Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики
Кафедра информационных и управляющих систем
Направление подготовки /специальность 09.04.04 – Программная инженерия
Направленность (профиль) образовательной программы Управление разработкой программного обеспечения

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой
_____А.В. Бушманов
«_____»_____2023 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему: Программа синтеза моделей процессов с использованием методов кластеризации

Исполнитель
студент группы 157-ом _____ Т. Е. Федосова
(подпись, дата)

Руководитель
профессор, доктор техн. наук _____ А.Д. Плутенко
(подпись, дата)

Руководитель научного
содержания программы
магистратуры
профессор, доктор техн. наук _____ И.Е. Еремин
(подпись, дата)

Нормоконтроль
доцент, канд. техн. наук _____ Л.В. Никифорова
(подпись, дата)

Рецензент
канд. техн. наук _____ Д.С. Щербань
(подпись, дата)

Благовещенск 2023

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики
Кафедра информационных и управляющих систем

УТВЕРЖДАЮ
Зав. кафедрой
_____А.В. Бушманов
«_____»_____ 2023 г.

З А Д А Н И Е

К магистерской диссертации студента Федосовой Татьяны Евгеньевны

1.Тема выпускной квалификационной работы: Программа синтеза моделей процессов с использованием методов кластеризации

(Утверждено приказом от 21.02.2023 № 442-уч)

2.Срок сдачи студентом законченной работы (проекта) 23.06.2023 г.

3.Исходные данные к выпускной квалификационной работе: предметная область, отчеты по практической подготовке

4.Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов): анализ предметной области проводимого исследования, алгоритмическое и программное обеспечение решения поставленной задачи, разработка и реализация плагина к системе ProM

5.Дата выдачи задания: 30.01.2023 г.

6.Руководитель выпускной квалификационной работы: Плутенко Андрей Долиевич, профессор, доктор техн. наук

(фамилия, имя, отчество, должность, уч. степень, уч. звание)

Задание принял к исполнению (дата):_____
(Подпись студента)

РЕФЕРАТ

Магистерская диссертация содержит 79 с., 30 рисунков, 45 источников.

PROCESS MINING, ЛОГИ СОБЫТИЙ, PROM, ПОИСК РАЗЛИЧИЙ, ОРИЕНТИРОВАННЫЕ ГРАФЫ, МОДЕЛИРОВАНИЕ И АНАЛИЗ ПРОЦЕССОВ

В современном мире нас повсюду окружают данные. Все действия на веб-сайтах, в социальных сетях и программных системах записываются, и с каждым годом данных становится больше и больше. Анализ таких данных является ключом к улучшению взаимодействия с пользователем и устранению тупиковых ситуаций из процессов. Интеллектуальный анализ процессов (Process mining) направлен на построение точных и компактных моделей из журналов событий.

В рамках данной работы стоит задача нахождения различий в логах событий. Для того, чтобы эксперт, который работает над анализом процессов, мог легко воспринимать полученную информацию о найденных различиях в логах событий, результаты работы должны быть представлены в хорошо читаемом и понятном пользователям виде.

Данная работа позволит повысить эффективность обработки больших наборов данных алгоритмами process mining с целью построения моделей процессов. Это позволит улучшить применимость алгоритмов, к большим журналам событий реальных информационных систем. Использование таких алгоритмов с моделями, которые получены на основе реальных журналов событий, дает возможность более комплексного изучения поведения систем, для которых выполняется построение процессных моделей.

Объектом исследования являются бизнес-процессы. Процессы представляются различными абстракциями в виде статистических,

графических и других видов моделей, а также конкретными выполненными экземплярами процессов, записанными в логах событий.

Цель работы заключается в разработке алгоритма, при помощи которого можно находить различия в логах событий до одного событий: удаленное (пропущенное) событие, добавленное (лишние) событие, измененное событие. Также есть возможность определять полностью идентичные последовательности событий.

Достоверность полученных результатов обеспечивается сравнением с результатами смежных алгоритмов и методов. Результаты находятся в соответствии с результатами, полученными другими авторами.

Методологическую основу исследования составляют теория множеств, теория графов, теория алгоритмов и математическая логика, теория формальных языков и грамматик, принципы машинного обучения, моделирование, синтез и анализ.

Практическое значение работы состоит в разработке прикладной программы, предназначенной для визуализации найденных различий в логах событий.

Результаты научно-исследовательской работы докладывались и обсуждались на научно-методических семинарах кафедры ИУС АмГУ. За время работы над темой научного исследования были опубликованы 2 научные работы.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время очень широко развивается область process mining, идея которой лежит в выявлении, отслеживании, улучшении реальных бизнес-процессов с помощью полученных знаний из логов событий. Такие логи сегодня широко распространены и доступны в современных информационных системах.

Помимо техник проверки соответствия между логом и моделью, существует необходимость в оценки моделей процессов и логов событий. С помощью техник оценки моделей процессов или логов событий появляется возможность определить одинаковое или похожее поведение различных моделей или логов.

Актуальность темы исследования определяется тем, что область process mining стремительно развивается, охватывая практически все сферы нашей жизни в настоящее время многие ведущие компании усиливают работу в направлении развития инновационного продукта, представляющего собой метод анализа процессов, направленный на обнаружение, мониторинг и улучшение реальных процессов путем извлечения информации из журналов событий, содержащихся в информационных системах организации.

В рамках данной работы стоит задача нахождения различий в логах событий. Для того, чтобы эксперт, который работает над анализом процессов, мог легко воспринимать полученную информацию о найденных различиях в логах событий, результаты работы должны быть представлены в хорошо читаемом и понятном пользователям виде.

Магистерская диссертация направлена на развитие и программную реализацию алгоритма нахождения различий в журналах событий до одного события: удаленное (пропущенное) событие, добавленное (лишние) событие, измененное событие с дальнейшим анализом полученных данных.

Предметом исследования выступают проектирование и разработка программы для визуализации найденных различий в журналах событий.

Объектом исследования являются модели процессов. Процессы представляются различными абстракциями в виде статистических, графических и других видов моделей, а также конкретными выполненными экземплярами процессов, записанными в логах событий.

Целью работы является разработка алгоритма поиска различий в журналах событий основанного на алгоритме поиска в ширину на графе с некоторыми модификациями относительно данной задачи, а также реализация разработанного алгоритма в качестве плагина для системы ProM.

В связи с этим ставятся следующие задачи:

– изучение теоретических основ process mining и возможных способов проверки соответствия и оценки моделей процессов и логов событий.

– разработка алгоритма для сравнения логов событий между собой.

– осуществить проектирование программного продукта;

– разработать программный продукт;

– провести тестирование программного продукта на реальном примере;

– провести оценку качества и надежности программного продукта;

Научно-практическая значимость данной работы заключается в том, что результаты работы могут быть использованы экспертами по работе с бизнес-процессами на основе логов событий для их сравнения, анализа, оптимизации, улучшения. Также, полученные результаты могут быть использованы при работе над созданием и разработкой различных алгоритмов и методов сравнения логов событий.

За время работы над темой научного исследования опубликованы 2 научных работы.

Структура работы состоит из введения, трех глав, заключения, библиографических ссылок и библиографического списка.

1 ОБЩАЯ ХАРАКТЕРИСТИКА ИССЛЕДУЕМОЙ ЗАДАЧИ

1.1 Предметная область и объект проводимого исследования

Process Mining является основным направлением исследований, проводимых в лаборатории процессно-ориентированных информационных систем. Этот термин трудно перевести на русский дословно, обычно мы называем это «извлечение и анализ процессов». Дисциплина Process Mining стоит на стыке между исследованием бизнес-процессов, их моделированием, Data Mining, машинным обучением и вовлекает в свою орбиту множество подходов для работы с «большими данными».

Фундаментальная идея, лежащая в основе Process Mining, заключается в том, что сопровождение большинства процессов — технологических, в бизнесе, социальной сфере, медицине и т.д. — организовано с помощью информационных систем, которые в результате своей работы ведут так называемые «журналы или логи событий», в которых фиксируется информация о том, как система функционирует в действительности: например, оплачен чек, выставлен счет или назначено обследование. Этих состояний накапливается очень много, и на их основе можно наблюдать реальное поведение системы. В итоге выявляются несоответствия между тем, как бизнес-процессы работают в действительности, и тем, как они были запланированы изначально [1].

В настоящее время информационные системы все больше и больше становятся связанными с операционными процессами, которые эти системы поддерживают. В результате чего, множество событий записываются современными информационными системами. Несмотря на это, у многих организаций часто возникают проблемы с извлечением полезной информации из этих данных.

Process mining (извлечение процессов) — это общее название ряда методов, подходов и техник, которые предназначены для анализа и усовершенствования бизнес-процессов на основе изучения журналов событий.

Чаще всего техники и методы process mining используются в тех случаях, когда формальное описание или модель системы отсутствуют или качество существующей документации, находится на очень низком уровне [2].

Process mining является относительно молодой исследовательской дисциплиной, которая располагается между машинным обучением и извлечение данных с одной стороны и моделированием процесса, и анализом с другой. Как показано на рисунке 1, process mining устанавливает связь между реальными процессами и их данными и моделированием процесса [3].

Цель Process Mining – автоматическое формирование точного представления о реальном ходе процесса. Журнал событий (event log) и модель процесса – это два ключевых артефакта, используемых в Process Mining. Журнал событий хранит информацию о датах выполнения событий процесса, записанных информационной системой. Модель процесса используется для визуализации.

Методы Process Mining предполагают, что данные исполнения процесса хранятся в журнале событий. В журнале событий информация хранится как последовательность событий [4].

Модели процесса используются для описания поведения процесса организации, для достижения ряда различных целей: установления взаимосвязи между стейкхолдерами, усовершенствования процесса, управления процессом, автоматизации процесса, поддержки исполнения процесса. Чаще всего модель процесса используется для сравнения as-is процесса с to-be процессом, проверки соответствия процесса нормативным требованиям (как, например, ISO 9001), анализа проблем, связанных с производительностью, узких мест (bottlenecks) и других неэффективностей.

1.2 Концепции process mining

В рамках Process Mining можно выделить три концепции: выявление (discovery), согласование (conformance) и улучшение (enhancement). На рисунке 1 изображен обзор этих составляющих с точки зрения реального течения процесса и информационных систем организации. На рисунке 2

данные концепции представлены в формате входных и выходных дынных. Как показано на рисунке 1, методы Process Mining используют журналы событий и модели процесса для представления реального течения процесса. Основная цель Process Mining заключается в выявлении точной и понятной модели процесса, основанной исключительно на данных, записанных в журнал событий. Данные модели могут быть использованы для понимания и улучшения реального течения процесса.

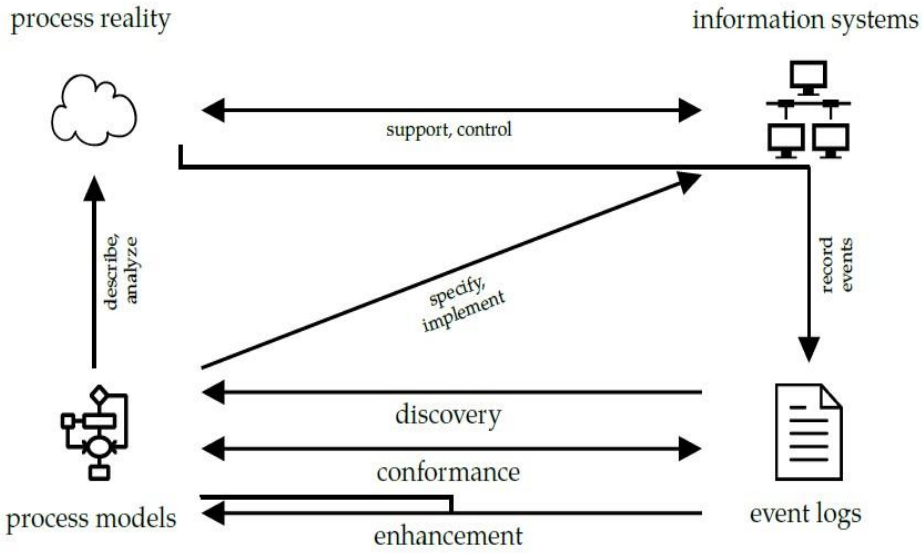

Рисунок 1 – Обзор трех главных концепций Process Mining

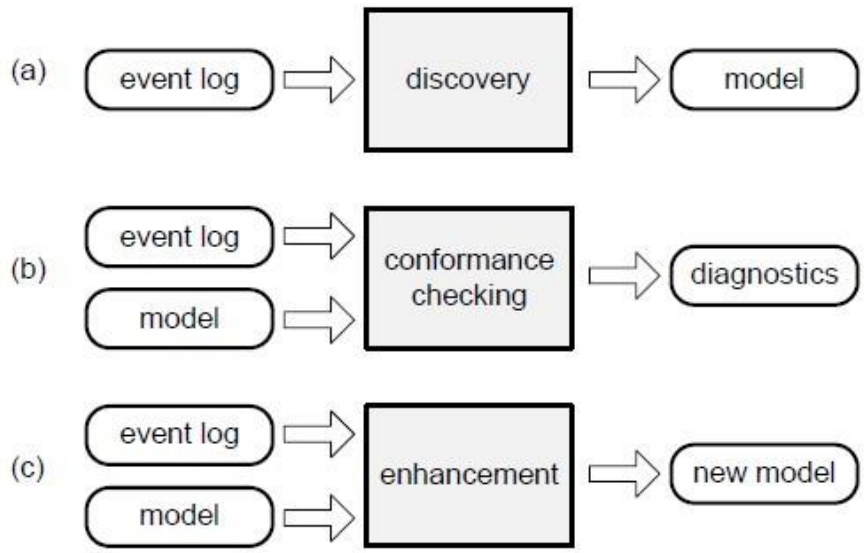

Рисунок 2 – Три концепции Process Mining, описанные со стороны входящих и исходящих данных.

Как правило, Process Mining не ограничен оффлайн-средой. Он также включает методы составления предсказаний и рекомендаций на основе текущих данных процесса в онлайн-режиме.

Методы process mining разделяются на три главных типа и журнал событий может быть использован для всех трех типов.

– Извлечение

Извлечение данных является неотъемлемой частью любого исследования процессов. На основе журнала событий, который содержит запись функционирования информационной системы за некоторый промежуток времени, создается модель процесса [5].

К сожалению, большинство информационных систем хранят необходимую информацию в неструктурированной форме, например, логи событий могут быть размещены сразу в нескольких таблицах. В таких случаях для извлечения данных необходимо приложить некоторые усилия.

В качестве примера рассмотрим ситуацию, когда все события выполняются последовательно и каждое событие соотносится с определенной активностью в процессе. Этап процесса является уникальным в рамках процесса и состоит из цепочки событий (последовательности событий). Примером такой цепочки событий может быть следующая: запрос регистрации, проверка билета, принятие решения, повторный запрос, проверка билета, принятие решение, оплата. В этом примере встречаются два одинаковых события: «проверка билета» и «принятие решение» по два раза каждый. Очевидно, что является очень важным умение различать эти события. По этой причине большинство логов событий хранят дополнительную информацию о событиях (например, ресурс события — человек или устройство, временные метки, дополнительные элементы данных) и когда это возможно, эта дополнительная информация используется техниками process mining.

Например, извлечение информации может быть осуществлено с помощью альфа-алгоритма, с помощью которого по существующего логу событий строится модель процесса в виде сети Петри, по которой можно наблюдать поведение процесса.

– Проверка соответствия (Conformance analysis)

Проверка соответствия заключается в сравнении существующей модели процесса с журналом событий, анализе выявленных несоответствий в поведении реальной системы и моделируемом поведении.

Выявлять расхождения между смоделированным и наблюдаемым поведением помогают алгоритмы проверки соответствия. Они выдают показатели степени соответствия и диагностические сведения, объясняющие наблюдаемые различия. С их помощью можно детально анализировать прецеденты, не соответствующие построенной модели.

Соответствие модели определяется только для конкретного журнала событий. Для проведения проверки соответствий требуется, помимо лога событий, некоторая заранее определенная модель. Это модель может быть построена вручную или получена с помощью методов извлечения. Для расчета соответствия журнал событий последовательно исполняется моделью.

Трасса журнала событий считается соответствующей модели, если может быть получена в результате исполнения этой модели. Журнал событий идеально соответствует модели, если все его трассы соответствуют модели. На практике редко встречаются пары журнал–модель с идеальным соответствием. Поэтому обычно журнал считают соответствующим модели, если некоторая доля трасс в журнале соответствует модели. Значение уровня отсечки задается экспертом в каждом конкретном случае.

Анализ и интерпретация выявленных несоответствий зависит от типа модели:

– для дескриптивной (описательной) модели выявленные расхождения между моделью и журналом событий указывают на необходимость улучшения самой модели.

– для нормативной модели такие отклонения могут указывать на необходимость лучшего контроля процесса, а также не необходимость проверки неэффективных исполнителей.

Технологии проверки соответствия могут применяться при аудиторских проверках, оптимизации процессов и контроле исполнения нормативных актов.

– Усовершенствование

Идея усовершенствования состоит в расширении и улучшении существующей модели процесса с помощью расширения моделируемого поведения или повышения эффективности моделирования с использованием дополнительной информации, полученной в ходе проверки соответствия модели и журнала событий. В то время как проверка соответствия направлена на измерение несоответствий между моделью и реальностью, усовершенствование нацелено на изменения или расширение модели [6].

У методов process mining – проверки соответствия и оценки модели процессов очень много общего. Как уже было написано ранее, проверка соответствия заключается в анализе сравнения журнала событий и описанного поведения в разработанной модели, в то время как оценка модели процессов определяет сходства между поведением в различных моделях процесса. Например, с помощью оценивая моделей процессов можно ответить на вопрос «Какая из моделей лучше?».

Помимо журнала событий, для проведения проверки соответствий необходима заранее определенная модель процесса. Эта модель может быть построена вручную или получена с помощью методов извлечения. Независимо от источника, система ProM (Process Mining Framework)

предоставляет различные способы проверки действительно ли лог соответствует этой модели.

Существует несколько распространенных заблуждений, связанных с process mining. Ряд вендоров, аналитиков и ученых узко трактуют process mining как один из методов глубинного анализа данных (data mining) для извлечения процесса, используемый исключительно для анализа в режиме офлайн. Это неверно, и поэтому ниже мы подчеркнем следующие три характерные особенности.

– Process mining не ограничивается извлечением моделей потока управления. Извлечение моделей процесса из журналов событий разжигает воображение как специалистовпрактиков, так и представителей академического сообщества. Поэтому извлечение моделей потока управления зачастую рассматривается как наиболее примечательная составляющая process mining. Однако, process mining не ограничен извлечением потока управления. С одной стороны, извлечение модели – это лишь одна из трех основных форм process mining (извлечение, подтверждение и усовершенствование). С другой стороны, область применения не сводится только к потоку управления: важную роль также играют организационный, временной ракурсы и ракурс кейсов.

– process mining не является просто еще одним видом data mining. Process mining может рассматриваться как «недостающее звено» между data mining и традиционным управлением процессами (BPM). Большинство методов data mining вовсе не процессно-ориентированы. Модели процессов, потенциально демонстрирующие параллелизм, не сравнимы с простыми структурами data mining, такими как дерево принятия решений и ассоциативные правила. Поэтому необходимы совершенно новые алгоритмы и формы представления данных.

– **p**rocess mining не ограничен офлайн-анализом. Методы process mining извлекают знания из накопленных данных о событиях. Несмотря на то, что используются данные постфактум, результаты применимы и к тем кейсам,

которые еще находятся в обработке. Например, используя выявленную модель процесса, можно спрогнозировать время завершения обработки заказа клиента.

Для проведения проверки соответствий требуется, помимо лога событий, некоторая заранее определенная модель. Это модель может быть построена вручную или получена с помощью методов извлечения. Независимо от источника, система ProM предоставляет различные способы проверки действительно ли лог соответствует этой модели.

