержательном и временном критериях оценки эффективности консультативного взаимодействия. Ему предлагается определить полноту достижения поставленных целей консультирования, а также учесть при необходимости пост консультационного контакта с клиентом. Оценке подлежат: непосредственный результат с точки зрения клиента, непосредственный результат с точки зрения консультанта; отсроченный результат с точки зрения клиента; отсроченный результат с точки зрения консультанта; при необходимости непосредственный и отсроченный результат третьего лица (третьих лиц).

Подсистема работы с личностными основаниями деятельности консультанта

Пользователь получает информацию об объективных требованиях, предъявляемых к коммуникативным и личностным свойствам консультанта различными направлениями консультативной практики, а также возможность соотнести самооценочные характеристики с системами этих требований. На основе проводимого сопоставления пользователь имеет возможность принятия решения о степени готовности к реализации того или иного подхода в собственной практике.

Далее описаны основные блоки ЭС «Психодиагностика»

Функциональные блоки ЭС «Психодиагностика»

Блок формулировки проблемы ЭС «Психодиагностика».

Пользователю предлагается путём выбора из списка определить заказчика предстоящей диагностической работы (например, для школьной психодиагностики это может быть учитель, администрация школы, родитель, сам подросток и т.д.). Информация о заказчике необходима как для уточнения запроса (определения объекта диагностики), так и для формирования рекомендаций по представлению результатов обследования заинтересованным лицам (организация обратной связи с заказчиком и клиентом). В связи с этим данная информация сохраняется до последнего этапа работы пользователя с ЭС.

Далее пользователю предлагается ответить на вопросы, связанные с фиксацией терминологии заказчика и уточнением его планов использования результатов диагностики. Эта информация также сохраняется до последнего этапа работы пользователя с ЭС, что позволяет ему опереться при составлении заключения и рекомендаций на аутентичные представления заказчика о содержании запроса и перейти – в соответствии с правилами, представленными в рекомендациях ЭС – от психологического диагноза к представлению результатов обследования в доступной и желательной для заказчика форме.

Из всего многообразия характеристик клиента только две позволяют ввести алгоритмы в деятельность диагноста на данном этапе – принадлежность к определённой возрастной группе и статус индивидуального или группового субъекта. Эти характеристики указываются пользователем в соответствующем окне интерфейса и используются системой для первичного отбора диагностических методик.

Перевод запроса на язык психологических терминов осуществляется пользователем при обращении к разделу «Объект диагностики». Процедура выбора объекта организована таким образом, что позволяет пользователю видеть всё поле объектов, связанных с выбранным родовидовыми отношениями и отношениями часть-целое. Такое представление объекта в ЭС поддерживает работу пользователя по формированию диагностической гипотезы, позволяя сформулировать её с необходимой широтой и точностью за счёт учёта понятийного контекста, в котором существует выбираемый объект диагностики.

Блок диагностического инструментария: теоретические модели отдельных методов и методик, психометрические показатели методик, библиотека методик.

Блок методического инструментария содержит в общей сложности около 150 методик. Каждая методика представлена в базе знаний системы как объект раздела «Объект диагностики» и характеризуется рядом формальных показателей, а также содержательными показателями, в которых проявляются диагностируемые данной методикой психологические особенности (объекты диагностики). При формировании списка содержательных показателей разработчики опирались на показатели, введённые авторами методики, авторами её адаптации или уважаемыми в психологическом сообществе интерпретаторами методики. В качестве объектов диагностики, исследуемых с помощью показателя данной методики в системе, используются и те, которые выделены в методике в качестве её направленности, и те, которые соответствуют названиям показателей методики (например, шкалам опросника), и те, которые можно выделить в

описании типа, принадлежность к которому диагностируется по данному показателю, и те, которые связаны с особенностями профиля шкал или иных мета показателей методики. Такая схема связей между объектами диагностики и показателями методик, по которым можно судить об этих объектах, позволяет ЭС предлагать решения задачи выбора методик (вторичный отбор методик по содержательным характеристикам) и поддерживать процесс формирования батареи измерительных средств, адекватных диагностическим гипотезам. Отметим, что установленные в ЭС связи между показателями методик и объектом диагностики, с одной стороны, и объектом диагностики и другими психологическими характеристиками, с другой, создают для пользователя возможность отнести к выбранному объекту как к контекстно связанному. Фактически, содержание соответствующего понятия раскрывается в двух планах – в плане интерпретации этого термина автором методики, имплицитно содержащейся в выделяемых им показателях, и в плане общепсихологических теоретических представлений о психических свойствах, процессах и состояниях.

В целом, ЭС своей работой ориентирует пользователя-диагноста как на учет принципов психологии - принципа системности психических качеств, принципа развития психических качеств, принципа активности субъекта, так и на учёт требований к батарее методик - соответствие гипотезе, соответствие возможностям клиента, соответствие заказу, соответствие принципам экономности, мотивирования клиента, перепроверки.

