Модел. и анализ информ. систем. Т. **21**, № **4** (2014) 181–198

УДК 517.9

Синтез моделей процессов по журналам событий с шумом[1]

Мицюк А. А., Шугуров И. С.

Национальный Исследовательский Университет Высшая Школа Экономики
Международная лаборатория процессно-ориентированных информационных систем
125319 Россия, г. Москва, Кочновский проезд, д. 3

e-mail: amitsyuk@hse.ru, shugurov94@gmail.com

получена 23 августа 2014

Ключевые слова: Извлечение и анализ процессов, сети Петри, журналы событий, генерация журналов событий, ProM, Process mining

Извлечение процессов (process mining) – новая и активно развивающаяся область исследований, тесно связанная с управлением процессами, формальными моделями процессов и извлечением данных (data mining). Одна из основных задач извлечения процессов – синтез (извлечение) модели процесса на основании анализа журнала событий. Разработан широкий спектр алгоритмов для извлечения, анализа и усовершенствования моделей процессов. Журналы событий реальных систем часто содержат шум различных видов.

В данной работе описываются основные причины возникновения шума в журналах событий и изучается влияние шума на эффективность применения основных алгоритмов извлечения процессов. Приводятся экспериментальные результаты применения основных алгоритмов извлечения моделей процессов к искусственным журналам событий с шумами различного типа. Для этого специальным образом сгенерированные журналы событий с шумом обрабатывались с использованием четырех основных методов извлечения процессов. Хотя современные алгоритмы могут справляться с некоторыми типами шума, в большинстве случаев их применение не приводит к получению удовлетворительного результата. Таким образом, существует необходимость в разработке более совершенных подходов для журналов событий с шумом.

Введение

Извлечение процессов (process mining) – собирательное название различных методов, предназначенных для синтеза, анализа и усовершенствования моделей процессов в результате работы с журналами событий информационных (и других) систем.

[1]Работа выполнена в рамках Программы фундаментальных исследований НИУ ВШЭ в 2014 году.

Выделяют три основных раздела извлечения процессов: (1) синтез моделей процессов, (2) проверка соответствия между моделями процессов и журналами событий и (3) усовершенствование моделей процессов с использованием различной информации из журналов событий и не только [1]. Исходной информацией для применения подходов извлечения моделей являются журналы событий процесса, проверки соответствия – модель и журнал событий, усовершенствования – модель, журнал событий и, возможно, дополнительные данные. Разработано большое количество алгоритмов, предназначенных для решения задач в каждом из этих трех разделов [1–4,9–12,22,23]. Первым и простейшим алгоритмом извлечения модели процесса является Альфа-алгоритм [2]. Существует большое количество вариаций этого алгоритма.

Среди других алгоритмов извлечения стоит выделить три основные, задающие целые семейства подходов. Эвристические подходы используют частотные характеристики процесса для синтеза модели [23]. Алгоритмы с использованием линейного программирования работают на основаниях теории регионов [22]. Индуктивные алгоритмы предназначены для синтеза структурированных моделей [11]. В качестве результата работы алгоритма извлечения можно получить модель процесса в одном из формализмов: сеть Петри, сеть потоков работ (WF-сеть), сеть BPMN, эвристическая сеть и другие [1].

Все эти алгоритмы реализованы в качестве расширений среды для извлечения и анализа моделей под названием ProM Framework [8,13]. В данной работе использовались все четыре основных подхода к извлечению модели процесса: альфа-алгоритм, эвристические алгоритмы, алгоритмы с использованием линейного программирования, индуктивный алгоритм. Использовались версии алгоритмов для среды ProM версии 6.3.

Необходимым условием для применения извлечения процессов является наличие у каждой событийной записи в журнале минимум двух составляющих: имени и временной метки. Модель процесса строится по причинным зависимостям. Последовательность событий задается их временными метками. Переходы (с именами, соответствующими именам событий в журнале) в извлекаемой модели располагаются последовательно. Если временные метки событий совпадают, соответствующие переходы располагаются параллельно.