На рисунке 3 изображена основная идея проверки соответствия: сравнение поведения, записанного в логе событий и поведения модели процесса с целью поиска расхождения и общего. Результаты анализа сравнения модели и лога события делятся на глобальные меры соответствия (например, 85% всех случаев в логе события могут быть воспроизведена на построенной модели) и локальные диагностики (например, событие x было исполнено 15 раз согласно логу, хотя это оказывается невозможным согласно модели).

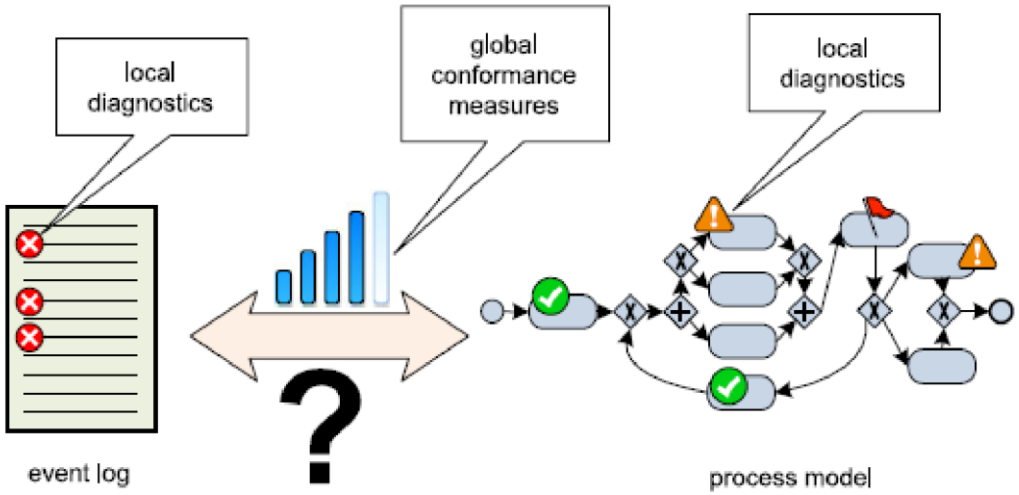

Рисунок 3 – Проверка соответствия: сравнение рассматриваемого поведения с модельным

Большинство информационных систем используют логи событий для аудита и мониторинга процессов, которые эти системы поддерживают. Данные, извлеченные с помощью техник process mining могут быть использованы для построения моделей, описывающих процесс и то, как этот бизнес-процесс будет выполняться. Вместе с данными, которые записаны в лог, эта ситуация обращает наше внимание на следующий вопрос: «Соответствуют ли друг другу лог событий и модель процесса?».

У техник проверки соответствия и техник оценки моделей процессов очень много общего. Проверка соответствия заключается в проверке того, насколько данные в логе событий соответствуют описанному поведению с разработанной модели, в то время как оценка моделей процессов (process model evaluation) вычисляет тот факт, что многие модели, возможно, имеют одинаковое или похожее поведение (например, это может служить ответом на вопрос, «Какая их моделей является лучшей?»)

1.3 Журнал событий

Журнал событий содержит данные из процесса, и каждое событие в этом журнале соответствует одному экземпляру процесса, известному как обращение или трассировка. Обращение — это набор событий, а событие включает в себя атрибуты, описывающие то, что произошло в процессе.

B(X) обозначает множество всех мультимножеств над некоторым множеством X. Для некоторого мультимножества $b \in$ B(X), $b\ (x)$ обозначает число раз, которое элемент $x \in$ X появляется в b. Таким образом, $x \in$ b тогда и только тогда, когда $b\ (x) > 0$. Через $b = [x_1, x_2^3, x_3^5]$ будем обозначать, что элементы x_1, x_2, $x_3 \in$ X появляются в b один, три и пять раз соответственно. Далее, будем писать $b' \subseteq b$ тогда и только тогда, когда $\forall\ x \in$ X: $b'\ (x) \leq b$ (x). Размер мультимножества b над множеством X обозначим как $|\ b\ |$ и определим $|\ b\ | = \sum_{x \in X} b(x)$.

Для заданного множества Y обозначим через Y + множество всех непустых конечных последовательностей, составленных из элементов множества Y

Пусть A — множество активностей. Трасса (событий) представляет собой последовательность $\sigma = \langle a_1, a_2, \dots, a_i, \dots, a_n \rangle \epsilon A^+$. С помощью $\sigma(i) = a_i$ мы обозначаем i-й элемент (событие) трассы. $[i, k]$ −подтрасса трассы σ, заканчивающейся i -м событием a_i, обозначается следующим образом:

$$\sigma[i,k] = \begin{cases} \langle \sigma(1), \sigma(2), \dots, \sigma(i) \rangle, \ if\ k > i \\ \langle \sigma(i-k+1), \dots, \sigma(i) \rangle, \ if\ 1 \le k \le i, \\ \langle\ \rangle, \ if\ k = 0 \end{cases} \quad (1.1)$$

Полная подтрасса трассы σ, заканчивающаяся i-м событием, a_i это: σ$[i]$= σ $[i, i]$. Длину трассы мы обозначаем с помощью |σ|. При k ≤ i, k задает длину подтрассы.

Пример журнала событий, содержащего 7 (классов) активностей, 8 трасс и 41 событие:

$$L1 = [\langle a,b,c,d,e,f \rangle, \langle a,b,c,d,e,g \rangle, \langle a,b,c,d,f,e \rangle,$$
$$\langle a,b,c,d,f,g \rangle, \langle a,b,d \rangle, \langle a,b,d,g \rangle, \langle a,b,d,e,f \rangle, \langle a,b,d,e,g \rangle] \quad (1.2)$$

Предполагаем, что журналы событий не содержат состояния процессов в явном виде.

Помеченная система переходов — это кортеж $TS = (S, E, T, s_0, AS)$, где S — множество состояний, E — множество пометок, $T \subseteq S \times E \times S$ — множество переходов, $s_0 \in S$ — начальное состояние и AS ⊆ S — множество принимающих (конечных) состояний. Мы определяем множество выходящих (входящих) переходов состояния $s \in S$ как $s\bullet = \{t = (s, e, s') \in T \mid e \in E, s' \in S\}$ ($\bullet s = \{t = (s', e, s) \in T | e \in E, s' \in S\}$).

Под $TS = (S, E, T, s_0, AS)$, где $L \in B(A^+)$ это журнал событий, мы обозначим систему переходов, для которой E = A.

Пусть $\sigma = \langle a_1, \dots, a_n \rangle$ — это трасса ($\sigma \in L$) и n = |σ|. Будем считать, что трасса σ может быть воспроизведена в этой системе переходов TS(L), если существует последовательность состояний $\langle s_0, \dots, s_n \rangle$ такая, что $\exists t1 = (s_0, a_1, s_1), t2 = (s_1, a_2, s_2), \dots, t_n = (s_{n-1}, a_n, s_n)$, где $s_0, s_1, \dots, s_n \in S, t_1, t_2, \dots, t_n \in T$. Будем обозначать это следующим $s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} \dots \xrightarrow{a_n} s_n$.

Также будем считать, что трасса σ может быть частично воспроизведена по ее префиксу в системе переходов TS(L), если $\exists k < n$: $s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} \dots \xrightarrow{a_k} s_k$ и $\nexists s_k \xrightarrow{a_{k+1}} s_{k+1} \xrightarrow{a_{k+2}} \dots \xrightarrow{a_n} s_n$.

Через $\sigma^+(TS(L)) = \langle a_0, a_1, \dots, a_k \rangle$ обозначим воспроизводимую часть трассы и через $\sigma^-(TS(L)) = \langle a_{k+1}, \dots, a_n \rangle$ соответственно невоспроизводимую часть трассы. Тогда, σ(TS(L)) = σ^+ (TS(L)) + σ^- (TS(L)) где + обозначает конкатенацию двух последовательностей.

1.4 Обзор существующих методов решения аналогичных задач

Рассмотрим основные методы проверки соответствия, разработанные и реализованные на текущий момент.

1.4.1 Плагин «Conformance Checker»

Один из способов осуществления проверки соответствия реализован в рамках плагина для системы ProM «Conformance Checker», с помощью которого можно осуществить проверку соответствия между логом событий и моделью процесса, представленного в виде сети Петри [8].

Перед началом фактического анализа можно выбрать, какой вид анализа следует выполнить. Можно выбирать и отменять выбор целых категорий или конкретных показателей. Краткое описание объясняет каждую из этих опций. Средство проверки соответствия поддерживает анализ пригодности, Точности (или поведенческой уместности) и структуры (или структурной уместности) измерения. Для расчета показателей соответствия используются следующие методы анализа:

– Воспроизведение журнала. Воспроизведение журнала выполняется неблокирующим способом и с точки зрения журнала, т. Е. каждая

трассировка журнала будет воспроизводиться в модели, и, если для запуска соответствующего перехода не хватает токенов, они создаются искусственно, и воспроизведение продолжается. При этом собираются диагностические данные, к которым впоследствии можно получить доступ. Используется для вычисления показателей f, pSE, pPC, saB и aaB.

– Анализ пространства состояний. График охвата модели процесса просматривается, в то время как циклы выполняются не более двух раз. Используется для вычисления как метрик aaB, так и aaS.

– Анализ пространства состояний. График охвата модели процесса просматривается, в то время как циклы выполняются не более двух раз. Используется для вычисления как метрик aaB, так и aaS.

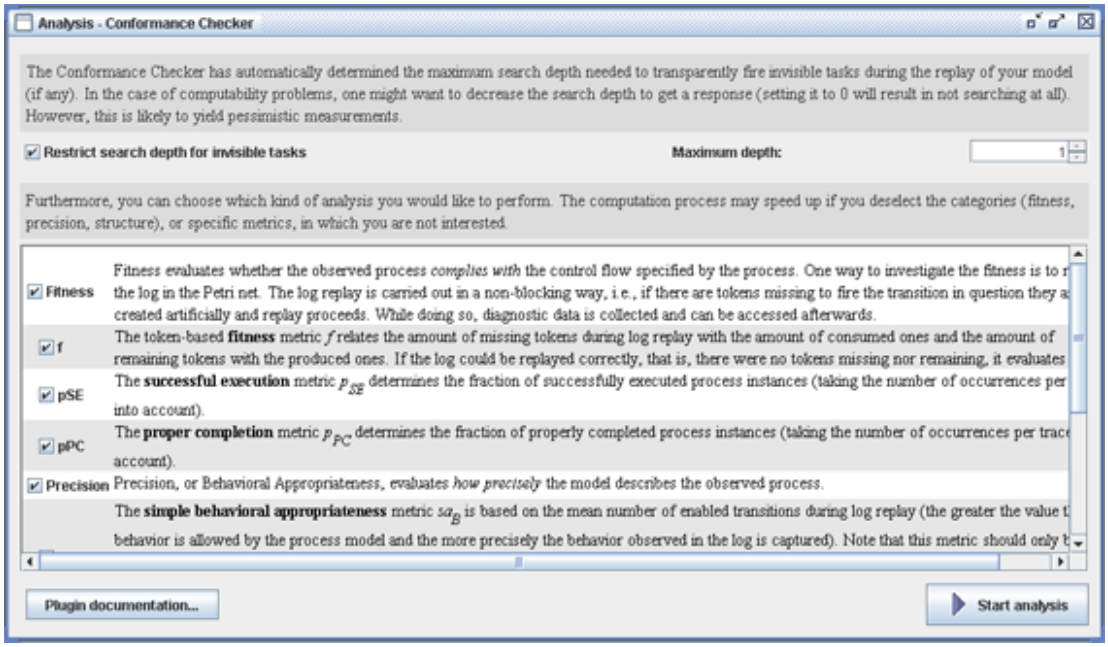

Рисунок 4 – Выбор показателей для расчета

Как правило, настройки по умолчанию должны быть в порядке для анализа (вы можете прервать каждый тип анализа, если он занимает слишком много времени). Однако вот несколько советов, если при анализе вашего журнала у вас возникнут проблемы с производительностью:

Может быть целесообразно вычислять показатели в отдельных сеансах (поскольку может оказаться, что один довольно дорогой метод излишне

ограничивает вычисление других, более совершенных методов масштабирования).

Также можем дополнительно ограничить глубину поиска невидимых задач во время воспроизведения журнала. Это означает, что, если воспроизводимая в данный момент задача напрямую не включена, мы создаем частичное пространство состояний на основе текущей маркировки, чтобы выяснить, можно ли ее включить с помощью некоторой последовательности невидимых задач. Хотя глубина поиска, необходимая для корректного воспроизведения вашей модели, автоматически определяется программой проверки соответствия, вы можете дополнительно ограничить ее, чтобы получить результат. Например, если установить максимальную глубину поиска равной 0, пространство состояний вообще не создается.

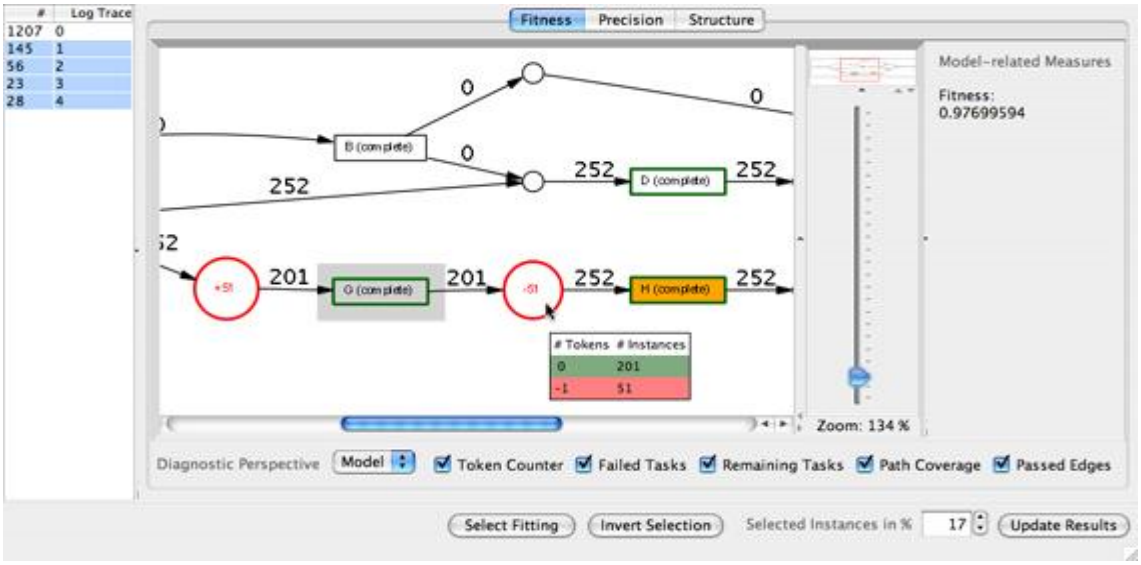

Рисунок 5 – Вид модели показывает места в модели, где возникали проблемы во время воспроизведения журнала

Существует ряд опций, которые можно использовать для улучшения визуализации модели процесса путем указания:

– Счетчик токенов. Визуализирует недостающие и оставшиеся токены во время воспроизведения журнала для каждого места. Это позволяет локализовать те части модели, в которых имело место несоответствие, если таковое имело место.

– Неудачные задачи. Визуализирует переходы, которые не были включены (то есть не готовы) во время воспроизведения журнала и, следовательно, не могли быть успешно выполнены.

– Оставшиеся задачи. Визуализирует переходы, которые оставались включенными после воспроизведения журнала, что указывает на неправильное завершение указанного процесса и намекает на то, что эта задача должна была быть выполнена.

– Пройденные грани. Указывает на каждом из ребер, как часто оно выполнялось во время воспроизведения данных экземпляров процесса.

Перспектива диагностики может быть изменена для визуализации файла журнала или подмножества трассировок журнала соответственно. Следующие показатели рассчитываются с точки зрения журнала, чтобы измерить степень пригодности:

– Успешное выполнение. Доля успешно выполненных экземпляров процесса (с учетом количества вхождений в трассировку).

– Надлежащее завершение. Доля правильно завершенных экземпляров процесса (с учетом количества вхождений в трассировку).

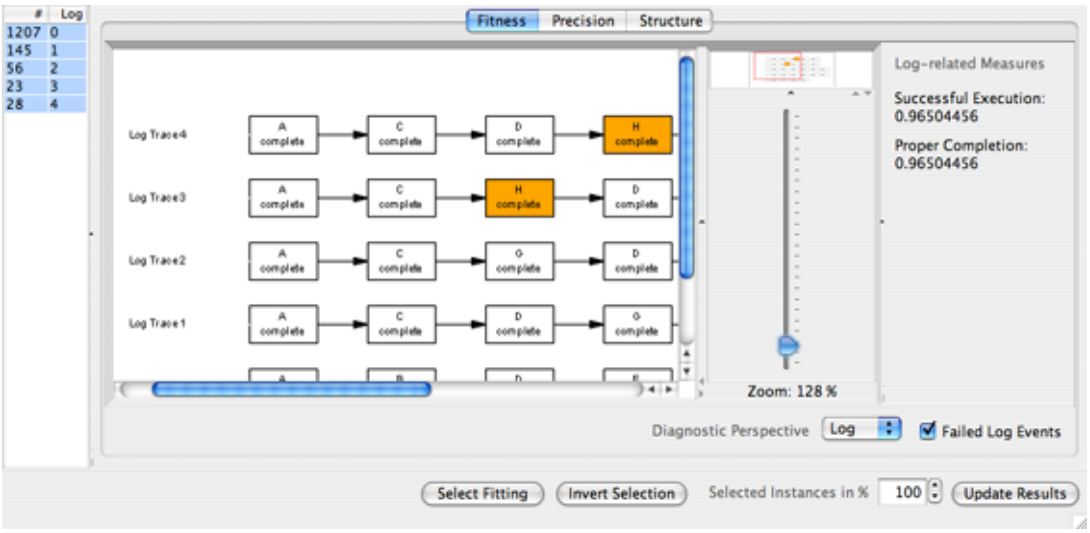

Рисунок 6 – Просмотр журнала показывает, где в журнале возникали проблемы с воспроизведением

С другой стороны, модель процесса может допускать большее поведение, чем то, которое записано в журнале. Мы называем анализ и обнаружение такого “дополнительного поведения” поведенческой уместностью, или измерением точности, т. е. точность составляет 100%, если модель “точно” учитывает поведение, наблюдаемое в журнале. Таким образом, можно, например, обнаружить альтернативные ветви, которые никогда не использовались при выполнении процесса.

Для измерения степени соответствия поведения доступны следующие показатели:

– Простая поведенческая целесообразность. Простая мера соответствия поведения saB, основанная на среднем количестве разрешенных переходов во время воспроизведения журнала (чем больше значение, тем меньшее поведение допускается моделью процесса и тем точнее фиксируется поведение, наблюдаемое в журнале). Обратите внимание, что этот показатель следует использовать только в качестве сравнительного средства для моделей без альтернативных повторяющихся задач.

– Простая поведенческая целесообразность. Простая мера соответствия поведения saB, основанная на среднем количестве разрешенных переходов во время воспроизведения журнала (чем больше значение, тем меньшее поведение допускается моделью процесса и тем точнее фиксируется поведение, наблюдаемое в журнале). Обратите внимание, что этот показатель следует использовать только в качестве сравнительного средства для моделей без альтернативных повторяющихся задач.

Кроме того, существует ряд опций, которые можно использовать для улучшения визуализации модели процесса путем указания:

– Всегда предшествует. Визуализирует те действия, которые всегда предшествовали друг другу в журнале, но только иногда предшествуют друг другу в соответствии с моделью.

– Никогда не предшествует. Визуализирует те действия, которые никогда не предшествовали друг другу в журнале, но иногда действительно

предшествуют друг другу в модели (см. рисунок 7, где последовательность выполнения GG никогда не выполнялась, хотя это было бы разрешено моделью).