Блок требований к диагносту.

В ЭС содержится набор требований к диагносту, традиционно предъявляемых к нему как к субъекту формирования психодиагностических гипотез и психологического диагноза. Для пользователя эти требования представлены как в виде прямых указаний на последовательности интеллектуальных операций в структуре психодиагностической деятельности, необходимых для продуктивного выдвижения гипотез и

корректного составления диагноза, так и в форме имплицитного знания об «узловых точках» в работе диагноста.

Структура системы выделяет для пользователя в качестве объектов внимания этапы психодиагностического исследования и их операциональный состав: формулировка проблемы, планирование обследования, сбор и обработка данных, интерпретация полученных результатов, формирование психологического диагноза и формулирование заключения/выводов.

Следующий аспект работы с требованиями к диагносту связан в ЭС «Психодиагностика» с организацией базы знаний. В качестве центрального элемента, с которым и через который связаны все другие элементы сети, выбран «показатель методики». При планировании обследования именно краткое описание показателей методики (в случае необходимости даётся и расширенное описание, вынесенное в Комментарий) служит для пользователя основанием для выбора данной методики как отвечающей задаче исследования данного объекта диагностики. При интерпретации результатов обследования единицей анализа данных для пользователя становится интерпретация уровня показателя методики. Таким образом, пользователь получает возможность рассмотреть уже совокупность характеристик деятельности клиента/испытуемого в ходе выполнения диагностических заданий, что позволяет ставить задачу построения непротиворечивого и целостного описания.

Блок диагностической деятельности школьного психолога.

В данный блок входят методики, используемые в школьной психодиагностике (более 90 методик и методических приёмов), объекты диагностики, характерные только для практики школьного психолога, а также комментарии к обратной связи заказчику и клиенту и к возрастным группам, с которыми работает школьный психолог.

Для младших школьников диапазон методик и процедур тестирования значительно расширяется. Помимо объективных тестов способностей, проективных личностных методик применяются **небольшие**

опросники (7-10 пунктов) с выбором вариантов ответов или короткими односложными альтернативами (например, опросники учебной мотивации). В ряде случаев возможно **групповое тестирование**, использование бланковых методик, когда дети самостоятельно выполняют задание в индивидуальных брошюрах, на листочках, выслушав общую инструкцию, а также компьютерное обследование. В условиях группового тестирования необходимо следить, чтобы учащиеся не общались между собой и не оказывали влияния друг на друга. Предпочтительными остаются **непродолжительные по времени (10-20 мин), увлекательные по форме практические задания**.

В **среднем школьном возрасте** возможны все виды тестирования: индивидуальное, групповое, бланковое, компьютерное. **Продолжительность тестирования**, в том числе сложных интеллектуальных заданий может быть увеличена до 40 мин – 1 часа. Снижается эффективность проективных рисуночных методов, воспринимаемых в этом возрасте «как слишком детские» или несерьезные, вследствие чего дети начинают выполнять их небрежно. Информация, получаемая посредством опросников в весьма сильной степени подвержена влиянию **социальной желательности, идеала-Я**.

В **старшем школьном возрасте** значительно повышается интерес к **личностным опросникам** и проективным личностным методикам, эффективность и достоверность которых значительно возрастает вследствие стремления подростков к самоисследованию и самораскрытию.

Достоверность и эффективность диагностики **взрослых людей** (от 18-20 лет) в значительной степени зависит от уровня принятия цели исследования и практической заинтересованности испытуемых в решении конкретных жизненных проблем: устройство на работу, нормализация семейных, межличностных отношений и т.д. Выбор методики и формы тестирования и ничем не ограничен кроме соображений практической целесообразности».

Подсистема работы с методиками

Подсистема работы с методиками организована как автономная справочная система по имеющимся в базе знаний методикам.

Каждая методика представлена списком формальных и содержательных свойств. К числу формальных свойств отнесена такая информация о методике, как полное её название, литературный источник информации о ней, тип метода, к которому принадлежит методика, количество вопросов или заданий, содержащихся в методике, необходимость стимульного материала, который должен быть приобретён или изготовлен диагностом, необходимость бланков или иных расходных материалов, которые должны быть подготовлены диагностом заранее, время, необходимое для проведения методики, наличие нормативов (возрастных или половозрастных, если таковые имеются в авторском описании).

Содержательные свойства – это показатели методики, фиксирующие наличие/отсутствие или уровень выраженности определённой психологической характеристики. Благодаря установленным в базе связям между показателями каждой методики и психологическими характеристиками, организованными в дерево объектов диагностики, информация о любой методике оказывается доступна пользователю при произвольной степени обобщенности вопроса.