В дополнение к этим необходимым составляющим событийная запись может содержать и другие: имена исполнителей, численные данные, записи о ресурсах, вовлеченных в исполнение операции. Существуют алгоритмы, использующие эти составляющие для извлечения моделей процессов в других формализмах, например, в виде социальных сетей.

В реальной жизни редко случается, что информационная система работает идеально [18, 19]. Внешний мир вносит коррективы: технические помехи, неисправности, ошибки людей, участвующих в исполнении процесса. Все эти события влияют на записанное в журнале событий поведение систем. Данная работа посвящена изучению различных помех, встречающихся в событийных журналах.

Основные результаты данной работы: (1) приводится классификация различных типов шума, встречающихся в журналах событий; (2) приводятся результаты экспериментов по исследованию влияния шума в журналах событий на эффективность применения методов извлечения моделей процессов; (3) обосновывается необходи-

мость учёта различной дополнительной информации для извлечения качественных моделей; (4) описываются сильные и слабые стороны алгоритмов извлечения процессов.

1. Сети потоков работ

В данной работе для представления процессов в информационной системе используются сети потоков работ – специальный класс сетей Петри [2]. Сеть потоков работ имеет одно начальное и одно конечное состояния, а все другие состояния расположены на путях между ними.

На рисунке 1 показан пример сети потоков работ, моделирующей простейший медицинский процесс. Круги обозначают позиции в сети, а квадраты – переходы. Выделены начальная i и конечная o позиции. Внутри позиций могут изображаться закрашенные круги, изображающие маркеры в позициях. Текущая разметка изображается добавлением необходимого количества маркеров (закрашенных кругов) в каждую позицию. Разметка обозначает текущее состояние процесса. Начальной разметкой для сети потоков работ является маркер в начальной позиции.

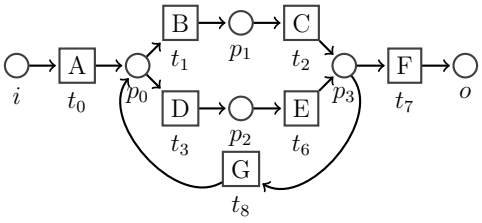

Рис. 1. Пример сети потоков работ

Возможный процесс, описываемой моделью на рисунке 1: в больницу поступает пациент (маркер в начальной позиции i) и по результатам анализов принимается решение (A) о дальнейшем лечении – консервативном (B, C) или оперативном (D, E); после проведенного курса лечения пациент либо выписывается из больницы (F, маркер в позиции o), либо снова отправляется на лечение (G).

Исполнение действия или операции в процессе моделируется срабатыванием перехода. При срабатывании перехода потребляется необходимое количество маркеров из позиций, входных для сработавшего перехода. Одновременно в выходные для соответствующего перехода позиции помещаются произведенные маркеры. Таким образом, срабатывание перехода изменяет разметку модели (состояние системы).

В классической сети Петри переходы и позиции не носят имен. Тем не менее, при анализе процессов требуется именовать действия или операции процесса. Поэтому в данной работе рассматриваются именованные сети потоков работ, в которых каждый узел имеет имя.

Строгое формальное описание сетей потоков работ содержится, например, в [1,2].

2. Журналы событий, формат XES

Журналы событий информационных систем могут иметь разные форматы. В общем случае, журнал событий – это набор некоторых текстовых записей, содержащий информацию о событиях и действиях процесса. При извлечении процессов используют формализованное определение журнала событий [1]. *Журнал состоит из записей*, соответствующих событиям. Процесс, как правило, исполняется многократно. Каждая запись в журнале должна принадлежать строго одному из исполнений процесса. Записи о событиях в журнале связаны с выполнением какой-либо операции процесса. При моделировании процесса именованной сетью потоков работ выполнение операции соответствует срабатыванию перехода с соответствующим именем в сети. Дополнительно к имени операции каждая событийная запись в журнале содержит временную метку, которая отражает время (и дату) исполнения операции. Событийные записи *могут* содержать и дополнительную информацию (имена или роли исполнителей действия, стоимости, числовые или текстовые данные). Таблица 1 содержит фрагмент журнала событий, соответствующего модели на рисунке 1.