– Никогда не предшествует. Визуализирует те действия, которые никогда не предшествовали друг другу в журнале, но иногда действительно предшествуют друг другу в модели (см. рисунок 7, где последовательность выполнения GG никогда не выполнялась, хотя это было бы разрешено моделью).

– Никогда не следует. Визуализирует те действия, которые никогда не следовали друг за другом в журнале, но иногда следуют друг за другом в модели (см. рисунок 7, где последовательность выполнения GG никогда не выполнялась, хотя это было бы разрешено моделью).

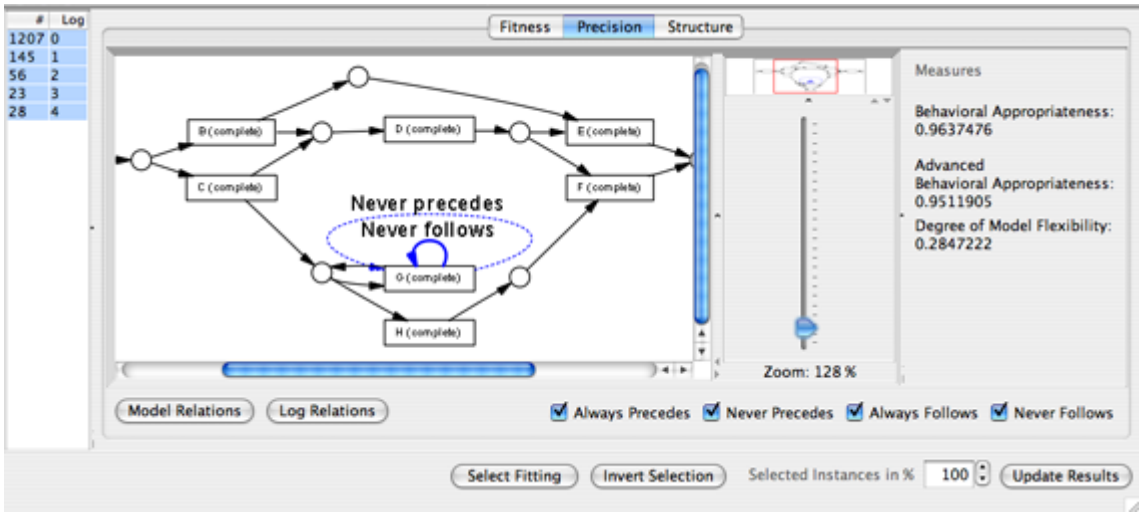

Рисунок 7 – Анализ точности модели позволяет обнаружить общие части

В модели процесса структура — это синтаксическое средство, с помощью которого можно задать поведение (то есть семантику), используя словарь языка моделирования (например, маршрутизирующие узлы, такие как AND или XOR). Однако часто существует несколько синтаксических способов выражения одного и того же поведения, и могут быть “предпочтительные” (например, более простые для понимания) и “менее подходящие” представления. Очевидно, что этот аспект оценки сильно зависит от формализма моделирования процессов, и его трудно оценить объективным

образом (в конце концов, могут быть личные или даже корпоративные предпочтения). Однако можно сформулировать и оценить определенные “рекомендации по проектированию”, такие как требование минимального количества повторяющихся задач в модели.

Для измерения степени структурного соответствия доступны следующие показатели:

– Простая структурная целесообразность. Простой показатель структурной пригодности saS — это простая мера, основанная на размере графика (чем больше значение, тем компактнее модель). Обратите внимание, что этот показатель следует использовать только в качестве сравнительного средства для моделей, допускающих одинаковое поведение.

– Усовершенствованная метрика структурной приемлемости aaS основана на наказании повторяющихся задач, которые используются только для составления списка альтернатив (обнаруживается с помощью анализа пространства состояний), и невидимых задач, которые могут быть удалены без изменения поведения. Если нет альтернативных повторяющихся задач и нет избыточных задач, то метрика оценивается в 1. Если каждая задача в модели является либо дублированием для перечисления альтернативного поведения, либо избыточной, метрика оценивается в 0.

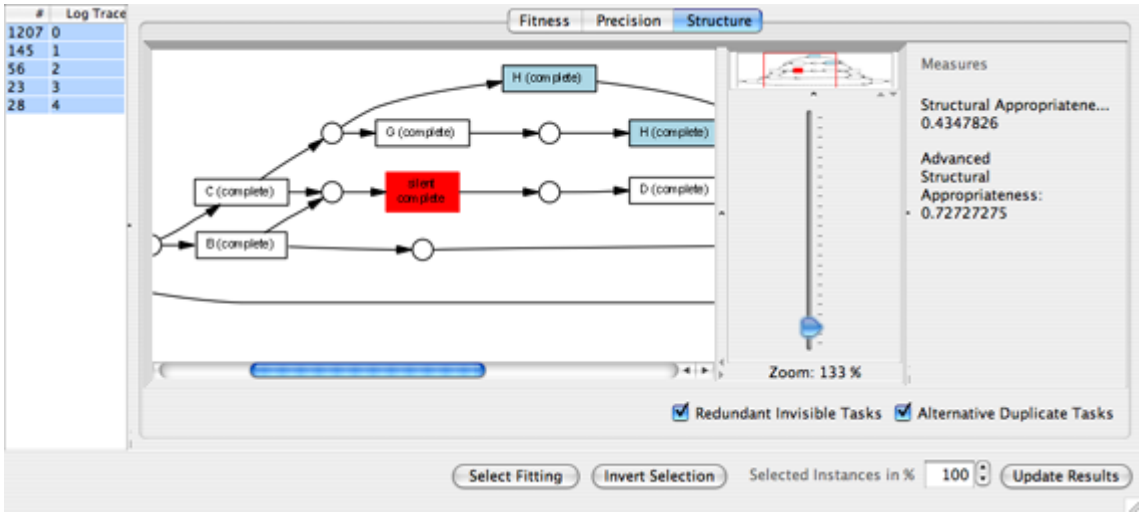

Рисунок 8 – Структурный анализ обнаруживает дублирующуюся задачу, в которой перечислены альтернативные варианты поведения и избыточные задачи

Кроме того, существует ряд опций, которые можно использовать для улучшения визуализации модели процесса путем указания:

– Избыточные невидимые задачи. Визуализирует те повторяющиеся задачи, которые никогда не используются вместе в одном из возможных путей модели. В общем, повторяющиеся задачи могут быть желательны в рамках хорошей модели процесса, например, для выражения того, что одно действие выполняется в начале и в конце процесса. Обратите внимание, что альтернативные повторяющиеся задачи не обнаруживаются средством проверки соответствия, если они содержатся в цикле.

– Альтернативные повторяющиеся задачи. Визуализирует те невидимые задачи, которые не влияют на поведение (на основе этих правил сокращения) и, следовательно, могут быть безопасно удалены из модели, делая ее более компактной. Обратите внимание, что сокращенный файл может быть экспортирован с помощью средства проверки соответствия.

1.4.2 Плагин «LTL Checker Plugin»

Метод используется для проверки соответствия и реализован в рамках плагина «LTL Checker Plugin». В данном случае сравниваются модель процесса и лог событий в случаях, когда существует неполная модель процесса (например, существует лишь некоторый набор требований — так называемых бизнес-правил). Для того, чтобы можно было работать с данным плагином, необходимо использовать журнал событий и соответствующую LTL (линейнотемпоральная логика) модель. После того, как события в логе проверены согласно формулам линейно-темпоральной логике, которые задаются при настройке плагина, результаты сохраняются для дальнейшего использования и анализа. По окончанию сохранения приводится графическое представление результатов. В итоге по результатам работы плагина есть возможность восстановить полную картину происходящих событий.

Анализ соответствия требует, в дополнение к журналу событий, некоторой априорной модели. Эта модель может быть изготовлена вручную или получена с помощью process discovery. Каким бы ни был его источник,

ProM предоставляет различные способы проверки того, соответствует ли реальность такой модели. Например, может существовать модель процесса, указывающая, что для заказов на поставку на сумму более миллиона евро требуются две проверки. Другим примером является проверка так называемого “принципа четырех глаз”. Анализ соответствия может использоваться для обнаружения отклонений, локализации и объяснения этих отклонений, а также для измерения серьезности этих отклонений [9].

– Программа проверки соответствия сравнивает априорную модель с наблюдаемой реальностью, хранящейся в некотором журнале MXML, и визуализирует и количественно оценивает обнаруженные несоответствия.

– Плагин проверки LTL, который можно использовать в случае, если не существует полной априорной модели процесса, а есть только набор требований (например, бизнес-правила). В папке documentation ProM есть руководство, в котором описывается, как их можно использовать.

– Средство проверки семантического LTL позволяет включать семантическую информацию из онтологий.

Чтобы сравнить результаты, полученные с помощью различных алгоритмов обнаружения процессов, или просто выяснить, насколько хорошо обнаруженная модель отражает наблюдаемое поведение (сколько случаев фактически охвачено и т.д.), необходимы методы оценки.

– Ряд показателей, которые оценивают качество (обработанной) модели, разрабатывались различными исследователями в области интеллектуального анализа процессов на протяжении многих лет, и подключаемый модуль Control Flow Benchmark объединяет те, которые реализованы в ProM.

– Средство проверки соответствия предоставляет диагностические визуализации для некоторых показателей в подключаемом модуле Control Flow Benchmark.

– Подключаемый модуль анализа минимальной длины описания оценивает модели процессов на основе принципа MDL, известного из области машинного обучения.

– Подключаемый модуль поведенческой точности / отзыва (показатели также доступны в подключаемом модуле Control Flow Benchmark)

– Подключаемый модуль структурной точности / отзыва (показатели также доступны в подключаемом модуле Control Flow Benchmark)

1.4.3 Плагин «Trace Diff Analysis»

Один из методов сравнения реализован в качестве плагина «Trace Diff Analysis» к инструменту ProM. Данный плагин используется с целью сравнений между собой двух трасс разных или одного и того же логов событий и визуализации, найденных различия.

Визуализация различий между двумя выбранными трассами событий из лога событий основана на стандартном алгоритме поиска наибольшей общей подпоследовательности, который также используется в утилите diff для сравнения файлов в системах контроля версий.

В рамках данного плагина делается допущение, что два события являются одинаковыми, если их типы и имена полностью совпадают. На вход плагину поступают логи событий (один или два), последовательности событий которых отображаются в специальном меню для каждого из логов в отдельном списке.

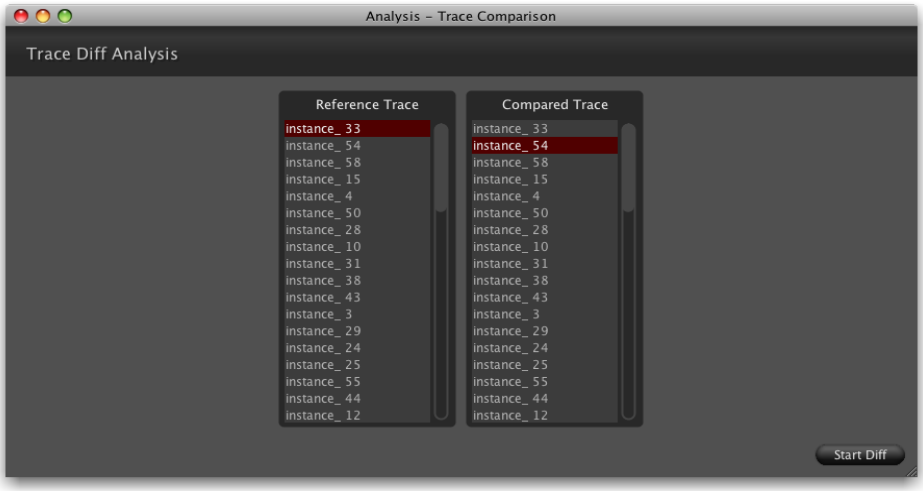

Рисунок 9 – Плагин «Trace Diff Analysis – выбор трасс для сравнения

После того, как выбраны две трассы событий, которые необходимо сравнить, запускается процесс анализа сравнений меду двумя выбранными трассами. Результаты показаны графически для удобства восприятия, как показано на рисунке 10. По данным результатам можно отследить все различия и сходства.

Этот алгоритм заключается в следующем. Предположим, что существует некоторый образец — короткая строка и текст — длинная. В рамках алгоритма пытаемся выяснить, появляются ли буквы образца в том же порядке, но, возможно, на разном расстоянии в тексте. Если да — то образец является подпоследовательностью текста.

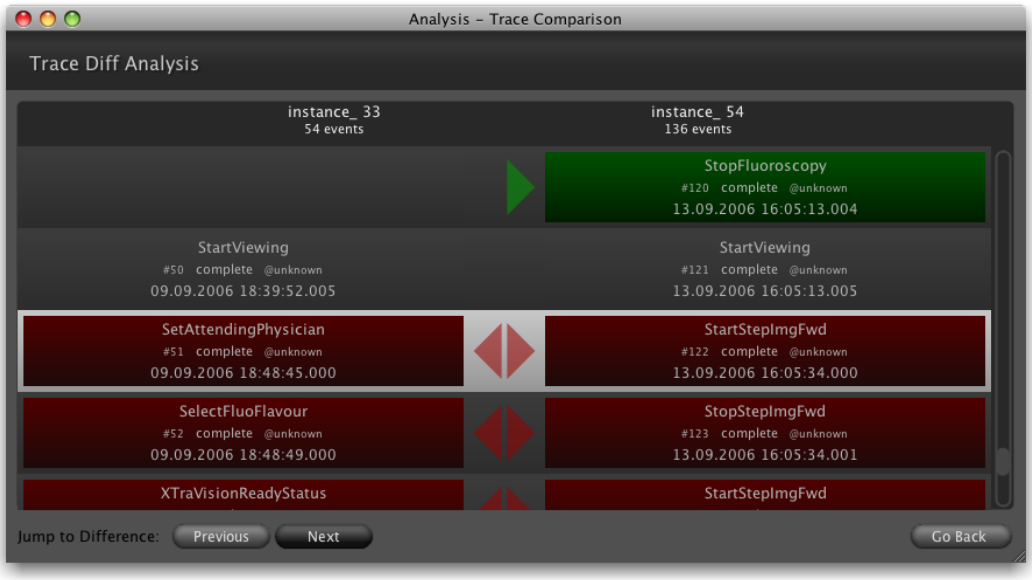

Рисунок 10 – Результат работы плагина «Trace Diff Analysis»

Результат сравнения визуализируется графически, и можно переходить от различия к различию, используя кнопки “предыдущий” и “следующий”.

2 АЛГОРИТМИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ РЕШЕНИЯ ЗАДАЧИ

2.1 Предлагаемый алгоритм компьютеризированного решения задачи

Сравнение двух логов между собой является довольно большой задачей и требуется большого количество ресурсов таких как время. Так как исходные логи могут быть достаточно объемными, для их сравнения может понадобится много времени. В данном случае получается, что чем больше событий содержится в логе, тем больше времени необходимо потратить на процесс сравнения логов между собой.

Поиск в ширину (англ. Breadth-First Search, BFS) позволяет вычислить кратчайшие расстояния (в терминах количества рёбер) от выделенной вершины ориентированного графа до всех остальных вершин, и/или построить корневое направленное дерево, расстояния в котором совпадают с расстояниями в исходном графе. Кроме того, поиск в ширину позволяет решать задачу проверки достижимости (существуют ли пути между вершиной источником и остальными вершинами графа). Впервые алгоритм поиска в ширину описан в работах Мура и Ли [10].

Алгоритм основан на обходе вершин графа "по слоям". На каждом шаге есть множество "передовых" вершин, для смежных к которым производится проверка, относятся ли они к еще не посещенным. Все еще не посещенные вершины добавляются в новое множество "передовых" вершин, обрабатываемых на следующем шаге. Изначально в множество "передовых" вершин входит только вершина-источник, от которой и начинается обход [11].

В последовательном случае алгоритм имеет алгоритмическую сложность O(|V|+|E|), где |V| - число вершин в графе, |E| - число ребер в графе.

Пусть задан граф G=(V,E) без весов, и с выделенной вершиной-источником u. Путем P(u,v) между вершинами u и v называется множество ребер (u,v1),(v1,v2),...(vn−1,v). Длиной пути d(u,v) обозначим число ребер в

данном пути между вершинами u и v. Поиск в ширину находит кратчайшие пути d(u,v) от вершины u до всех остальных вершин графа описанным далее образом.

В начале работы алгоритма расстояние до вершины-источника d(u)=0, до остальных вершин d(v)=∞, ∀v≠u. Также в начале работы алгоритма инициализируется множество F={u}.

Далее на каждом шаге алгоритма строится множество вершин P=w, таких, что для ∀v∈F∃(v,w)∈E|d(w)=∞, при этом обновляются расстояния d(w)=d(v)+1 для ∀w∈P. Затем производится переход на следующий шаг до тех пор, пока P≠∅; при этом в начале каждого шага множество F заменяется на P.

Вычислительным ядром алгоритма является обход вершин, смежных с выбранной вершиной v, с последующим добавлением еще не посещенных вершин в множество P. Данная операция выполняется на каждом шаге для каждой вершины v∈F.

Макроструктура алгоритма. Алгоритм последовательно уточняет значения функции d(v).

Структуру можно описать следующим образом:

– Инициализация: всем вершинам присваивается предполагаемое расстояние d(v)=∞, кроме вершины-источника, для которой d(u)=0.

– Помещение вершины источника v в множество "передовых" вершин F.

– Обход вершин множества F.

Инициализация множества P=∅.

Для каждой вершины v∈F обход всех вершин w|∃(v,w) (смежных с ней), с помещением в множество P таких вершин w|d(w)=∞.

Замена множества F на P и переход на шаг 3 в случае, если множество F≠∅.

Анализ времени работы. Оценим время работы для входного графа G= (V, E), где множество ребер E представлено списком смежности. В очередь добавляются только непосещенные вершины, поэтому каждая вершина

посещается не более одного раза. Операции внесения в очередь и удаления из нее требуют O (1) времени, так что общее время работы с очередью составляет O(|V|) операций. Для каждой вершины v рассматривается не более deg(v) ребер, инцидентных ей. Так как ∑v∈Vdeg(v)=2|E|, то время, используемое на работу с ребрами, составляет O(|E|). Поэтому общее время работы алгоритма поиска в ширину — O(|V|+|E|)

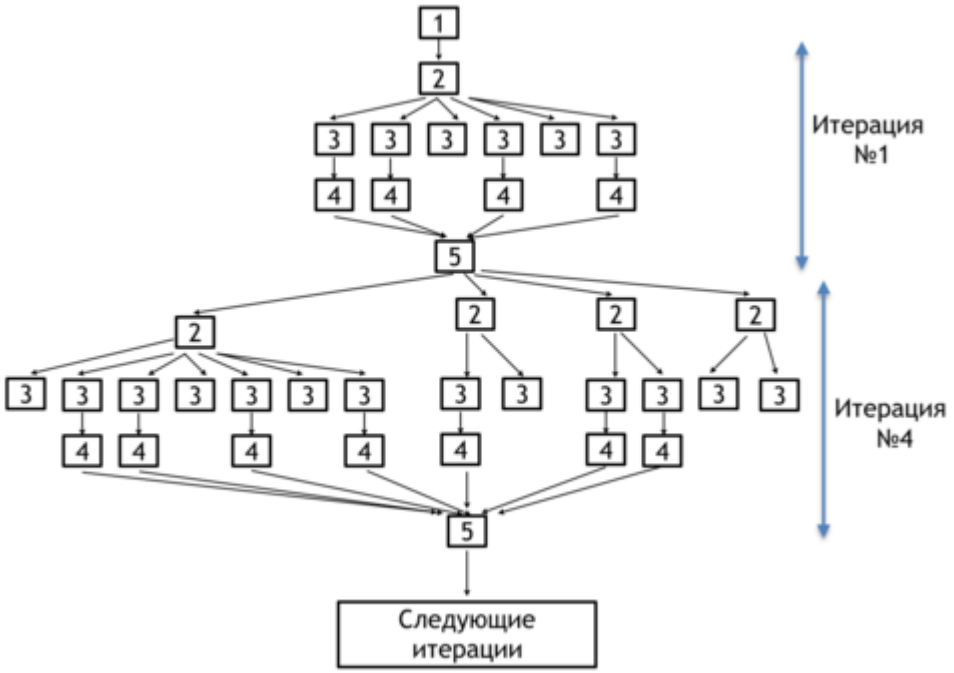

Рисунок 11 – Информационный граф классического алгоритма поиска в ширину

На рисунке 11 используются следующие обозначения:

[1] - добавление вершины-источника u к множеству F.