Некоторые методики снабжены комментариями экспертов. Эти комментарии содержат указания на трудности, с которыми может столкнуться неопытный пользователь при работе с данной методикой, а также описания тех возможностей методики, которые не были отмечены автором, но были выявлены в ходе её практического использования. Отдельные показатели некоторых методик также снабжены комментариями, раскрывающими их содержание (в первую очередь, это касается новых или редко используемых методик).

К методам диагностики – в системе они выступают одним из формальных свойств методик – даны комментарии, основная задача которых – указать молодому специалисту, на что необходимо обратить внимание при принятии решения о выборе методики, относящейся к данному типу методов.

Подсистема статистической обработки

Подсистема работы статистической обработки организована как автономная справочная система по имеющимся статистическим методам. Основное назначение подсистемы статистической обработки – помощь психодиагносту в подборе адекватных его задаче статистических методов. Подсистема реализована в виде отдельного приложения, которое запускается путем нажатия кнопки на панели диагностической системы.

База знаний ЭС «Психодиагностика»

База знаний ЭС «Психодиагностика» включает в себя следующие разделы: «Сферы диагностики», «Клиент», «Объект диагностики», «Свойства методик», «показатели методики», «уровни показателя методики», «интерпретация уровня показателя методики», «Итоги планирования». Разделы БЗ содержат набор объектов. Объекты связаны друг с другом связями различных типов.

В БЗ «Психодиагностика» применяются следующие типы связей:

- ARootOfB – объект A является корнем объекта B, объект B является листом (принадлежит, находится в подчинении) объекта A.
- A LeafOf B – объект A является листом (принадлежит, находится в подчинении) объекта B, объект B является корнем объекта A.
- APokazatelOfB – объект A является показателем объекта B, объект B является методикой объекта A.
- AMethodicsOfPokazatelB – объект A является методикой объекта B, объект B является показателем объекта A.
- AFeatureOfB – A является особенностью B, B является методикой A.

- AMethodicsOfFeatureB – A является методикой B, B является особенностью A.
- AClientOfB – A является описанием клиента для B, B является методикой A.
- AMethodicsOfClientB – A является методикой B, B является описанием клиента для A.
- ADiagObjectOfB – A является объектом диагностики для B, B является методикой для A.
- AMethodicsOfDiagObjectB – A является методикой B, B является объектом диагностики для A.
- решатель для системы «Психологическая диагностика» интерпретирует эти связи следующим образом:
- связь RootOf и LeafOf интерпретируются решателем как иерархические связи, на их основании строится дерево объектов.
- связи ClientOf и MethodicsOfClient необходимы для построения батареи методик по клиентским данным, а также для ограничения числа методик, исходя из возраста клиента или формы его обследования.
- связи DiagObjectOf и MethodicsOfDiagObject необходимы для построения батареи методик по набору объектов диагностики.
- FeatureOf и MethodicsOfFeature – данная связь соединяет два объекта, один из которых является методикой, а второй – свойством этой методики, если методика попала в батарею методик, то свойства должны быть видны пользователю.
- PokazatelOf и MethodicsOfPokazatel – связь соединяет два объекта, один из которых является методикой, а второй – показателями, по которым оцениваются результаты методики, если пользователем была выбрана и проведена некоторая методика, то он должен иметь доступ к показателям этой методики.

База знаний ЭС «Психологическое консультирование».

База знаний «Психологическое консультирование» включает в себя достаточно много разделов («Принятие решения о консультировании», «Показания к выбору вида консультирования», «Типичные проблемы», «Этапы психологического консультирования», «Контакт», «Сбор информации», «Формулировка проблемы», «Влияние», «Эффективность процесса консультирования», «Параметры профессиональной пригодности консультанта», «Типология клиентов» и т.д.). Психолог-консультант сам решает, с какими разделами данной системы ему работать. Разделы приписаны к некоторым большим группам разделов, каждая группа выводится на отдельной закладке (например, группа «Консультативный блок деятельности школьного психолога», куда входят разделы, связанные с деятельностью школьного психолога). При работе с каждым из разделов система находит отдельное решение. Например, для раздела «Контакт» результатом является возможная причина нарушения контакта и рекомендуемые пользователю действия.

- ARootOfB – объект A является корнем объекта B, объект B является листом (принадлежит, находится в подчинении) объекта A.
- A LeafOf B – объект A является листом (принадлежит, находится в подчинении) объекта B, объект B является корнем объекта A.
- ARS-RSB – при наблюдении объекта A может наблюдаться объект B, при наблюдении объекта B может наблюдаться объект A.

Приложение Б

Разработка системы поддержки процесса принятия решений (СППР) о кардиологическом вмешательстве, которая была разработана на базе Медицинского центра Банка России. СППР представляет собой web-приложение, которое обеспечивает эффективную навигацию между всеми блоками, реализующими процесс формализации клинических руководств и поддержки принятия решения о проведении коронарного вмешательства.

Архитектура системы изображена на рисунке 3.

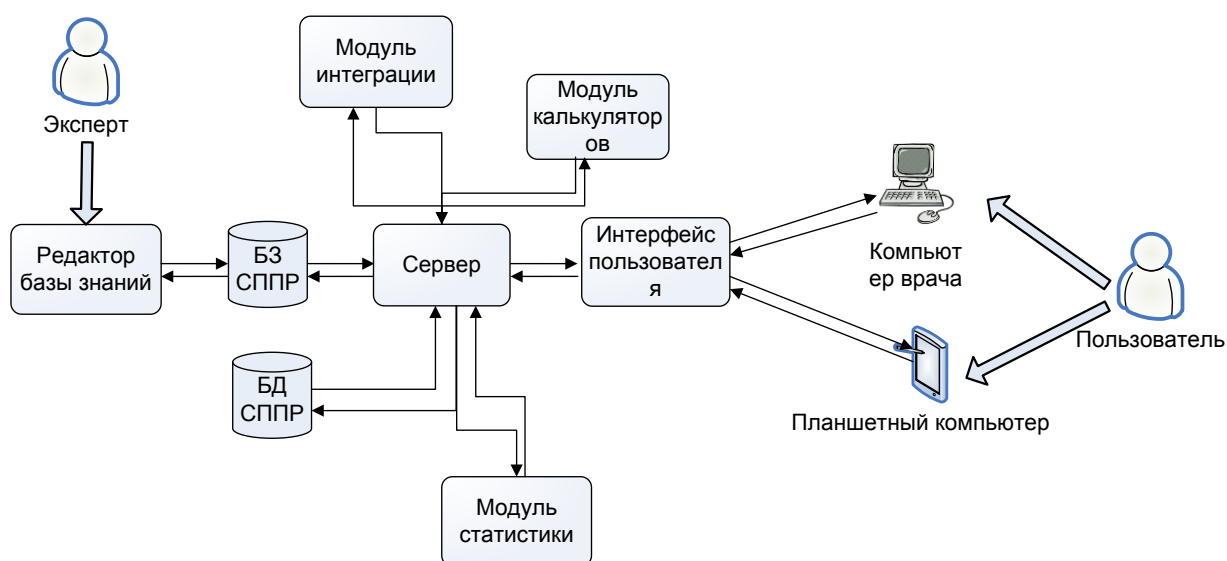


Рисунок 3. Архитектура системы поддержки принятия решений

Весь основной функционал СППР расположен на сервере. Для работы с системой пользователю предоставляется программный интерфейс. Этот может быть web-интерфейс. Web-интерфейс позволяет работать с системой с помощью различных браузеров, установленных на компьютере, планшете или мобильном устройстве пользователя. Этот способ реализации не требует установки дополнительно программного обеспечения на компьютере пользователя. Для обновления системы достаточно обновить модули на сервере. Архитектура системы позволяет разрабатывать и win32-интерфейс пользователя. Однако в этом случае для различных платформ придется разрабатывать отдельное приложение. При обновлении необходимо обновить

не только серверную часть, клиентское программное обеспечение на всех компьютерах пользователя. В этом случае удобными являются облачные технологии.

Модуль интеграции содержит функционал для взаимодействия с установленными в лечебном учреждении медицинскими информационными системами. Как уже было сказано выше, способ интеграции зависит от того, какие системы установлены в медицинском учреждении.

Модуль калькуляторов и шкал, содержит в себе набор средств, позволяющих проводить все необходимые расчеты в процессе поддержки различных этапов лечебно-диагностического процесса.

Модуль статистики позволяет собирать всю информацию, хранящуюся в лечебном учреждении, и содержит функционал для ее статистической обработки, оценки качества оказываемой медицинской помощи, отслеживания отклонений от процесса лечения, оценки этих отклонений и принятия мер по их дальнейшему устранению.

Редактор базы знаний предназначен для внесения медицинских знаний в базу знаний системы. Внесением информации в базу знаний занимается эксперт.

Процесс поддержки принятия решения был разбит на пять сценариев принятия решения о коронарном вмешательстве:

- Сценарий 1: «Принятие решения о проведении коронарной ангиографии»;
- Сценарий 2: «Принятие решения о проведении коронарной реваскуляризации при стабильной стенокардии без АКШ в анамнезе»;
- Сценарий 3: «Принятие решения о проведении коронарной реваскуляризации при стабильной стенокардии с АКШ в анамнезе»;
- Сценарий 4: «Принятие решения о проведении коронарной реваскуляризации при остром коронарном синдроме с подъемом сегмента ST»;

- Сценарий 5: «Принятие решения о проведении коронарной реваскуляризации при остром коронарном синдроме без подъема сегмента ST».