[2] - извлечение добавленной вершины v из множества F.

[3] - проверка расстояний до вершин, смежных с вершиной v.

[4] - добавление еще не посещенных вершин в множество P.

[5] - замена множества F на P и проверка его пустоты. В случае непустого множества - переход на следующую итерацию, иначе завершение работы алгоритма.

Данный алгоритм имеет один важный недостаток при реализации: операция [4] требует бесконфликтной возможности добавления элементов в множество P, что, на практике, всегда будет сводиться к сериализации обращений к структуре данных, моделирующей данное множество.

В результате часто используется модификация алгоритма (далее алгоритм-2), использующая набор независимых структур данных для каждого из параллельных процессов. Информационный граф данного подхода приведен на рисунке1 2.

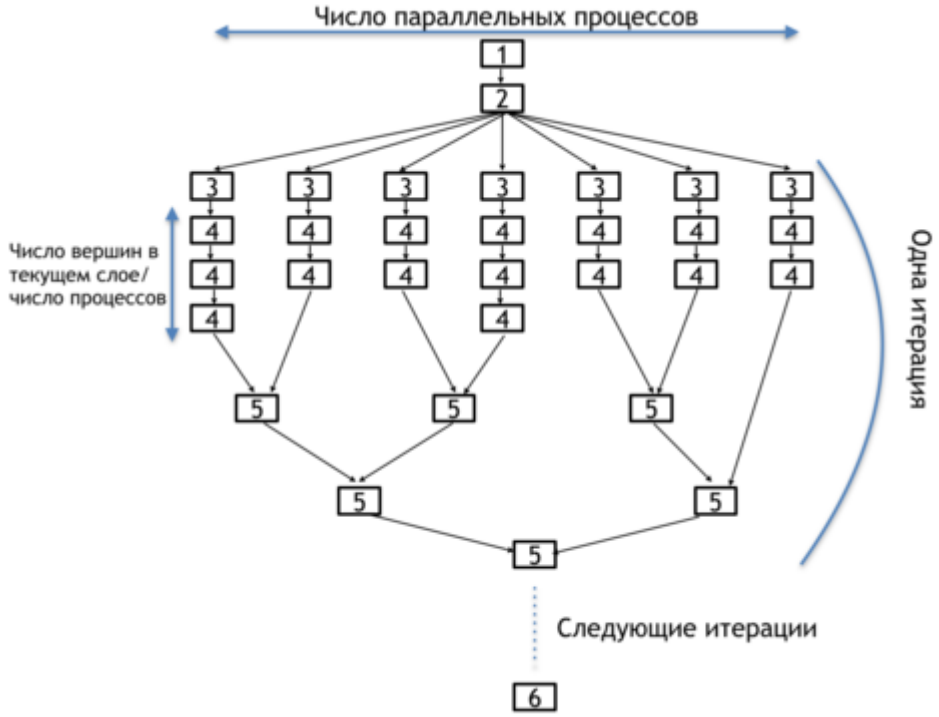

Рисунок 12 – Информационный граф алгоритма BFS (независимые структуры данных)

Обозначения для рисунка 12:

[1] - добавление вершины-источника в множество F.

[2] - разделение данных множества F между процессами

[3] - помещение в множества Fi соответствующих данных из F каждым процессом с номером i.

[4] - извлечение очередной вершины из множеств Fi, обход её соседей и добавление их в множество Pi в случае, если они еще не посещены

[5] - попарное слияние множеств Pi для различных процессов, итоговое преобразование их в множество F.

[6] - проверка условия выхода из цикла

Кроме того, в случае, если реализация структур данных, моделирующих множества F и P, невозможна, может использоваться квадратичный по сложности алгоритм, схожий с алгоритм Беллмана-Форда. Основная идея заключается в том, что на каждом шаге производится обход всех ребер графа с обновлением текущего массива дистанций. Информационный граф данной модификации алгоритма приведен на рисунке 13.

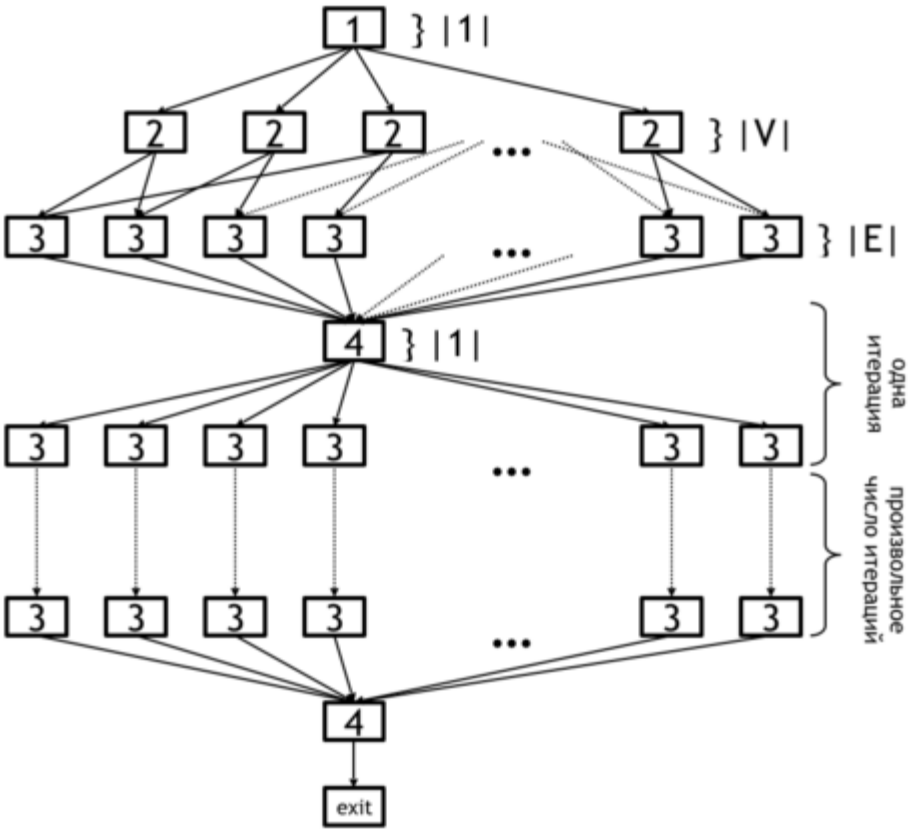

Рисунок 13 – Информационный граф алгоритма BFS (квадратичный подход к распараллеливанию).

Обозначения для рисунка 13:

[1] - инициализация расстояний до вершины-источника

[2] - инициализация расстояний до остальных вершин графа

[3] - загрузка информации об очередном ребре и обновление дистанций до соответствующих вершин.

[4] - проверка условия выхода из цикла

В ходе работы классический вариант алгоритма обходит граф по слоям. В каждый слой добавляются еще не посещенные вершины, достижимые из вершин предыдущего слоя. Обход вершин каждого слоя, как и их соседей, может производиться параллельно. Точно оценить число вершин в каждом слое невозможно в силу того, что их количество зависит от структуры связанности входного графа. Аналогично невозможно оценить число шагов алгоритма, за которое будут найдены все кратчайшие пути [13].

Произведем оценку ширины ярусно-параллельной формы алгоритма через максимальное число вершин p в слое среди всех шагов алгоритма. Тогда число параллельных операций на данном слое будет равно сумме числа смежных вершин для каждой вершины слоя: $\sum_{n=1}^{p} degree(v_i)$, при этом для каждого слоя данное значение будет различным. Высота ярусно-параллельной формы будет равна числу шагов в алгоритме и может быть оценена только сверху (не более $|V|$).

При квадратичном подходе к параллельной реализации алгоритма на каждом шаге выполняется $O(|E|)$ операций, которые могут быть выполнены параллельно; таким образом, ширина ЯПФ данной модификации алгоритма равна $O(|E|)$. Число шагов алгоритма, как и в классическом случае, зависит от структуры графа и может быть оценено сверху как $O(|V|)$.

Для того, чтобы было возможным сравнивать два лога между собой, необходимо получить их представления моделей процессов, представленные в виде системы переходов для каждого из исходных логов. После того, как графическое представление двух логов получено можно приступить к их сравнению. В рамках данной работы поиск различий будет выполняться с помощью следующего алгоритма, за основу которого взят алгоритм поиска в ширину на графе [14].

Входные данные: граф G(V,E), |V| вершин vi и |E| рёбер ej=(v(1)j,v(2)j), вершина-источник u.

Объём входных данных: O(|V|+|E|).

Выходные данные (возможные варианты):

– для каждой вершины v исходного графа расстояние d(v), определенное как число ребер, лежащих на кратчайшем пути от вершины u к v.

– для каждой вершины v исходного графа значение достижимости (достижима или нет) от вершины-источника u.

Объём выходных данных: O(|V|).

Поиск в ширину является методом обхода графа и является одним из неинформированных алгоритмов поиска (алгоритм поиска методом грубой силы), в котором используется только та информация, которая представлена в определении задачи и не используется дополнительная информация о состояниях. Данный метод представляет стратегию поиска решений в пространстве состояний, в которой вначале развертывается корневой узел, затем все приемники корневого узла, после этого развертываются приемники этих приемников и так далее до тех пор, пока все узлы не пройдены. На рисунке 1 изображен граф, вершины которого отмечены в порядке их прохождения с помощью метода поиска в ширину.

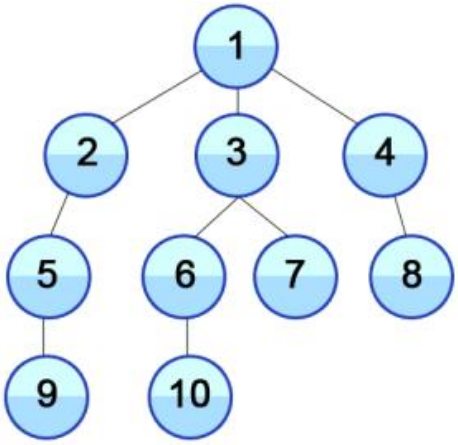

Рисунок 14 – Поиск в ширину

Пошаговое описание алгоритма.

1. Рассматриваем первую систему переходов (одну из систем переходов, которые сравниваем).

2. Берем корневой узел (стартовую вершину системы переходов) первой системы переходов и корневой узел второй системы переходов

3. По-очереди берем все переходы, которые выходят из выбранных узлов двух систем переходов (переходы представляю собой событий лога) и сравниваем переходы первой системы переходов с переходами второй системы переходов

4. Если переход равны, то запоминаем вершину, в которую входят эти переходы

5. По итогам прохождения шагов 2-4 может получится от 0 (нет совпадений) до N (все переходы совпали), где N – минимальное количество переходов из вершины одной из системы переходов, взятой на шаге 3

6. Если нет совпадений, то переходим на шаг 9

7. Если есть совпадения между переходами, то рассматриваем далее все переходы из вершин, которые были сохранены на шаге 4

8. Повторяем шаги 3-5 для полученных вершин до тех пор, пока все вершины не будут пройдены

9. Если на шаге 6 выяснилось, что при сравнивании переходов не было обнаружено совпадающих, то во второй системе переходов выбираем вершину, которая следует за текущей вершиной (то есть переходим вниз по второму графу на одну вершину).

10. Сравниваем переходы из вершин первой системы переходов с переходами, которые выходят из полученной на 9 шаге вершины

11. Если переходы совпадают, то переход первой системы переходов и предыдущий переход второй системы переходов отмечаются специальной меткой. Если переходы не совпадают, то в первой системе переходов выбираем вершину, которая следует за текущей вершиной (т. е. переходим вниз по первому графу на одну вершину) и переходим на шаг

12. Если переходы совпадают, то выполняем шаг 12

12. Сохраняем вершины, в которые входят переходы и переходим снова на шаг 3

13. Выполняем шаги 10-11

14. В случае, если на шаге 13 найдены совпадающие переходы, то переход второй системы и предыдущий переход первой системы отмечаются специальной меткой. Переходим на шаг 12. Если не найдены совпадающие переходы, то переходим на шаг 15

15. Выбираем вершины, которые следуют за текущими вершинами

16. Сравниваем переходы. В случае равенства, предыдущие переходы отмечаются специальной меткой. Переходим на шаг 3 В случае, если нет совпадающих переходов, то берем следующие вершины (переходим вниз по обоим системам перехода) и переходим на шаг 3

17. Все шаги выполняются до тех пор, пока все вершины будут не пройдены. Если у второй системы переходов остаются не рассмотренные вершины, в то время как у первой системы переходов все вершины являются пройденными, то оставшиеся переходы второй системы переходов отмечаются специальной меткой.

2.2 Источники данных

Волна компьютеризации, которая затронула большинство предприятий в конце XX–начале XXI в., принципиально изменила внутренние структуры их бизнеспроцессов. Это выразилось во все большей интеграции процессов с автоматизированными системами при одновременном снижении зависимости достоверности информации от конкретных исполнителей.

Использование систем ERP, WFM или CRM позволяет существенно усилить контроль над протеканием бизнес-процессов, а также в автоматическом режиме накапливать объективную информацию о событиях, происходящих в рамках этих процессов [15]. В результате работы информационных систем накапливается значительный объем данных о протекании процессов на предприятии, информация обычно содержится в так

называемом журнале событий (в англоязычной литературе применяется термин log, мы будем также использовать термин лог).

В зависимости от полноты охвата информационной системой бизнес-процессов, эта информация может как давать полную картину происходящего на предприятии, так и отражать состояние отдельных процессов. Обычно форма информации структурирована, и ее можно свести к единому формату представления. Большинство логов, формируемых информационными системами и описывающих протекающие процессы, содержат следующие данные: название события, идентификатор экземпляра процесса, к которому оно относится, тип события (например, «начало задачи», «окончание задачи» и др.), сотрудник-инициатор события, время начала события и др. [16]

Различные информационные системы характеризуются разными форматами сохраняемых данных об истории процессов, и для обобщения схемы работы с ними необходимо использовать единый формат, к которому было бы возможно свести большинство логов. Существует несколько универсальных форматов описания логов, такие как MXML, Open XES. Верхний уровень (WorkFlowLog) содержит в себе информацию о журнале событий в целом. На этом уровне хранятся два основных вида информации: «данные» (Data) и «источники данных» (Source). Оба вида информации представляют собой словарь значений. Каждый журнал событий содержит информацию о наборе процессов (Process). Каждый процесс может исполняться неограниченное число раз, и каждый такой экземпляр процесса будет сохраняться в логе (Process Instance) [17].

Каждый экземпляр процесса создает записи о событиях (AuditTrailEntry), содержащие необходимую информацию о событиях: Название события, Тип события, Дата события и Ответственный за событие. Кроме того, каждый уровень структуры может содержать дополнительную неструктурированную информацию в словаре «данные» (Data), состоящем из записей ключ-значение.

Формат Open XES в большей степени ориентирован на расширяемость и универсальность. Основное отличие Open XES заключается в отсутствии стандартных атрибутов, в базовой версии. Любое добавление атрибутов осуществляется либо через «расширение» (Extension), либо через «классификатор» (Classifier), которые позволяют описать формат этих атрибутов. Это гибкая модель, делающая возможным описание истории протекания широкого класса процессов.

Недостатком Open XES является отсутствие наглядности, поскольку использование расширений является неудобным в простых случаях, излишне загромождая описание. Можно сделать вывод, что формат Open XES является более универсальным и подходящим для описания широкого класса исторических данных процессов, формат MXML напротив не является столь гибким, но позволяет коротко и наглядно структурировать данные процессов во многих практически значимых случаях. Наличие исторических (ретроспективных) данных о деятельности предприятия открывает широкие возможности для прогнозирования. Так, в организациях, внедривших автоматизированные информационные системы управления бизнес-процессами, накопившиеся данные позволяют предсказывать будущие состояния таких процессов. Основной метод прогнозирования в этой области – математическое моделирование, в рамках которого для отдельных частей или для бизнес-процесса в целом строится математическая модель, позволяющая формировать прогнозы [18].

Лог событий содержит данные, которые относятся к конкретному одиночному процессу, то есть для того, чтобы начать работать с данными, необходимо убедиться, что все события в логе относятся к одному процессу. Каждое событие в логе представлено как некоторое действие. В таблице (рис.15) представлены следующие события — «Регистрировать запрос», «Проверить билет», «Отклонить запрос», «Принять решение» и другие. События должны быть отсортированы для того, чтобы была возможность исследовать зависимости внутри модели процесса.

		Свойства				
Id	Id события	Время	Действие	Ресурс	Стоимость	...
1	35653323	30-12-2010:11.02	Регистрировать запрос	Петр	50	...
	35653324	31-12-2010:10.06	Проверить (тщательно)	Светлана	400	...
	35653325	05-01-2011:15.12	Проверить билет	Михаил	100	...
	35653326	06-01-2011:11.18	Принять решение	Светлана	200	...
	35653327	07-01-2011:14.24	Отклонить запрос	Петр	200	...
2	35654483	30-12-2010:11.32	Регистрировать запрос	Михаил	50	...
	35654485	30-12-2010:12.12	Проверить билет	Михаил	100	...
	35654487	30-12-2010:14.16	Проверить (базовые проверки)	Петр	400	...
	35654488	05-01-2011:11.22	Принять решение	Светлана	200	...
	35654489	08-01-2011:12.05	Выплатить компенсацию	Елена	200	...
3	35653521	30-12-2010:14.32	Регистрировать запрос	Петр	50	...
	35653522	30-12-2010:15.06	Проверить (базовые проверки)	Михаил	400	...
	35653524	30-12-2010:16.34	Проверить билет	Елена	100	...
	35653525	06-01-2011:09.18	Принять решение	Светлана	200	...
	35653526	06-01-2011:12.18	Инициировать запрос повторно	Светлана	200	...
	35653527	06-01-2011:13.06	Проверить (тщательно)	Шамиль	400	...
	35653530	08-01-2011:11.43	Проверить билет	Ппетр	100	...
	35653531	09-01-2011:09.55	Принять решение	Светлана	200	...
	35653533	15-01-2011:10.45	Выплатить компенсацию	Елена	200	...
4	35653641	06-01-2011:15.02	Регистрировать запрос	Петр	50	...
	35653643	07-01-2011:12.06	Проверить билет	Михаил	100	...
	35653644	08-01-2011:11.43	Проверить (тщательно)	Шамиль	400	...
	35653645	09-01-2011:12.02	Принять решение	Светлана	200	...
	35653647	12-01-2011:15.44	Отклонить запрос	Елена	200	...
...	...	...	...	...	...	...

Рисунок 15 – Фрагмент журнала события

Также на рисунке показана дополнительная информация для каждого события, такая как всех событий есть временные метки, ресурсы и стоимость. Временные метки содержат информацию о дате и времени события. Эта информация может оказаться полезной при анализе характеристики некоторых свойств (например, время ожидания между двумя действиями). Ресурсы представляют собой людей, которые выполняли то или иное действие. В контексте process mining эти свойства называются атрибутами.