Сценарии содержат все необходимые для принятия решения вопросы, а также калькуляторы и шкалы для расчетов риска, определения необходимости и объема перипроцедурной подготовки пациентов. Таким образом, уже с первых моментов принятия решения о необходимости проведения ЧКВ, а также определения сроков операции, разработанные экспериментальные средства обеспечивают:

1. Поддержку решения о необходимости проведения вмешательства на основании рекомендаций, изложенных в клинических руководствах, для пациентов со стабильной стенокардией напряжения, острым коронарным синдромом, а также определение показаний для диагностической коронарной ангиографии у пациентов с подозрением на ИБС.
2. Оценку индивидуального риска КВ на основании шкал риска, включая TIMI, GRACE, Mayo, EuroSCORE, SYNTAX, а также дополнительных индивидуальных факторов.
3. Поддержку решения о выборе типа стента на основании рекомендаций, изложенных в клинических руководствах.
4. Поддержку решения о перипроцедурном ведении пациентов в зависимости от индивидуального профиля риска.
5. Автоматическое формирование протокола заседания кардиологического совета

В СППР о проведении коронарного вмешательства входит автоматизированный протокол операции, включая заполнение SYNTAX score (шкала для оценки тяжести поражения коронарного русла), что обеспечивает полноту сбора информации и упрощает автоматическую обработку данных. В зависимости от объема проведенного вмешательства, типа установленного стента, топографических и анатомических особенностей, имеющих у

данного пациента, автоматически формируются рекомендации по послеоперационному ведению пациента.

Была сформирована база знаний на основе неоднородной семантической сети, в которой содержатся объекты следующих типов: «Сценарий», «Показания», «Значения показаний», «Симптомокомплексы», «Уровни доказательности». Объекты имеют свойства, набор которых является характеристикой этого объекта.

Объекты в базе знаний сгруппированы в разделы. Объекты могут попадать в разные разделы в зависимости от их функциональной роли. Чаще всего они сгруппированы тематически. Количество разделов, как и объектов не ограничено. В СППР имеются следующие разделы

- **Модуль** – это раздел системы для независимой работы специалистов одного направления.
- **Заполняемая форма (Форма)** – представляет собой набор вопросов и ответов.
- **Выходная форма** - автоматически генерируемые Системой наборы рекомендаций для конкретной заполняемой формы. Наполнение расчетной формы определяется содержанием справочников Показания и Наборы показаний.
- **Сценарий** – это набор заполняемых и расчетных форм на один клинический случай (например, на одно коронарное вмешательство, на случай госпитализации пациента с ишемическим инсультом). В каждом модуле свой набор сценариев и соответствующих им клинических случаев. Одна из заполняемых форм сценария является основной, по ней идентифицируют весь сценарий.
- **Вкладка** – это интерфейсный элемент системы, предназначенный для отображения форм. На каждой вкладке отображается одна форма. Набор вкладок для одного клинического случая формирует сценарий в интерфейсе системы. Вкладки бывают следующих типов:

- **Основная вкладка** предназначена для отображения основной заполняемой формы сценария. Принадлежность формы основной вкладке как раз делает эту форму основной.
- **Дополнительная вкладка** предназначена для отображения прочих (не являющихся основными) заполняемых форм сценария
- **Показания** – это клинические рекомендации, соответствующие конкретному набору ответов на определенный набор вопросов в заполняемой форме.

СППР интегрирована с МИС, установленной в Медицинском центре Банка России, и другими системами, используемыми в процессе лечения. Интеграция была проведена на уровне базы данных. Схема взаимодействия СППР с МИС изображена на рисунке 4.

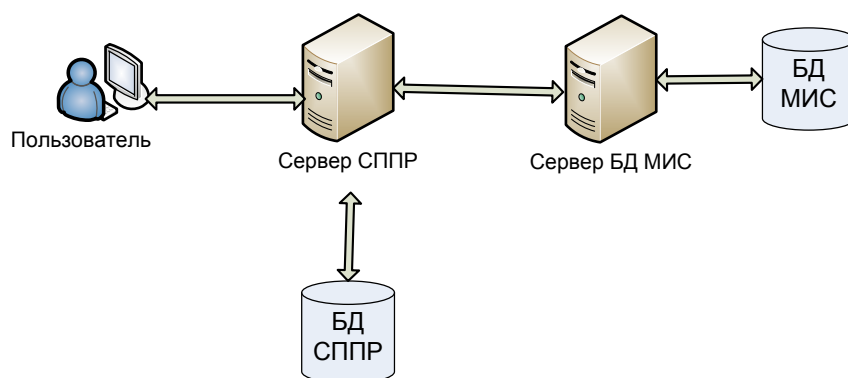


Рисунок 4. Схема взаимодействия СППР с МИС МЦЦБ

Внедрение системы поддержки процесса принятия решения о коронарном вмешательстве позволило существенно повысить качество оказанной медицинской помощи.

Приложение В

Для проведения экспериментальных исследований разработанного алгоритма автоматического синтеза медицинских технологических процессов была сделана их программная реализация с помощью технологии .NET на языке программирования C#. В этом приложении представлены фрагменты программного кода основных функций. Перед работой алгоритма необходимо преобразовать формат описания технологических процессов в лог файле, в формат, интерпретируемый алгоритмом. Фрагмент этого кода представлен ниже.

```
ArrayList ar = new ArrayList();

//Считывание медицинских данных
string filename = Application.StartupPath +
"\medical_data\Hospital_log.xes";
XmlDocument doc = new XmlDocument();
doc.Load(filename);

XmlNodeList list = doc.GetElementsByTagName("trace");

//Обрабатываем Hospital_log.xes
XmlNodeList ProcessInstance_nodes =
list[i].SelectNodes("string[@key='Diagnosis'][@value='maligniteit cervix']");
if (ProcessInstance_nodes.Count > 0)
{
    strsr.Add("<trace>" + list[i].InnerXml + "</trace>");

    XmlNodeList ProcessEvents_nodes = list[i].SelectNodes("event");
    for (int j = 0; j < ProcessEvents_nodes.Count; j++)
    {
        XmlNodeList event_nodes =
ProcessEvents_nodes[j].SelectNodes("string[@key='concept:name']");
        string val = event_nodes[0].Attributes["value"].Value;
        events.Add(val);
    }
}
else
{
    ProcessInstance_nodes =
list[i].SelectNodes("string[@key='Diagnosis:1'][@value='maligniteit cervix']");
    if (ProcessInstance_nodes.Count > 0)
    {
        strsr.Add("<trace>" + list[i].InnerXml + "</trace>");
        XmlNodeList ProcessEvents_nodes = list[i].SelectNodes("event");
        for (int j = 0; j < ProcessEvents_nodes.Count; j++)
        {
            XmlNodeList event_nodes =
ProcessEvents_nodes[j].SelectNodes("string[@key='concept:name']");
            string val = event_nodes[0].Attributes["value"].Value;
            events.Add(val);
        }
    }
}
```

```

    }
}
//----- конец обработки Hospital_log.xes -----

```

Ниже представлен код функции построения экземпляров МТП. Матрицы смежности представлены с помощью класса SparseArray, который позволяет эффективно и быстро работать с матрицами и массивами большой размерности.

```

//Функция извлечения частных МТП из лога событий
private KeyValuePair<Hashtable, SparseArrays.Sparse2DMatrix<int, int, int>>
get_Private_MTP(ArrayList events)
{
    if (events.Count > 0)
    {
        Hashtable event_keys = new Hashtable(); //содержит номера строк и
        столбцов, которые соответствуют событиям
        SparseArrays.Sparse2DMatrix<int, int, int> matr = new
        SparseArrays.Sparse2DMatrix<int, int, int>(0); //матрица смежности событий
        int k = 0;

        string prev_event = Convert.ToString(events[0]);
        event_keys.Add(prev_event, k);
        k++;
        for (int i = 1; i < events.Count; i++)
        {
            int c_k = -1;
            int p_k = Convert.ToInt32(event_keys[prev_event]);
            string curr_event = Convert.ToString(events[i]);
            if (event_keys.ContainsKey(curr_event))
            {
                c_k = Convert.ToInt32(event_keys[curr_event]);
            }
            else
            {
                event_keys.Add(curr_event, k);
                c_k = k;
                k++;
            }
            matr[p_k, c_k] = 1;
            prev_event = curr_event;
        }

        return new KeyValuePair<Hashtable, SparseArrays.Sparse2DMatrix<int, int,
int>>(event_keys, matr);
    }
    else
        return new KeyValuePair<Hashtable, SparseArrays.Sparse2DMatrix<int, int,
int>>(null, null);
}

```

Процедура объединения матриц смежности.

```

private SparseArrays.Sparse2DMatrix<int, int, int>
Summ_matr(SparseArrays.Sparse2DMatrix<int, int, int> matrix1,
SparseArrays.Sparse2DMatrix<int, int, int> matrix2, int count)

```

```

{
    SparseArrays.Sparse2DMatrix<int, int, int> res = new
SparseArrays.Sparse2DMatrix<int,int,int>(0);

    for (int i = 0; i < count; i++)
    {
        for (int j = 0; j < count; j++)
            res[i, j] = Math.Max(matrix1[i, j], matrix2[i, j]);
    }

    return res;
}