Лог также должен соответствовать критериям качества. Некоторые из них:

– Лог должен быть отсортирован. Как правило, лог сортируется сначала по ID_события, затем по времени.

– Не должно быть пропусков указанных ключевых полей. Иначе потребуется или перевыгрузка, или удаление всего ID_события с пропусками.

– Если один ID_события содержит только одно событие, это служит «желтым» сигналом: необходимо перепроверить качество логирования.

Каждое событие имеет один или несколько атрибутов. Расширенный лог может содержать формально не ограниченный перечень дополнительных полей — атрибутов.

На рисунке 16 показано структурированное дерево лога событий. С помощью этого дерева, можно сделать следующие выводы о логе событий:

– Процесс состоит из различных случаев

– Каждый случай содержит в себе различные события

– Событии внутри каждого случая отсортированы

– У событий могут быть атрибуты. Примерами атрибутов могут быть действия, время, стоимость, ресурс.

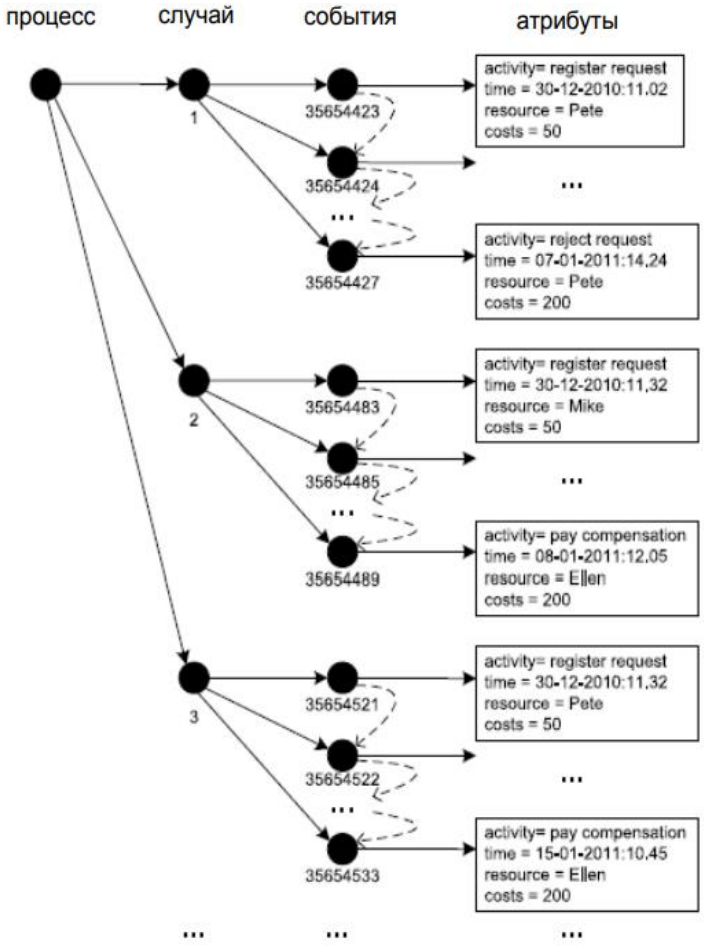

Рисунок 16 – Схема формирования записей о событии

Согласно описанию стандарта XES, объект log на самом верхнем уровне формата содержит всю информацию о событиях, которая относится к одному определенному процессу (например, процесс использования рентгеновского аппарата или обработка страховых запросов). Для описания данного объекта существует тег <log>. Лог может содержать произвольной количество объектов последовательностей событий, в том числе количество может быть нулевым. Каждая последовательность описывает выполнение одного конкретного случая процесса из лога события — например, обработка конкретного страхового случая, одно посещение веб-сайта конкретным пользователем, один осмотр с помощью рентгеновского аппарата. Имя тега для хранения этих данных – <trace>. Каждая последовательность в свою очередь содержит произвольное число объектов — событий, так же может не содержать ни одного.

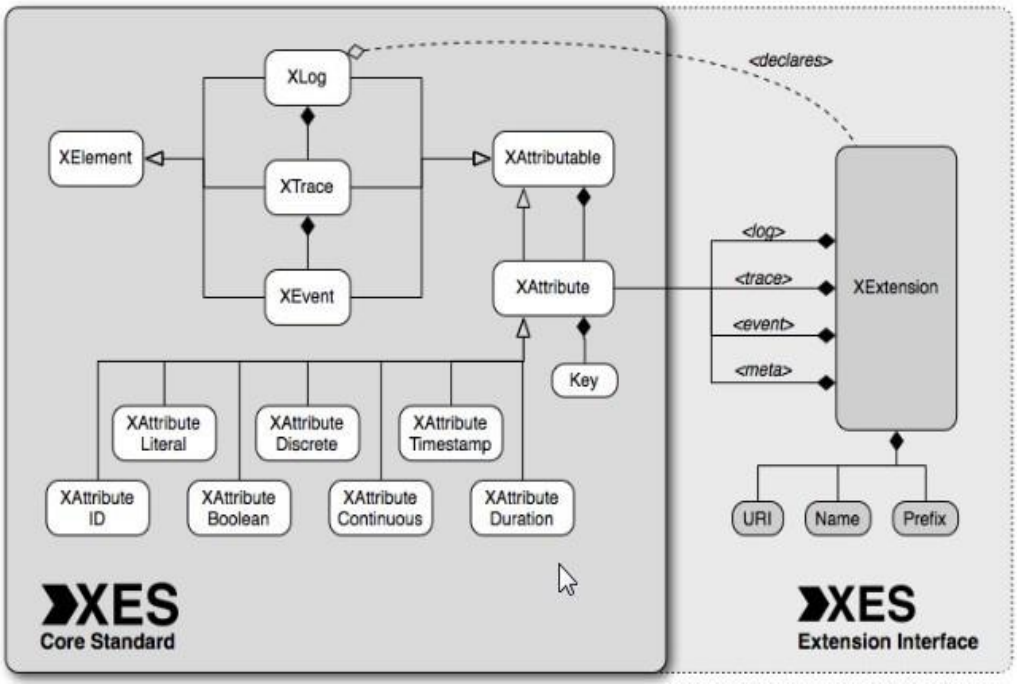

Рисунок 17 – Главные компоненты формата XES

События представляют собой неделимую частицу активности, которая произошла в течении исполнения рассматриваемого процесса. Примерами событий являются — запись персональной информации о клиенте в базу, импорт или экспорт изображения с определенного сайта, осуществление

снимка с помощью рентгеновского аппарата. Имя тега для данного объекта – <event>. Описанные объекты (лог, последовательность, событие) не содержат никакой информации, а только задают структуру документа. Вся смысловая информация в логе событий хранится в так называемых атрибутах лога. У каждого из атрибутов существует строка-ключ.

Каждый из логов, последовательностей, событий содержат произвольное число атрибутов, которые разделяются на шесть типов: строка, дата, целое число, число с плавающей точкой, логическое выражение, идентификатор. В качестве дополнительной информации существуют вложенные атрибуты, созданные для того, чтобы формат лога был более гибким. Вложенными называют такие атрибуты, у которых могут быть атрибуты-потомки.

2.3 Обзор возможностей профильного программного обеспечения

Данное ПО предназначено для нахождения различий в логах событий для того, чтобы сравнивать логи событий, необходимо четко представлять цель сравнения. Предполагается, что для поиска различий берутся логи, сравнение которых имеет смысл (например, это могут быть логи о выплате компенсационных выплат за разные промежутки времени, логи о процессах покупки товаров в интернет-магазине (или разных интернетмагазинах одной компании)). Целью нахождения различий может быть оптимизация процессов, анализ протекающих процессов в зависимости от времени (суток, года и прочее). Внедрение предполагает увеличение коэффициента качества сравнения, а также повышение, оптимизация и улучшения бизнес-процессов. Также, полученные результаты могут быть использованы при работе над созданием и разработкой различных алгоритмов и методов сравнения логов событий.

Данное ПО будет реализовано в качестве плагина для системы ProM. На вход работы плагина подается два лога формата XES после выбора файлов и во время процесса импорта, файлы формата XES конвертируются в файлы поддерживаемого системой ProM формата MXML. При помощи алгоритма

можно находить различия в логах событий до одного событий: удаленное (пропущенное) событие, добавленное (лишние) событие, измененное событие. Также есть возможность определять полностью идентичные последовательности событий.

Разработанный плагин к системе ProM позволит решить задачу визуализации различий в логах событий. Также разработанный алгоритм и реализация его в рамках плагина к инструменту ProM может быть использован, как и непосредственно для сравнения двух логов событий экспертами в области работы с бизнеспроцессами и другими заинтересованными лицами, так и в качестве основы для работы над алгоритмами сравнения логов событий между собой.

Задачи ПО заключаются в представлении логов событий в виде систем переходов с помощью предназначенного для построения системы переходов плагина. В системе перехода каждый из переходов соответствуют одному событию из рассматриваемого лога, поэтому внимание акцентируется именно на переходах, а не вершинах. После того, как системы переходов построены, можно начинать сравнивать их между собой, отмечая переходы соответствующей цветовой маркировкой в случае нахождения различий.

2.4 Характеристика выбранного программно-технического обеспечения

IDE — это интегрированная, единая среда разработки, которая используется разработчиками для создания различного программного обеспечения [19].

IDE является комплексом из нескольких инструментов: текстового редактора, компилятора или интерпретатора, средств автоматизации сборки и отладчика. Интегрированные среды разработки позволяют максимизировать производительность программиста, ускорить процесс разработки.

Помимо вышеперечисленных инструментов IDE зачастую содержит средства конструирования GUI, средства интеграции с системами контроля версий, средства для объектно-ориентированной разработки и т. д.

Интегрированная среда позволяет программисту абстрагироваться от выполнения вспомогательных задач и избежать потери времени при выполнении типичных действий.

Наиболее популярные IDE для программирования на языке Java:

– IntelliJ IDEA

– Eclipse IDE

– NetBeans

– JDeveloper

– DrJava

– Android Studio

NetBeans IDE — свободная интегрированная среда разработки приложений (IDE) на языках программирования Java, Python, PHP, JavaScript, C, C++, Ада и ряда других.

Проект NetBeans IDE поддерживается и спонсируется компанией Oracle, однако разработка NetBeans ведётся независимым сообществом разработчиков-энтузиастов (NetBeans Community) и компанией NetBeans Org.

Последние версии NetBeans IDE поддерживают рефакторинг, профилирование, выделение синтаксических конструкций цветом, автодополнение набираемых конструкций на лету и множество предопределённых шаблонов кода.

Для разработки программ в среде NetBeans и для успешной инсталляции и работы самой среды NetBeans должен быть предварительно установлен Sun JDK или J2EE SDK подходящей версии. Среда разработки NetBeans по умолчанию поддерживала разработку для платформ J2SE и J2EE. Начиная с версии 6.0 NetBeans поддерживает разработку для мобильных платформ J2ME, C++ (только g++) и PHP без установки дополнительных компонентов [20].

В сентябре 2016 года Oracle передала интегрированную среду разработки NetBeans в руки фонда Apache.

В версии NetBeans IDE 6.1 декларируется поддержка UML, SOA, языка программирования Ruby (включая поддержку Ruby on Rails), а также средства для создания приложений на J2ME для мобильных телефонов. В версии 6.5 добавлена поддержка языка PHP. Также для тестирования выложен модуль поддержки Python.

NetBeans IDE поддерживает плагины, позволяя разработчикам расширять возможности среды. Одним из самых популярных плагинов является мощный дизайнер отчётов iReport (основанный на библиотеке JasperReports).

На идеях, технологиях и в значительной части на исходном коде NetBeans IDE базируются предлагаемые фирмой Sun коммерческие интегрированные среды разработки для Java — Sun Java Studio Creator, Sun Java Studio Enterprise и Oracle Solaris Studio (для ведения разработки на C, C++ или Фортран). Сравнительно недавно Sun стала предлагать эти среды разработки бесплатно для зарегистрировавшихся в Sun Developer Network (SDN) разработчиков, сама же регистрация на сайте бесплатна и не требует никаких предварительных условий, кроме согласия с лицензией CDDL.

NetBeans IDE доступна в виде готовых дистрибутивов (прекомпилированных бинарных файлов) для платформ Microsoft Windows, Linux, FreeBSD, Mac OS X, OpenSolaris и Solaris (как для SPARC, так и для x86 — Intel и AMD). Для всех остальных платформ доступна возможность скомпилировать NetBeans самостоятельно из исходных текстов.

В релизе NetBeans IDE 6.7 была добавлена интеграция с Project Kenai, поддержка языка Groovy и веб-фреймворка Grails. В версии 6.8 — поддержка PHP-фреймворка Symfony, а в 6.9 — Zend Framework.

В версии 7 производитель отказался от поддержки языка Ruby и Ruby on Rails, объясняя этот отказ большим объёмом работ по поддержке Java 7 и сравнительно невысокой потребностью в этом языке и технологии среди пользователей NetBeans (компонент Ruby and Rails был взят на сопровождение сообществом и продолжает существовать). Таким образом, «из коробки»

последние версии NetBeans поддерживают только Java (включая Java FX, Java ME, Java EE), C/C++, Groovy, PHP, HTML, JavaScript, CSS. Также для версии 7 не имеется ни собственных, ни бесплатных сторонних компонентов поддержки UML-моделирования.

Плагин создается для программного обеспечения ProM Tools – набор инструментов для анализа процессов, который объединяет большинство существующих методов process mining в качестве подключаемых модулей. Программа не зависит от используемой платформы, так как реализована на языке Java и может быть загружена бесплатно. Кроме того, ПО имеет несколько компонентов, что обеспечивает максимальную гибкость [21].

ProM развивается более 15 лет на базе Технологического университета Эйндховена. Но разработка инструментария не ограничена пределами учреждения. На данный момент в проекте ProM принимают участие разработчики со всего мира: Австралии, Австрии, Китая, Германии, Италии и других стран.

В настоящее время доступно более 230 интегрируемых модулей, осуществляется интеграция и преобразование нескольких языков моделирования процессов, таких как Сети Петри (PNML, TPN), EPCs/EPKs (формат графов Aris, EPML), YAWL и многие другие.

По словам разработчиков, для использования ProM не требуется особых знаний в программировании. Это позволяет любому пользователю начать работать в сфере процессной аналитики.

Современные версии продукта призваны помочь избежать многих ограничений, существовавших в предыдущих вариантах разработок. В первую очередь это относится к интеграции между самим инструментом и графическим интерфейсом.

ProM 6 был разработан с нуля и использует полностью новую архитектуру. Изменения были вызваны новым пониманием дизайна программного обеспечения для анализа процессов. Кроме того, введение

XESame в этот инструментарий позволяет преобразовывать журналы в собственный формат ProM без программирования.

Инструмент представлен в виде загружаемого пакета с использованием лицензии GNU Public License (GPL), имеющего открытый исходный код. Это означает, что использовать этот компонент можно бесплатно и без ограничений, но любое программное обеспечение должно иметь лицензию GPL. Подключаемые модули ProM 6 выпускаются в виде отдельных приложений, которые обычно используют лицензию Lesser GNU Public License (L-GPL) также с открытым исходным кодом. Разработчики обращают особое внимание, что использование программного обеспечения третьими сторонами может потребовать плагин с лицензией, отличающейся от L-GPL.

ProM Lite содержит более 100 плагинов, хорошо протестирован, имеет открытый исходный код. Поэтому даже в процессе использования можно самостоятельно добавлять новые функции в ProM. С помощью этой программы можно преобразовывать данные из таблиц в формат журнала событий и приступать к анализу процессов.

На данный момент ProM включает в себя следующие группы программных модулей для:

– анализа процессов управления (Alpha algorithm, Genetic mining, Heuristics Miner, Multi-phase mining);

– анализа перспектив работы компании (SocialNetwork miner, Staff Assignment miner);

– анализа данных (Decision miner);

– выявления сложных бизнес-процессов (Fuzzy Miner);

– подробной визуализации данных (Cloud Chamber Miner) и многие другие.

Кроме того, существуют аналитические модули, позволяющие:

– анализировать модели процессов (Woflan analysis), линейные формулы (LTL) по логам;

– проверять соответствие между заданной моделью процесса и журналом, а также анализировать производительность (Basic statistical analysis, Performance Analysis with a given process model).

На сегодняшний день ProM применяется в страховании, финансах, сфере здравоохранения. Его активно используют транснациональные корпорации, производители высокотехнологичных систем и их клиенты – медиакомпании и госучреждения.

В качестве примера использается: по словам Вил ван дер Аалста, в компании Rijkswaterstaat (Департамент общественных работ Нидерландов) при помощи ProM провели анализ процесса выставления счетов. Результаты дали понять, что плохая производительность этого БП связана с частой работой сотрудников удаленно. Кроме того, исследование показало, что для более глубокого понимания процесса целесообразно комбинировать несколько методов анализа.

В данном случае, модель процесса помогла найти ошибки, а именно – наличие временных петель. Для выявления этого факта потребовались как модель работы компании для определения ключевых участников БП, так и кейс-анализ для оценки уровня воздействия «узких мест» на общую производительность процесса.

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПРЕДПОЛАГАЕМОГО АЛГОРИТМА РЕШЕНИЯ ЗАДАЧИ

3.1 Основные этапы практической разработки программного продукта

Для того, чтобы реализовать данное решение создан плагин к системе ProM.

ProM - это расширяемый фреймворк, который поддерживает широкий спектр методов интеллектуального анализа процессов в виде подключаемых модулей. Он не зависит от платформы, поскольку реализован на Java, и может быть загружен бесплатно. Мы приветствуем и поддерживаем практическое применение ProM и приглашаем исследователей и разработчиков внести свой вклад в виде новых плагинов.

ProM развивается более 15 лет на базе Технологического университета Эйндховена. Но разработка инструментария не ограничена пределами учреждения. На данный момент в проекте ProM принимают участие разработчики со всего мира: Австралии, Австрии, Китая, Германии, Италии и других стран.

В настоящее время доступно более 230 интегрируемых модулей, осуществляется интеграция и преобразование нескольких языков моделирования процессов, таких как Сети Петри (PNML, TPN), EPCs/EPKs (формат графов Aris, EPML), YAWL и многие другие, также существует более 500 плагинов для различных целей, которые можно использовать и изменять в рамках решаемой задачи.

По словам разработчиков, для использования ProM не требуется особых знаний в программировании. Это позволяет любому пользователю начать работать в сфере процессной аналитики.

Современные версии продукта призваны помочь избежать многих ограничений, существовавших в предыдущих вариантах разработок. В первую

очередь это относится к интеграции между самим инструментом и графическим интерфейсом:

ProM был разработан с нуля и использует полностью новую архитектуру. Изменения были вызваны новым пониманием дизайна программного обеспечения для анализа процессов. Кроме того, введение XESame в этот инструментарий позволяет преобразовывать журналы в собственный формат ProM без программирования.

Инструмент представлен в виде загружаемого пакета с использованием лицензии GNU Public License (GPL), имеющего открытый исходный код. Это означает, что использовать этот компонент можно бесплатно и без ограничений, но любое программное обеспечение должно иметь лицензию GPL. Подключаемые модули ProM 6 выпускаются в виде отдельных приложений, которые обычно используют лицензию Lesser GNU Public License (L-GPL) также с открытым исходным кодом. Разработчики обращают особое внимание, что использование программного обеспечения третьими сторонами может потребовать плагин с лицензией, отличающейся от L-GPL.