```

Ниже представлен код процедуры приведения матриц смежности экземпляров МТП к единому размеру.

```

//Приводим все матрицы с одному виду (они должны имет ону размерность,
//строкам и столбцам соответствуют одни и те же операторы
KeyValuePair<Hashtable, SparseArrays.Sparse2DMatrix<int, int, int>> workpair
= new KeyValuePair<Hashtable, SparseArrays.Sparse2DMatrix<int, int, int>>();

//Количество примеров, в которых есть связь от i-го элемента к j-му

double ddd = Math.Pow(2, 3);
Hashtable ex_count = new Hashtable();

//Количество встречаемости действий в примерах
Hashtable app_count = new Hashtable();

exemplars_count = pairs.Count; //запоминаем количество экземпляров частных
МТП

foreach (KeyValuePair<Hashtable, SparseArrays.Sparse2DMatrix<int, int, int>>
pair in pairs)
{
    if (workpair.Key == null)
    {
        workpair = pair;
        Hashtable w_key = workpair.Key;
        SparseArrays.Sparse2DMatrix<int, int, int> w_val = workpair.Value;

        for (int i = 0; i < w_key.Count; i++)
        {
            for (int j = 0; j < w_key.Count; j++)
            {
                if (w_val[i, j] == 1)
                    ex_count.Add("(" + i + ", " + j + ")", 1);
            }
        }

        foreach (DictionaryEntry de in w_key)
        {
            app_count.Add(Convert.ToString(de.Key), 1);
        }
    }
    else
    {
        Hashtable w_key = workpair.Key;
        SparseArrays.Sparse2DMatrix<int, int, int> w_val = workpair.Value;

        Hashtable c_key = pair.Key;
        SparseArrays.Sparse2DMatrix<int, int, int> c_val = pair.Value;
    }
}

```

```

foreach (DictionaryEntry de1 in c_key)
{
    string s_key = Convert.ToString(de1.Key);
    if (app_count.ContainsKey(s_key))
    {
        app_count[s_key] = Convert.ToInt32(app_count[s_key]) + 1;
    }
    else
    {
        app_count.Add(Convert.ToString(s_key), 1);
    }
}

for (int i = 0; i < w_key.Count; i++)
{
    string w_skey = GetKeyOfValue(w_key, i);

    if (w_skey != "")
    {
        if (c_key.ContainsKey(w_skey))
        {
            int c_ival = Convert.ToInt32(c_key[w_skey]);

            if (c_ival > i)
            {
                string c_skey = GetKeyOfValue(c_key, i);
                int count = c_key.Count;
                for (int j = 0; j < count; j++)
                {
                    int m_val = c_val[i, j];
                    c_val[i, j] = c_val[c_ival, j];
                    c_val[c_ival, j] = m_val;
                }

                for (int j = 0; j < count; j++)
                {
                    int m_val = c_val[j, i];
                    c_val[j, i] = c_val[j, c_ival];
                    c_val[j, c_ival] = m_val;
                }

                c_key[w_skey] = i;
                c_key[c_skey] = c_ival;
            }
        }
        else
        {
            string c_skey = GetKeyOfValue(c_key, i);
            if (c_skey != "")
            {
                int count = c_key.Count;
                for (int j = 0; j < count; j++)
                {
                    int v = c_val[i, j];
                    c_val[count, j] = v;
                    c_val[i, j] = 0;
                }

                for (int j = 0; j <= count; j++)
                {
                    c_val[j, count] = c_val[j, i];
                    c_val[j, i] = 0;
                }
            }
        }
    }
}

```

```

    }
    c_key[c_skey] = count;
    c_key.Add(w_skey, i);
    }
    }
}
if (c_key.Count > w_key.Count)
{
    for (int i = w_key.Count; i < c_key.Count; i++)
    {
        string c_skey = GetKeyOfValue(c_key, i);
        w_key.Add(c_skey, i);
    }
}

for (int i = 0; i < c_key.Count; i++)
{
    for (int j = 0; j < c_key.Count; j++)
    {
        if (c_val[i, j] == 1)
        {
            if (ex_count.ContainsKey("(" + i + ", " + j + ")"))
            {
                ex_count["(" + i + ", " + j + ")"] =
Convert.ToInt32(ex_count["(" + i + ", " + j + ")"]) + 1;
            }
            else
            {
                ex_count.Add("(" + i + ", " + j + ")", 1);
            }
        }
    }
}
}

```

Ниже представлен фрагмент кода удаления шумов.

```

Hashtable w_key1 = workpair.Key;
Hashtable c_wkey = new Hashtable();

SparseArrays.Sparse2DMatrix<int, int, int> w_val1 = workpair.Value;
SparseArrays.Sparse2DMatrix<int, int, int> w_val_without_noise = new
SparseArrays.Sparse2DMatrix<int, int, int>(0); //матрица смежности после удаления шумов
string matr_str2 = "";
string ex_count_str = "";
int ocount = 0;
for (int k = 0; k < w_key1.Count; k++)
{
    ex_count_str = "";
    string mer1 = GetKeyOfValue(w_key1, k); //Получаем название мероприятия
по индексу
    for (int l = 0; l < w_key1.Count; l++)
    {
        if (ex_count.ContainsKey("(" + k + ", " + l + ")"))
        {

            //Расчет метрики LM
            double atob = Convert.ToDouble(ex_count["(" + k + ", " + l +
")"]);
            double btoa = Convert.ToDouble(ex_count["(" + l + ", " + k +
")"]);
            double LM_N = atob + btoa;