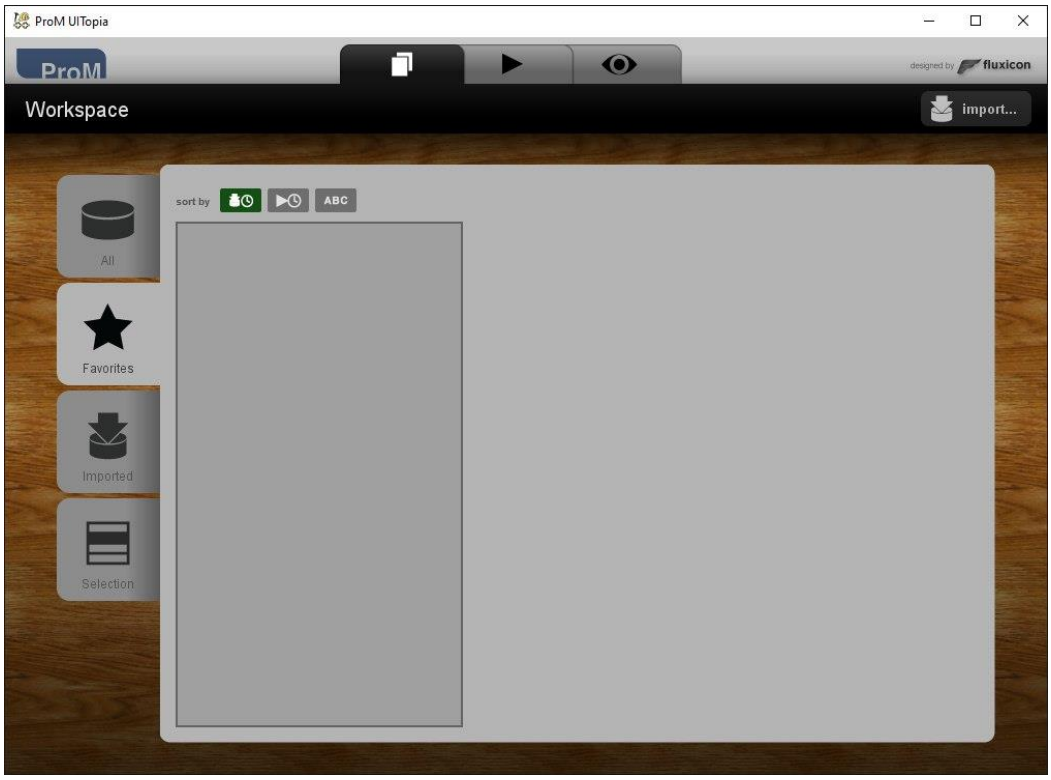

Рисунок 18 — Рабочая область системы ProM

3.1.1 Построение системы переходов

Для того, чтобы лог событий можно было представить в виде системы переходов, будет использован плагин к системе ProM – «Transition System Miner».

Данный плагин позволяет извлекать логи событий для системы переходов. Результатами работы данного плагина является построенная система переходов по логу событий, который передается данному плагину в качестве входных данных. Переходы построенной системы соответствуют событиям в логе, в то время как состояния соответствуют некоторой ситуации, которая происходит между двумя событиями.

В плагине «Transition System Miner» реализована функциональность для различия состояний между собой с помощью идентификатора каждого состояния. В исходном файле лога событий существуют записи о классификаторах лога, которые могут быть использованы для идентификации состояний в построенной системе перехода. В рамках данной работы больший интерес представляю переходы, которые соответствуют событиям лога, поэтому классификаторы для идентификации состояний использованы не будут.

На полученной с помощью плагина системе переходов толщина переходов зависит от частоты встречающихся в логе событий, соответствующих данному переходу.

Рассмотрим для примера лог со следующими последовательностями событий: ABGE, BDFG, ABDG, ABG, ABGE, ABGE, ABGE.

Система переходов для данного лога изображена на рисунке 19, на котором наглядно показано, что толщина переходов A, B отличается от толщины переходов G, E, так как события A, B встречаются в логе чаще, чем событий G, E. В свою очередь, события F, D случаются еще реже, поэтому толщина соответствующих данным событиям переходов меньше, чем остальных.

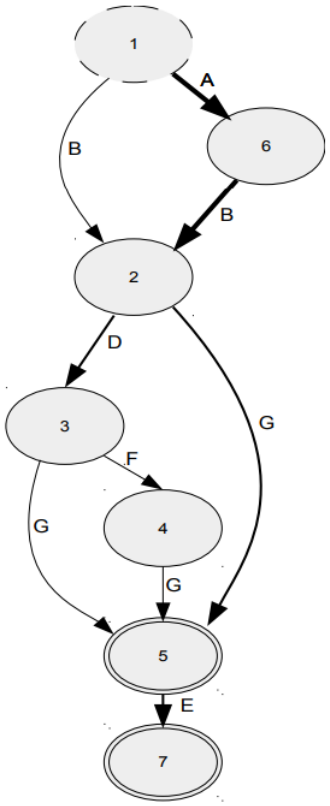

Рисунок 19 — Толщина переходов в построенной по журналу событий системе переходов

Для того, чтобы было возможным работать с плагином, необходимо, чтобы на вход поступал лог событий формата XES, который стандартными средствами системы ProM конвертируется в формат MXML. После того, как исходные данные загружены в систему, можно использовать их для построения системы переходов.

В качестве еще одного примера рассмотрим лог событий, который состоит из 100 экземпляров процесса, каждый из который содержит от 4 до 16 событий. После того, как исходный лог был загружен в систему, с помощью плагина строится система переходов. Сто последовательностей событий, которые хранятся в одном логе, можно представить в достаточно компактном виде с помощью системы переходов.

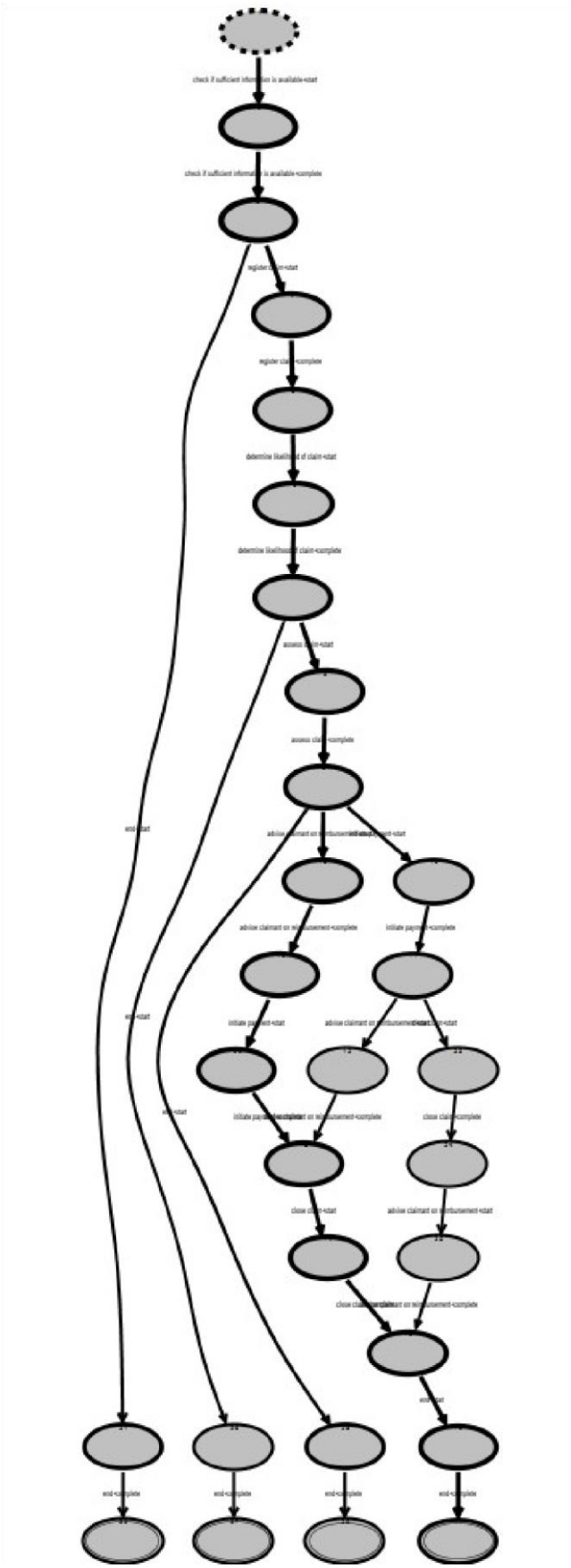

Рисунок 20 — Система переходов на основе большого лога событий

Для реализации задачи в рамках данной работы плагин «Transition System Miner» был изменен. Так как данный плагин работает только с одним логом, то необходимо было изменить некоторые методы плагина для того, чтобы было возможным загружать несколько логов, по которым были бы построены отдельные системы переходов. Для этого, метод для построения системы переходов TSMinerInput был изменен: входное значение исходного лога, передаваемое в данный метод, было заменено на массив значений. В данный массив можно передать несколько логов (больше двух), по которым будут построены системы переходов. В рамках данной работы нас интересует построение систем переходов по двум логам, однако данный метод является модифицированным и готовым для работы с большими данными, чем два лога события.

3.1.2 Описание работы программы

Для реализации предлагаемого решения визуализации различий в логах событий, написан плагин к системе ProM.

Для работы с плагином необходимо открыть систему ProM. Для запуска плагина необходимо загрузить два лога формата XES, которые необходимо сравнить, с помощью стандартного для системы ProM способа загрузки исходных файлов. Загрузка файлов осуществляется по нажатию кнопки Import на странице с рабочей областью системы ProM.

Плаги́н (англ. plug-in, от plug in «подключать») — независимо компилируемый программный модуль, динамически подключаемый к основной программе и предназначенный для расширения и/или использования её возможностей. Плагины обычно выполняются в виде библиотек общего пользования.

Для возможности подключения плагинов разработчик основного приложения должен предусмотреть в нем некоторый программный интерфейс, а также хотя бы минимальные возможности по управлению набором плагинов. В этом случае возможности программы могут быть

расширены сторонними разработчиками. В некоторых приложениях плагины могут настраиваться пользователем дополнительно.

Основное приложение предоставляет сервисы, которые плагин может использовать. К ним относится предоставляемая плагину возможность зарегистрировать себя в основном приложении, а также протокол обмена данными с другими плагинами.

Плагины являются зависимыми от сервисов, предоставляемых основным приложением, и зачастую отдельно не используются. В противоположность им, основное приложение независимо оперирует плагинами, предоставляя конечным пользователям возможность динамически добавлять и обновлять плагины без необходимости внесения изменений в основное приложение.

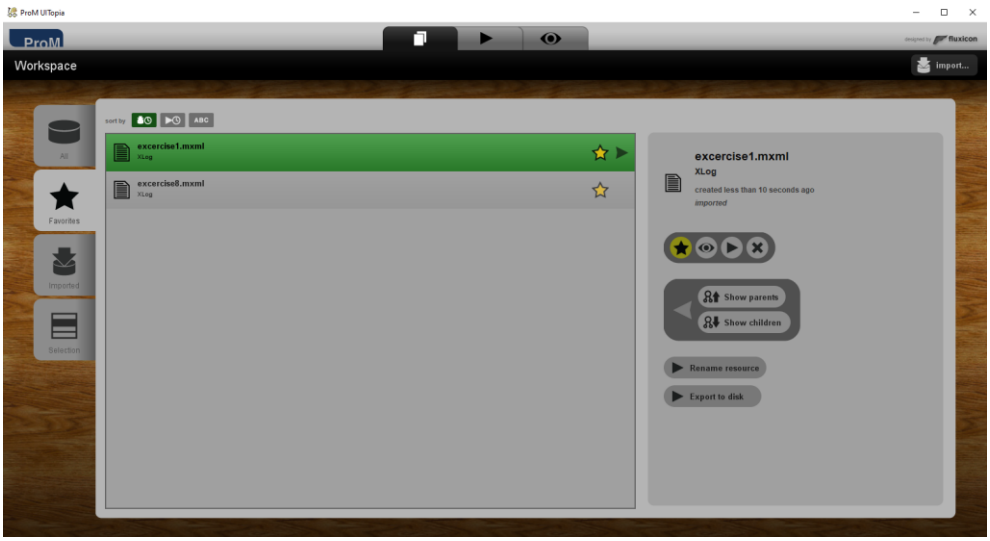

Рисунок 21 — Импорт исходных файлов в систему ProM

После выбора файлов и во время процесса импорта, файлы формата XES конвертируются в файлы поддерживаемого системой ProM формата MXML. По окончании импорта исходных файлов в систему (в случае, если импорт файлов прошел успешно), добавляем загруженные файлы в качестве исходных данных, необходимых для запуска плагина. Затем выбираем плагин из списка доступных плагинов (доступные плагины выделены отличающимся от других плагинов цветом) и нажимает кнопку Start.

Рисунок 22 — Выбор плагина для запуска

Для того, чтобы было возможно запустить этот же плагин с другими данными (для повторного запуска или в случае возникновения ошибки при загрузке или при выполнении плагина), необходимо нажать на кнопку Reset на вкладке Actions, изображенную на рисунке 22 и добавить другие файлы в качестве исходных данных. Если же файлы еще не загружены, то необходимо загрузить из с помощью кнопки Import на странице рабочей области системы ProM.

3.2 Результаты фактического тестирования программного продукта

По завершении реализации работа плагина была протестирована с различными исходными данными.

3.2.1 Проверка визуализации эквивалентных событий

Для проверки визуализации эквивалентности событий двух логов, загрузим логи со следующими событиями. $Лог_1$ содержит две последовательности: ABD, ACF. $Лог_2$ содержит одну последовательность событий — ABD.

$Лог_1$: ABD, ACF.

$Лог_2$: ABD.

Последовательность ABD для двух логов является одинаковой, следовательно все события данной последовательности отмечаются серым цветом (рис. 23). Все состояния системы переходов, в которые входят и из которой выходят только отмеченные серым цветом переходы, отмечаются темно-серым цветом.

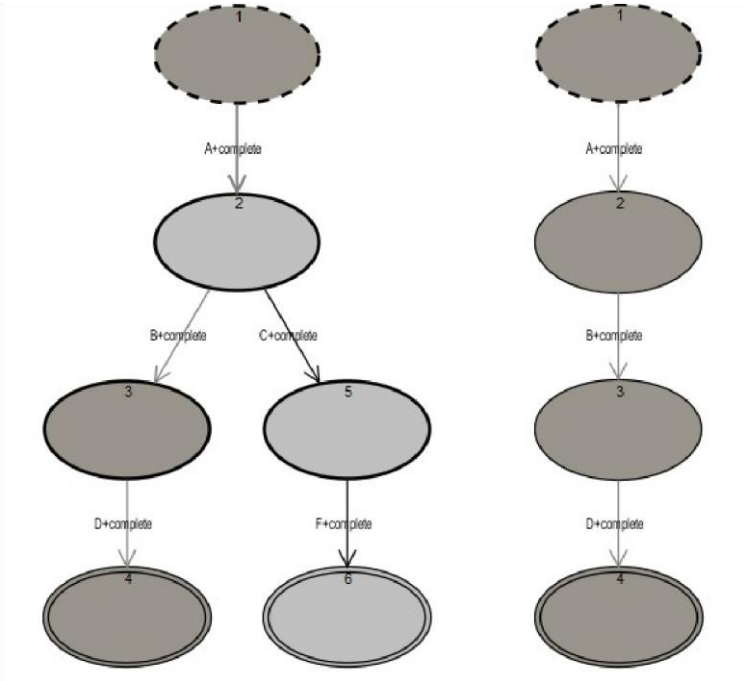

Рисунок 23 — Визуализация эквивалентных событий

3.2.2 Проверка визуализации отличий в логах события на одно событие в одном из логов

Для проверки визуализации различий в логах событий на одно событие, в качестве примера рассмотрим следующие два лога. $Лог_1$ содержит последовательности событий: AED, ACBD. $Лог_2$ содержит в свою очередь последовательности событий ACD, AED.

$Лог_1$: AED, ACBD.

$Лог_2$: ACD, AED.

Для двух данных логов события A, E, D последовательности AED являются эквивалентными и на построенной системе переходов (рис.24) отмечаются серым цветом. Событие B последовательности ACBD первого лога является удаленным (лишним) относительно последовательности ACD

второго лога и отмечается красным цветом и соответствующей пометкой на системе переходов. Событие D на второй системе переходов отмечается зеленым цветом и соответствующей пометкой с названием пропущенного события (B).

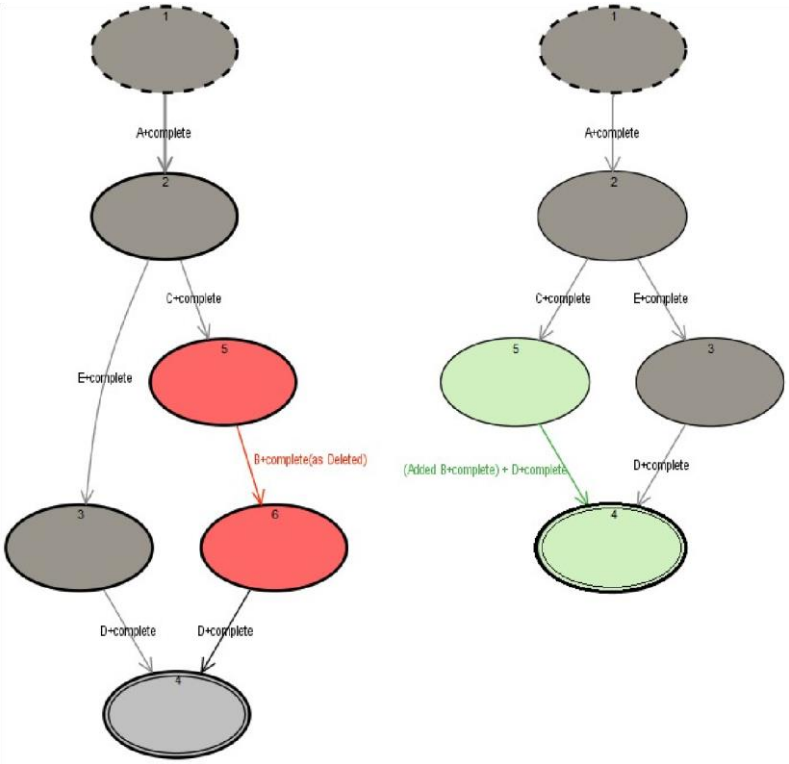

Рисунок 24 — Визуализация различий в логах событий на одно событие в одном из логов

3.2.3 Проверка визуализации отличий в обоих логах событий на одно событие в каждом

В качестве примера возьмем два исходных лога со следующими данными: последовательности событий ABD, ABCD в первом логе и последовательность AECD во втором логе, где каждая буква является отдельным событием.

$Лог_1$: ABD, ABCD.

$Лог_2$: AECD.

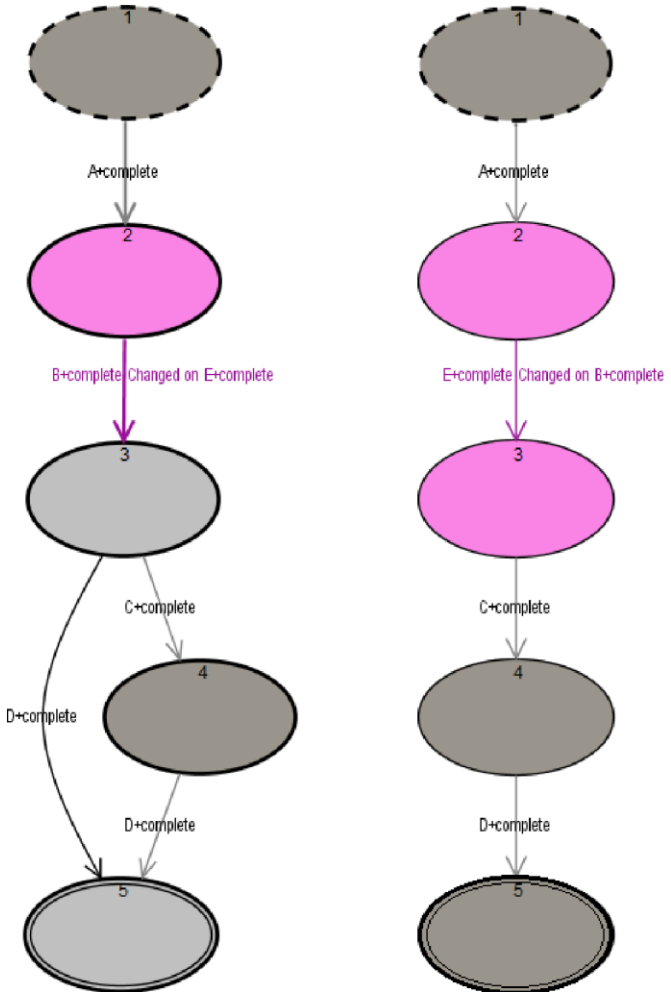

Рисунок 25 — Визуализация различий в логах событий на одно событие в каждом из логов

Последовательности ABCD и AECD отличаются ровно на одно событие — события A, C, D являются одинаковыми для двух последовательностей, события B и E присутствуют только в одном из логов и отсутствуют в другом.