```



```

double LM_P = atob / (LM_N + 1.0);
double LM = LM_P - 1.96 * Math.Sqrt(LM_P * (1 - LM_P) / (LM_N +
1));

string mer2 = GetKeyOfValue(w_key1, 1); //Получаем название
мероприятия по индексу
double mer1_count = Convert.ToDouble(app_count[mer1]);
//Расчет метрики GM
double GM = (atob - btoa) * Convert.ToDouble(exemplars_count) /
(Convert.ToDouble(app_count[mer1]) * Convert.ToDouble(app_count[mer2]));

//Расчет метрики CM
double CM_C = 0.0;
foreach (ArrayList pair in d_pairs)
{
    if (pair.Contains(mer1) && pair.Contains(mer2))
    {
        int index1 = pair.IndexOf(mer1);
        int index2 = pair.IndexOf(mer2, index1 + 1);
        if (index2 > -1)
            CM_C += Math.Pow(0.8, Convert.ToDouble(index2 -
index1 - 1));
    }
}

double CM = CM_C / Convert.ToDouble(app_count[mer1]);

//Расчет метрики LR
double LR = -8.280 + 6.376 * LM + 4.324 * GM + 8.654 * CM;
double p_ver = Math.Exp(LR) / (1 + Math.Exp(LR));

if (p_ver >= 0.8)
{
    w_val_without_noise[k, 1] = w_val1[k, 1];
    ocount++;
}
}
}
}
}

```

Следующий фрагмент кода реализует процедуру удаления тупиков.

```

//Функция удаления тупиков
/*Удаляются все вершины, у которых отсутствуют входящие или исходящие дуги*/
private KeyValuePair<Hashtable, SparseArrays.Sparse2DMatrix<int, int, int>>
delete_deadlocks(SparseArrays.Sparse2DMatrix<int, int, int> matrix, int count, Hashtable
del_indexes)
{
    int size = count;
    SparseArrays.Sparse2DMatrix<int, int, int> matr = new
SparseArrays.Sparse2DMatrix<int, int, int>(0);
    matrix.CopyTo(matr);
    SparseArrays.Sparse2DMatrix<int, int, int> res = new
SparseArrays.Sparse2DMatrix<int, int, int>(0);
    KeyValuePair<Hashtable, SparseArrays.Sparse2DMatrix<int, int, int>> res_pair
= new KeyValuePair<Hashtable, SparseArrays.Sparse2DMatrix<int, int, int>>();
    bool vertical = false;
    bool horisontal = false;
    int del_index = -1;
    int ccount = count - 1;

    for (int i = 1; i < ccount; i++)
    {

```

```

horizontal = false;
vertical = false;
for (int j = 0; j < count; j++)
{
    if (matrix[i, j] == 1)
        horizontal = true;
    if (matrix[j, i] == 1)
        vertical = true;
    if (horizontal && vertical)
        break;
}
if (!horizontal || !vertical)
{
    string oname = GetKeyOfValue(del_indexes, i);
    if (oname != "last")
    {
        del_index = i;
        break;
    }
}
}
if (del_index > -1)
{
    for (int i = del_index; i < ccount; i++)
    {
        for (int j = 0; j < size; j++)
        {
            matr[i, j] = matr[i + 1, j];
        }
    }
    //удаляем последнюю строку
    for (int i = 0; i < size; i++)
    {
        matr.Remove(ccount, i);
    }
    size--;

    for (int i = 0; i < size; i++)
    {
        for (int j = del_index; j < ccount; j++)
        {
            matr[i, j] = matr[i, j + 1];
        }
    }
    //удаляем последний столбец
    for (int i = 0; i < size; i++)
    {
        matr.Remove(i, ccount);
    }
    string oname = GetKeyOfValue(del_indexes, del_index);
    if (oname != "last")
    {
        Hashtable del_indexes1 = new Hashtable();
        foreach (DictionaryEntry de in del_indexes)
        {
            string skey = (string)de.Key;
            int ival = (int)de.Value;
            if (ival > del_index)
                del_indexes1.Add(skey, ival - 1);
            else
                del_indexes1.Add(skey, ival);
        }
        del_indexes1.Remove(oname);
        del_indexes = del_indexes1;
    }
}

```

```

        res_pair = delete_deadlocks(matr, size, del_indexes);
    }
    else
    {
        res_pair = new KeyValuePair<Hashtable, SparseArrays.Sparse2DMatrix<int,
int, int>>(del_indexes, matr);
    }
    return res_pair;
}

```