На рисунке 25 изображены системы переходов, построенные по двух логам событий. На данных системах событий, которые присутствуют в одном логе, и отсутствуют в другом, отмечены специальной цветовой отметкой. Переходы, которые соответствуют этим событиям отмечены фиолетовым цветом и особой пометкой, содержащей название события, которое присутствует в другом логе вместо события в текущем логе. Если заменить текущее событие на присутствующее в другом логе, то последовательности будут эквиваленты друг другу.

3.2.4 Проверка визуализации отличий больше, чем на одно событие

Так как с помощью реализованного в рамках данной работы плагина возможно визуализировать различия в логах событий только до одного элемента (события), то рассмотрим контрпример, в котором присутствуют различия в логах событиях больше, чем на одно событие. Для примера возьмем два лога со следующими последовательностями событий: ABCEFD в первом логе и ABCD во втором логе, где каждая буква является отдельным событием.

$Лог_1$: ABCEFD.

$Лог_2$: ABCD.

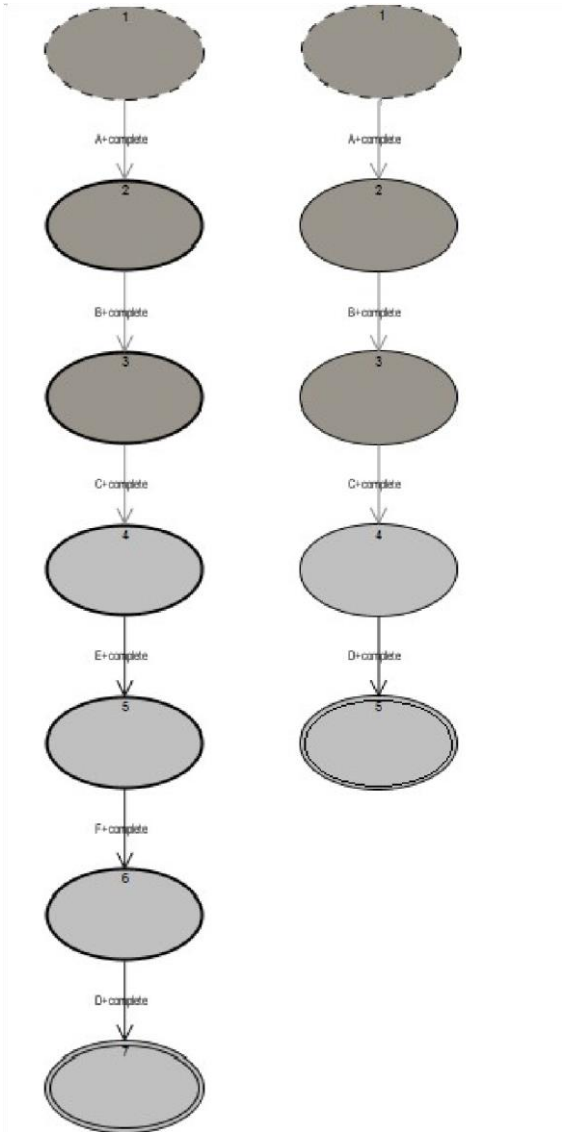

Рисунок 26 — Различие логов событий больше, чем на одно событие

Событий A, B, C для обоих логов событий являются одинаковыми, поэтому на системе переходов на рисунке 26, построенной для исходных логов, данные события отмечены соответствующим цветом — серым. Следующее одинаковое событие для обоих логов — событие D. Но для первого лога до события D существуют еще два события — E, F. Если бы в данном случае существовало только одно событие — E или F, то оно было бы отмечено как добавленное. Но в данном случае эти переходы на системах перехода никак не отмечаются.

3.2.5 Проверка визуализации отличий на больших логах событий

Пример 1. Рассмотрим в качестве примера различные большие логи событий, которые содержат по несколько сотен записей о событиях.

В первую очередь рассмотрим два лога событий, которые описывают процесс простого займа. Эти логи являются искусственно созданными в качестве обучающих для работы с системой ProM в Технологическом университете Эйндховена. В логах описаны различные варианты процесса займа. В первом логе событий — 630 событий и 105 трасс, во втором логе 420 событий и 70 трасс. Эти логи несколько отличаются друг от друга, в частности в первом логе пропущены некоторые события, которые присутствуют во втором логе. Также в первом логе существуют события, порядок которых не совпадает с порядком событий во втором логе.

В результате, логи событий представляются в виде небольших систем переходов (рис. 27). Первый лог, представленный на первой системе переходов рисунка 27, состоит из множества повторяющихся последовательностей событий: $AB_1B_2CD_2E$, $AB_1B_2CD_2F$, где каждое событие представлено отдельной буквой. Из системы переходов видно, что последовательность $AB_1B_2CD_2F$ встречается в логе событий несколько чаще последовательности $AB_1B_2CD_2E$, так как переход системы переходов для события F выделен более жирной стрелкой, что говорит о частоте появления события в логе.

Второй лог событий, представленный на второй системе переходов (справа) на рисунке 27, состоит из множества повторяющихся последовательностей событий, таких как AB_1DB_2CF, $AB_1D_2B_2CE$ и других, где каждая буква представляет отдельное событие лога.

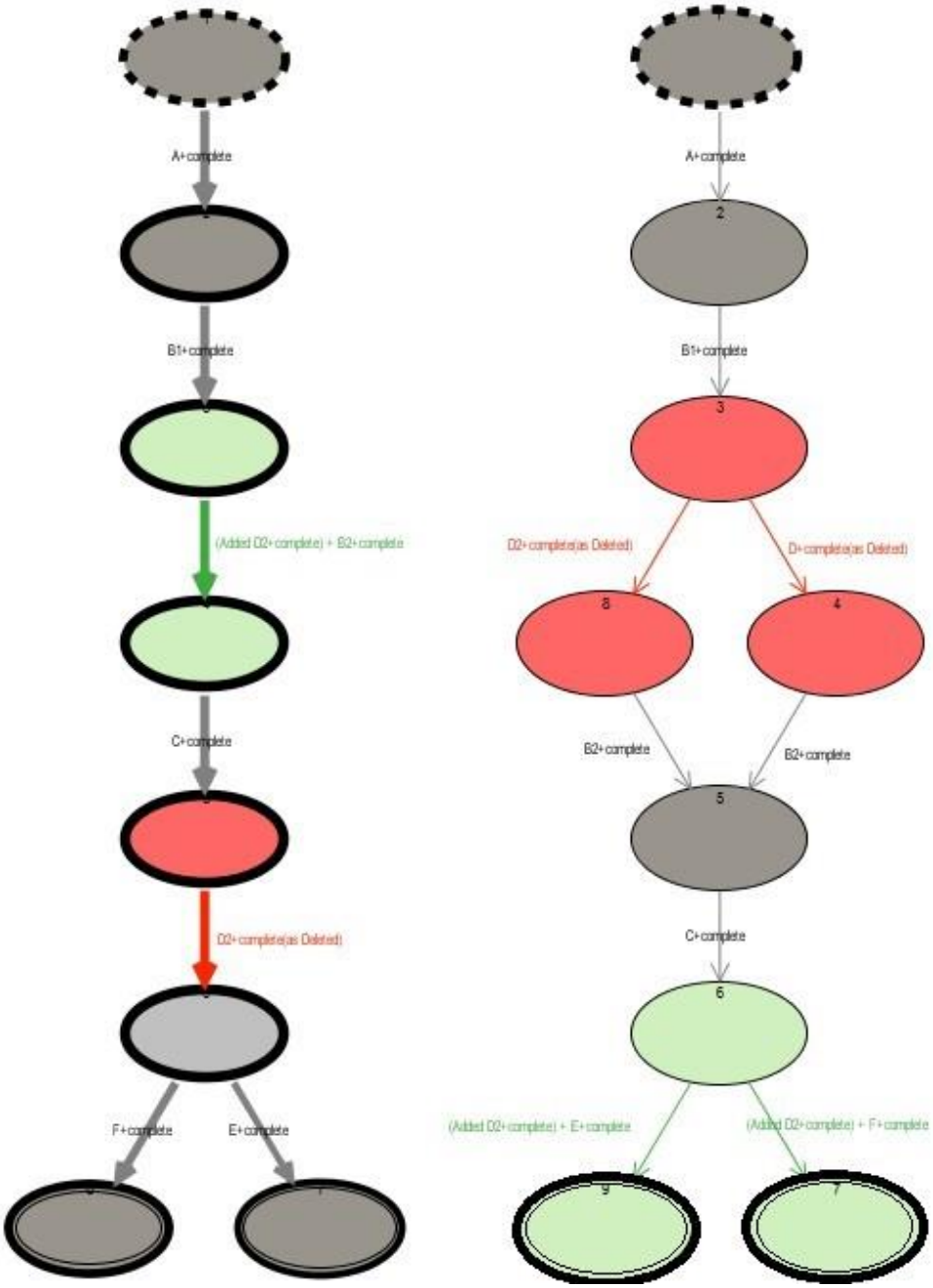

Рисунок 27 — Системы переходов для двух логов с большим количеством событий (пример 1)

В результате, по двум системам переходов (рис.27) можно сделать вывод, что в обоих логах встречается событие, которое отсутствует в другом

логе — это событие D_2 перед событиями E и F в первом логе и события D_2 и D после события B_1 во втором логе. Также, видно, что событие D_2 встречается в обоих логах, но в разных местах.

Пример 2. Рассмотрим работу плагину на примере еще одного большого лога событий. В каждом из логов хранится информация примерно о тысяче событий – 978 событий встречается в первом логе и 944 события во втором логе. Второй лог был создан вручную путем внесения изменений в первый лог с целью проверки обнаружения различий в логах событий.

В результате были построены системы переходов (рис. 28), с помощью которых наглядно видно, что два лога отличаются только на одно событие в начале. Остальная часть логов является полностью совпадающей. Для того, чтобы можно было рассмотреть подробнее, в плагине предусмотрена возможность увеличивать интересующую часть систем переходов (рис.29).

Различие в логах происходит ровно на одно событие, которое присутствует в первом логе, и отсутствует во втором логе и наоборот. В первом логе встречается событие «Register claim», которое отсутствует во втором логе. Во втором логе вместо этого события есть событие «Register claim 1» (измененное вручную). Остальные же события являются полностью эквивалентными и отличаются только частотой появления в логах событий.

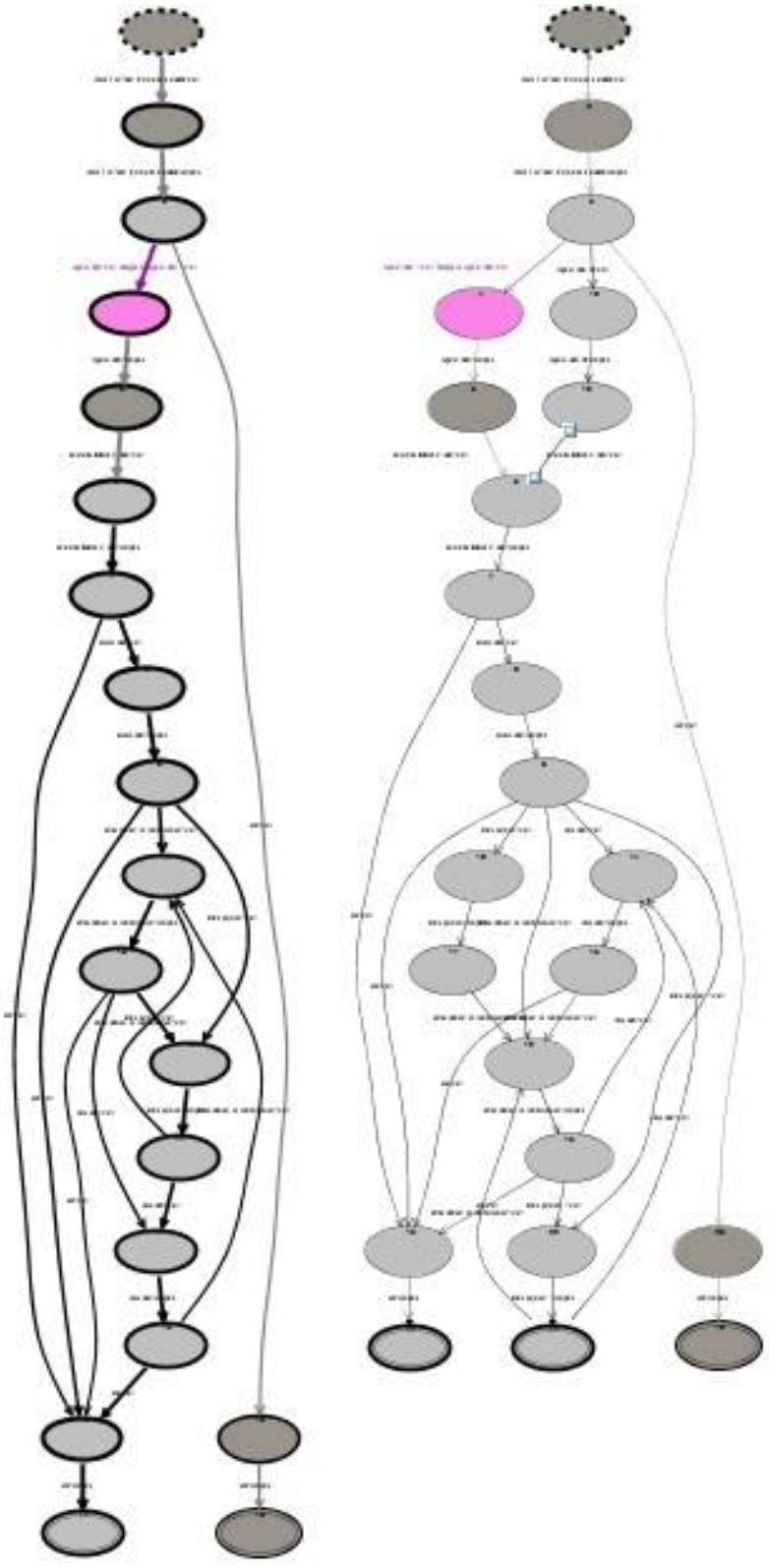

Рисунок 28 — Системы переходов, построенные на основе больших логов (пример 2)

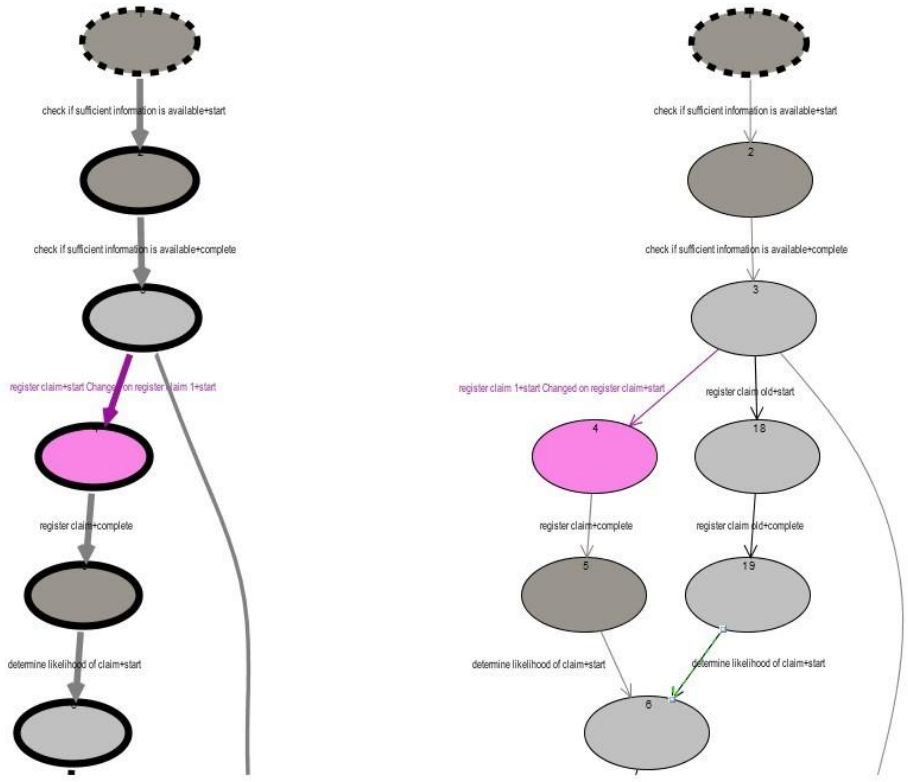

Рисунок 29 — Часть систем переходов из примера 2

Пример 3. Рассмотрим еще один пример, в котором встречаются все возможные различия. Даны два лога (взятые из примера 1), которые были немного изменены вручную. В первом логе стартовое событие A было изменено на событие B, которое больше нигде не встречается (ни в первом, ни во втором логах). В первом логе встречаются последовательности событий, такие как $BB_1B_2CD_2E$, $BB_1B_2CD_2F$, где каждое событие представлено отдельной буквой. Второй лог состоит из следующих последовательностей событий: AB_1DB_2CF,

$AB_1D_2B_2CE$, и так далее.

В результате, на построенной системе переходов (рис. 30) видно, что были найдены все возможные различия. Последовательности событий в логах начинаются с разных событий — событие B для первого лога и событие A для второго. На системе переходов (рис. 30) переходы, соответствующие этим событиям, отмечены фиолетовым цветом. Дальнейшие различия в данных логах были рассмотрены в примере 1.

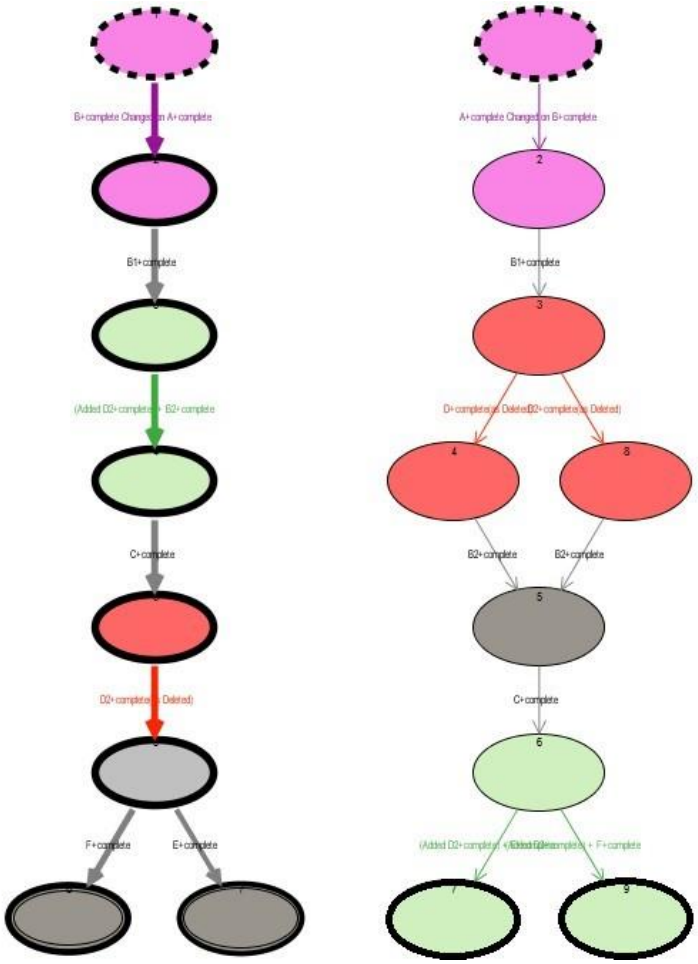

Рисунок 30. Системы переходов со всеми возможными различиями (пример 3)

3.3 Анализ достоверности и практической значимости результата

Значение проведенного исследования на практике состоит в предложении нового алгоритма для анализа логов события. При помощи алгоритма можно находить различия в логах событий до одного событий: удаленное (пропущенное) событие, добавленное (лишние) событие, измененное событие. Также есть возможность определять полностью идентичные последовательности событий.

Практическое значение работы состоит в визуализации найденных различий в логах событий.

Разработанный плагин к системе ProM позволил решить задачу визуализации различий в логах событий. В качестве исходных данных для плагина являются логи событий в формате XES.

Решение поставленной задачи заключается в представлении логов событий в виде систем переходов с помощью предназначенного для построения системы переходов плагина. В системе перехода каждый из переходов соответствуют одному событию из рассматриваемого лога, поэтому внимание акцентируется именно на переходах, а не вершинах. После того, как системы переходов построены, можно начинать сравнивать их между собой, отмечая переходы соответствующей цветовой маркировкой в случае нахождения различий. Алгоритм, представленный в рамках данной работы, основан на алгоритме поиска в ширину. Переходы двух систем переходов сравниваются по очереди до тех пор, пока все вершины и выходящие и них переходы первого лога не рассмотрены. Если на очередном шаге сравнения, для всех переходов одного состояния первого лога не найдены совпадающие переходы из тех, которые выходят из очередного состояния второго лога, то происходит переход вниз или вверх на одно состояние по одной из систем переходов. В случае нахождения совпадений, предыдущие рассматриваемые переходы отмечаются соответствующей маркировкой в зависимости от того, каким образом были найдены совпадения.

Метод Process Mining направлен на улучшение бизнес-процессов в различных областях и сферах деятельности, в том числе и в государственных организациях. Но поскольку данная дисциплина нова, в ходе реализации ее методов (особенно в сферах, где основным критерием успеха деятельности является не прибыль, а время выполнения или количество операций) могут возникнуть проблемы и ошибки. Далее будут перечислены основные проблемы, связанные с применением метода Process Mining при анализе деятельности бюджетного учреждения.

Сложность составления журнала событий. Поскольку журнал событий – основной документ, на основе которого происходит анализ процессов, очень важно, чтобы события в нем были хорошо обработаны, у них была четкая и понятная для человека формулировка.

К сожалению, одной из особенностей бюджетных организаций является огромный документооборот, стандарты по которому не редко меняются от года к году: к примеру, ведение одного документа может полностью прекратиться с последующей заменой его двумя другими документами.

Также сложность составления журнала событий заключается не только в возможной отмене ведения определенных документов, но и в их количестве и жизненном цикле. Нередко один небольшой бизнес-процесс бюджетной организации сопровождается огромным количеством бумаг, которые, кроме того, могут пройти жизненный цикл, равный нескольким годам. Наглядным примером могут послужить закупки по Федеральному закону № 44-ФЗ «О контрактной системе в сфере закупок товаров, работ, услуг для обеспечения государственных и муниципальных нужд». В соответствии с данным законом, даже небольшая закупка сопровождается такими документами, как: план закупок, план финансово-хозяйственной деятельности, заявка, техническое задание, обоснование цены, договора, акты приемки, экспертизы и прочее.

Вследствие этого создание стандартизированного журнала событий, который бы охватывал несколько лет, может оказаться трудной задачей.

Проблемы при построении модели бизнес-процесса. Данная проблема связана не столько с самим применением метода Process Mining, а сколько с объектом анализа – бюджетной организацией. Поскольку такое учреждение создается органами государственной власти или местного самоуправления, оно обычно становится частью системы других взаимосвязанных организаций.

Таким образом, к примеру, часть бизнес-процессов одной организации может переходить к другой без изменения официального ответственного лица. Поэтому возможны ситуации, когда бюджетное учреждение ответственно за бизнес-процесс, но по факту его не выполняет. Существуют и обратные ситуации.

Но даже если все бизнес-процессы бюджетной организации выполняются внутри ее, они зачастую настолько сложные, что постороннему человеку будет трудно в них разобраться.

В итоге эти особенности могут помешать построить реальную модель бизнес-процесса, из-за чего невозможно будет оценить реальную ситуацию в организации.

Трудность при реализации рекомендаций по совершенствованию. Даже после успешного завершения анализа по методу Process Mining и составления необходимых рекомендаций, бюджетные организации могут встретиться с проблемой реализации данных рекомендаций.

Основной причиной, опять же, является сложность уже работающих бизнес-процессов, кардинальные изменения в которых изменят систему в целом. Без грамотного и постепенного подхода при реализации рекомендаций невозможно будет добиться поставленного положительного результата.

Безусловно, специалист по анализу бизнес-процессов может встретиться и с другими проблемами во время применения метода Process Mining на практике. Среди них могут быть особенности сферы деятельности, нежелание персонала менять старые и проверенные процессы и другое. Но даже несмотря на то, что применение метода Process Mining в бюджетной организации может оказаться трудновыполнимым, его результаты позволят оптимизировать деятельность организации, которая нередко чрезвычайно время и трудозатратная.

ЗАКЛЮЧЕНИЕ

Актуальность темы исследования определяется тем, что область process mining стремительно развивается и ведущие компании усиливают работу в направлении развития инновационного продукта, представляющего собой метод анализа процессов, направленный на обнаружение, мониторинг и улучшение реальных процессов путем извлечения информации из журналов событий, содержащихся в информационных системах организации.

В результате выполнения работы были проанализированы существующие подходы и разработан алгоритм для нахождения различий в логах событий, который реализован в качестве плагина к инструменту ProM. С помощью разработанного алгоритма есть возможность находить различия в логах событий до одного события, идущего подряд — добавленное, удаленное, измененное событие в логе. Также, алгоритм позволяет находить полностью идентичные последовательности событий в логах.

Была выполнена программная реализация плагина для ProM с использованием среды Apache NetBeans. Разработанный плагин к системе ProM позволил решить задачу визуализации различий в логах событий. В качестве исходных данных для плагина являются логи событий в формате XES.

Цель достигнута, разработанный алгоритм и реализация его в рамках плагина к инструменту ProM может быть использован, как и непосредственно для сравнения двух логов событий экспертами в области работы с бизнес-процессами и другими заинтересованными лицами, так и в качестве основы для работы над алгоритмами сравнения логов событий между собой.

Результаты научно-исследовательской работы были опубликованы в 2 источниках:

– журнал III международной конференции научно-практической конференции «Актуальные вопросы современных научных исследований»;

– международный научный журнал «Молодой ученый»

БИБЛИОГРАФИЧЕСКИЕ ССЫЛКИ

1 Aalst W. M. P. van der Process Mining - Discovery, Conformance and Enhancement of Business Processes. — Springer, 2011. — С. I—XVI, 1—352.

2 IEEE Task Force on Process Mining Process Mining Manifesto // BPM 2011 Workshops. Т. 99 / под ред. F. Daniel, S. Dustdar, K. Barkaoui. — Springer-Verlag, Berlin, 2011. — С. 169—194. — (Lecture Notes in Business Information Processing).

3 Using process mining for the analysis of an e-trade system: A case study / A. Mitsyuk [и др.] // Business Informatics. — 2014. — Т. 29, № 3. — С. 15—27.

4 Rubin V., Lomazova I., Aalst W. M. van der Agile Development with Software Process Mining // In proceedings: ICSSP 2014, Nanjing, Jiangsu, China. — ACM, 2014. — С. 70—74.

5 Process Mining Can Be Applied to Software Too! / V. Rubin [и др.] // Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. — NY: ACM, 2014.

6 Jacob Perkins, Python 3 Text Processing with NLTK 3 Cookbook – Packt Publishing, 2014 – 304 с.

7 James J. Data never sleeps: How much data is generated every minute //Domo Blog. – 2012. – Т. 8.

8 Майер-Шенбергер, В. Большие данные. Революция, которая изменит то, как мы живем, работаем и мыслим / Виктор Майер-Шенбергер, Кеннет Кукьер; пер. с англ. Инны Гайдюк. — М. : Манн, Иванов и Фербер, 2014 – 235 с.

9 Carmona J., Cortadella J., Kishinevsky M. New Region-Based Algorithms for Deriving Bounded Petri Nets // IEEE Transactions on Computers. — 2010. — Т. 59, № 3. — С. 371—384.

10 Generating event logs for high-level process models / A. A. Mitsyuk [и др.] // Simulation Modelling Practice and Theory. — 2017. — Т. 74. — С. 1—16.

11 Rigin A., Shershakov S. SQLite RDBMS Extension for Data Indexing Using B-tree Modifications // Proceedings of the Institute for System Programming of the RAS (Proceedings of ISP RAS). — 2019. — Т. 31, № 3. — С. 203—216. — URL: https://ispranproceedings.elpub. ru/jour/article/view/1188/948.

12 Van Eck M. L. et al. PM2: A Process Mining Project Methodology //International Conference on Advanced Information Systems Engineering. – Springer, Cham, 2015. – С. 297-313.

13 Conforti, Raffael. Beyond Tasks and Gateways: Discovering BPMN Models with Subprocesses, Boundary Events and Activity Markers / Conforti, Raffael, Dumas, Marlon, García-Bañuelos, Luciano, La Rosa, Marcello // International Conference on Business Process Management. - 2014. - С. 101-117.

14 C. W. Gunther. Fuzzy mining – adaptive process simplification based on multi-perspective metrics / C. W. Gunther and W. M. P. van der Aalst // In Business Process Management, Springer. - 2007. - С. 328–343.

15 R. P. Jagadeesh Chandra Bose. Discovering Hierarchical Process Models Using ProM / R. P. Jagadeesh Chandra Bose, E. H. M. W. Verbeek, and W. M. P. van der Aalst // Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012 (т. 107). -С. 33–48

16 R. Conforti. BPMN miner: Automated discovery of BPMN process models with hierarchical structure / R. Conforti, M. Dumas, L. García Bañuelos, and M. La Rosa // Information Systems, 2016. - С. 284 – 303.

17 Leemans, Sander. Discovering Block-Structured Process Models from Event Logs - A Constructive Approach / Leemans, Sander, Fahland, Dirk, Aalst, Wil // International Conference on Applications and Theory of Petri Nets and Concurrency, 2013. - С. 311-329.

18 Jacob Perkins, Python 3 Text Processing with NLTK 3 Cookbook – Packt Publishing, 2014 – 304 с.

19 James J. Data never sleeps: How much data is generated every minute //Domo Blog. – 2012. – Т. 8.

20 Jans M., Alles M., Vasarhelyi M. The case for process mining in auditing: Sources of value added and areas of application //International Journal of Accounting Information Systems. – 2013. – Т. 14. – №. 1. – С. 1-20.

21 Программа синтеза моделей процессов / Т. Е. Федосова. — Текст : непосредственный // «Актуальные вопросы современных научных исследований» III Международная научно-практическая конференция. — 2023. — № 48. — С. 46-49.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Воронцов, К. В. Лекции по алгоритмам кластеризации и многомерного шкалирования [Электронный ресурс]. – Режим доступа: http://www.ccas.ru/voron/download/Clustering.pdf - 05.12.2022

2 Воэльо, Л.П. Построение систем машинного обучения на языке Python / Воэльо Л.П., Ричарт В. – 2-е изд.: пер. с англ. Слинкин А.А. – М.: ДМК Пресс, 2016 – 302 с.: ил.

3 Костевич Л.С. Математическое программирование (информационные технологии оптимальных решений) – Минск, ООО «Новое знание», 2003, – 424 с.

4 Майер-Шенбергер, В. Большие данные. Революция, которая изменит то, как мы живем, работаем и мыслим / Виктор Майер-Шенбергер, Кеннет Кукьер; пер. с англ. Инны Гайдюк. — М. : Манн, Иванов и Фербер, 2014 – 235 с.

5 Маккинли, У. Python и анализ данных / Уэс Маккинли; пер. с англ.. – М.: ДМК Пресс, 2015 – 482 с.

6 Программа синтеза моделей процессов / Т. Е. Федосова. — Текст : непосредственный // «Актуальные вопросы современных научных исследований» III Международная научно-практическая конференция. — 2023. — № 48. — С. 46-49.

7 Федосова, Т. Е. Программа синтеза моделей процессов / Т. Е. Федосова, С. Г. Самохвалова. — Текст : непосредственный // Молодой ученый. — 2022. — № 43. — С. 223-224.

8 Флах, П. Машинное обучение. Наука и искуство построения алгоритмов, которые извлекают знания из данных / Флах Петер; пер. с англ. А. А. Слинкин. – М.: ДМК Пресс, 2015 – 400 с.

9 Фрэнкс, Б. Укрощение больших данных: как извлекать знания из массивов информации с помощью глубокой аналитики / Билл Фрэнкс ; пер. с англ. Андрея Баранова. — М. : Манн, Иванов и Фербер, 2014. — 352 с

10 ГОСТ 2.051-2013 Единая система конструкторской документации (ЕСКД). Электронные документы. – Взамен ГОСТ 2.051–2006; введ. 2014–06–01. –М. : Стандартинформ, 2014. –17 с.

11 Звонарев, С. В. Основы математического моделирования: учеб. пособие / С. В. Звонарев. – Екатеринбург: Изд-во Урал. ун-та, 2019. – 112 с.

12 Agrawal R., Gunopulos D., Leymann F. Mining process models from workflow logs //International Conference on Extending Database Technology. – Springer, Berlin, Heidelberg, 1998. – С.483.

13 Becker J. et al. In search of information systems (grand) challenges //Business & Information Systems Engineering. – 2015. – Т. 57. – №. 6. – С. 390.

14 Cook J. E., Wolf A. L. Discovering models of software processes from event-based data //ACM Transactions on Software Engineering and Methodology (TOSEM). – 1998. – Т. 7. – №. 3. – С. 249.

15 Cook J. E., Wolf A. L. Event-based detection of concurrency //ACM SIGSOFT Software Engineering Notes. – ACM, 1998. – Т. 23. – №. 6. – С. 45.

16 Cook J. E., Wolf A. L. Software process validation: quantitatively measuring the correspondence of a process to a model //ACM Transactions on Software Engineering and Methodology (TOSEM). – 1999. – Т. 8. – №. 2. – С. 176.

17 Curtis B., Kellner M. I., Over J. Process modeling //Communications of the ACM. – 1992. – Т. 35. – №. 9. – С. 90.

18 Eder J., Olivotto G. E., Gruber W. A data warehouse for workflow logs //International Conference on Engineering and Employment of Cooperative Information Systems. – Springer, Berlin, Heidelberg, 2002. – С. 15.

19 Grigori D. et al. Improving business process quality through exception understanding, prediction, and prevention //VLDB. – 2001. – Т. 1. – С. 168.

20 Hammer M., Champy J. Reengineering the Corporation: Manifesto for Business Revolution, A. – Zondervan, 2009.

21 Herbst J. A machine learning approach to workflow management //European conference on machine learning. – Springer, Berlin, Heidelberg, 2000. – С. 194.

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

23 Herbst J. Ein induktiver ansatz zur akquisition und adaption von workflow-modellen. – Tenea Verlag Ltd., 2004.

24 Herbst J., Karagiannis D. An inductive approach to the acquisition and adaptation of workflow models //Proceedings of the IJCAI. – 1999. – T. 99. – C. 57.

25 Herbst J., Karagiannis D. Integrating machine learning and workflow management to support acquisition and adaptation of workflow models //Intelligent Systems in Accounting, Finance & Management. – 2000. – T. 9. – №. 2. – C. 92.

26 Herbst J., Karagiannis D. Integrating machine learning and workflow management to support acquisition and adaptation of workflow models //Intelligent Systems in Accounting, Finance & Management. – 2000. – T. 9. – №. 2. – C. 92.

27 IDS Scheer. ARIS Process Performance Manager (ARIS PPM). http://www.idsscheer.com, 2002.

28 ISO 9000: International Standards for Quality Management. Genève, Switzerland, 2015. url: http://www.iso.org/iso/home/standards/management-standards/iso_9000.htm (pages 3, 6, 51).

29 Jacob Perkins, Python 3 Text Processing with NLTK 3 Cookbook – Packt Publishing, 2014 – 304 c.

30 James J. Data never sleeps: How much data is generated every minute //Domo Blog. – 2012. – T. 8.

31 Jans M., Alles M., Vasarhelyi M. The case for process mining in auditing: Sources of value added and areas of application //International Journal of Accounting Information Systems. – 2013. – T. 14. – №. 1. – C. 20.

32 Kerremans M. Gartner market guide for process mining //Report G00353970. Gartner. – 2018.

33 Kiepusewski B. Expressiveness and suitability of languages for control flow modelling in workflows: дис. – Queensland University of Technology, Brisbane, 2003.

34 Korobov, M. Morphological Analyzer and Generator for Russian and Ukrainian Languages / M. Korobov // Analysis of Images, Social Networks and Texts, 2015 – 320-332 с.

35 Mannhardt F. Multi-perspective process mining //CEUR Workshop Proceedings. – 2018. – Т. 2196. – С. 41-45.

36 Mannila H., Rusakov D. Decomposition of event sequences into independent components //Proceedings of the 2001 SIAM International Conference on Data Mining. – Society for Industrial and Applied Mathematics, 2001. – С. 1-17.

37 Mannila H., Toivonen H., Verkamo A. I. Discovery of frequent episodes in event sequences //Data mining and knowledge discovery. – 1997. – Т. 1. – №. 3. – С. 259-289.

38 Mans R. et al. Process mining techniques: an application to stroke care //MIE. – 2008. – Т. 136. – С. 573-578.

39 Mans R. S. et al. Application of process mining in healthcare–a case study in a dutch hospital //International joint conference on biomedical engineering systems and technologies. – Springer, Berlin, Heidelberg, 2008. – С. 425-438.

40 Maruster L. et al. Automated discovery of workflow models from hospital data //B. Kr€ oose, M. de Rijke. – 2001. – Т. 18.

41 Maruster L. et al. Process mining: Discovering direct successors in process logs //International Conference on Discovery Science. – Springer, Berlin, Heidelberg, 2002. – С. 364-373.

42 Maxeiner M. K., Küspert K., Leymann F. Data Mining von Workflow-Protokollen zur teilautomatisierten Konstruktion von Prozeßmodellen //Datenbanksysteme in Büro, Technik und Wissenschaft. – Springer, Berlin, Heidelberg, 2001. – С. 75-84.

43 Parekh R., Honavar V. Automata induction, grammar inference, and language acquisition //Dale, Moisl, and Somers, editors, Handbook of Natural Language Processing. New York: Marcel Dekker. – 2000. – Т. 184.

44 Артемова, А. О. Математическое моделирование, алгоритмы и программы управления манипуляторами: автореф. дис…канд. физ.-матем.

наук : 05.13.18 / А. О. Артемова.– Ульяновск: Изд-во Ульянов. гос ун-та, 2013. – 21 с.

45 Горностаева, Т. Н. Математическое и компьютерное моделирование: учеб. пособие [Электронный ресурс] / Т. Н. Горностаева, О. М. Горностаев. – Режим доступа: https://izd-mn.com/PDF/50MNNPU19.pdf – 02.12.2022.