Трасса	ID	Имя события	Метка времени	Пациент
1	1001141	A (поступление, анализы)	10-01-2014:15.12	Иванов
1	1001142	B (поступление в терапию)	10-01-2014:19.10	Иванов
1	1501141	C (выписка из терапии)	15-01-2014:10.10	Иванов
1	1601141	F (анализы, выписка)	16-01-2014:9.14	Иванов
2	1101141	A (поступление, анализы)	11-01-2014:18.12	Петров
2	1101142	B (поступление в терапию)	11-01-2014:20.00	Петров
2	1301141	C (выписка из терапии)	13-01-2014:18.10	Петров
2	1401141	G (решение о лечении)	14-01-2014:8.23	Петров
3	1401142	A (поступление, анализы)	14-01-2014:15.00	Семенов
2	1401143	D (поступление в хирургию)	14-01-2014:16.10	Петров
2	2001141	E (выписка из хирургии)	20-01-2014:10.25	Петров
2	2101145	F (анализы, выписка)	21-01-2014:9.01	Петров
...	...	...	...	...

Таблица 1. Пример фрагмента из журнала событий, соответствующего модели на рисунке 1.

Более формально, журнал событий – это мультимножество трасс, где каждая трасса содержит ровно одно исполнение процесса в виде последовательности событийных записей. Каждая событийная запись содержит имя операции, временную метку и другие дополнительные атрибуты. Каждая трасса может встречаться в журнале много раз [1].

Для хранения журналов событий существует открытый формат OpenXES [14]. OpenXES – это xml-схема, описывающая формат текстовых файлов, содержащих поведение информационной системы в виде потока событий. Существует проработанный стандарт, описывающий тонкости использования формата. На момент написания данной работы новейшей версией является XES Standard Definition 2.0 [15].

Стандарт обладает всеми достоинствами (и недостатками) XML: простотой обработки, расширяемостью, простотой проверки корректности, относительно удобными

операциями чтения и записи. В то же время, говорить о существенной эффективности не приходится. Количество памяти, необходимой для хранения разметки очень велико по отношению к полезным данным. Операции поиска работают медленно.

Большинство инструментов для извлечения процессов, как исследовательских, так и научных, поддерживают стандарт XES. Фактически – это основной формат, используемый ProM 6 Framework наряду с MXML. В данной работе для хранения журналов событий также используется XES. Важно отметить, что стандарт является расширяемым, а потому каждый может разрабатывать свои специализированные форматы журналов событий. В этом случае, однако, нельзя гарантировать работоспособность стандартных инструментов, предназначенных для обработки стандартизированных журналов событий.

3. Соответствие моделей и журналов событий

Качество модели информационной системы оценивается прежде всего ее способностью воспроизводить корректное поведение системы и отбраковывать некорректное. Для моделей процессов, извлеченных из журналов событий, сформулированы основные аспекты качества: (1) *соответствие*, (2) *простота*, (3) *точность* и (4) *обобщенность* [1].

Соответствие модели определяется только для конкретного журнала событий. 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.

Для расчета соответствия журнал событий последовательно исполняется моделью. Устанавливается начальная разметка модели (один маркер в начальной позиции). Алгоритм последовательно проходит каждую трассу журнала. Для каждого очередного события возможны два варианта: (1) переход с соответствующим именем активен и может сработать или (2) нет активного перехода с соответствующим именем. В первом случае переход срабатывает, разметка модели меняется, а алгоритм переходит к следующему событию журнала. Во втором случае в некоторых из позиций, необходимых для срабатывания соответствующего перехода модели, нет достаточного количества маркеров. В эту позицию добавляется искусственный маркер, который позволяет продолжить исполнение модели. Позиция помечается как содержащая соответствующее количество *пропавших* маркеров. Если при дальшейшем исполнении снова возникает подобная ситуация, количество пропавших маркеров в рассматриваемой позиции увеличивается.

Возможна также ситуация, что при достижении финальной разметки в сети остаются маркеры, которые не были потреблены в ходе исполнения. Позиции, в которых эти маркеры расположены, помечаются как содержащие *лишние* маркеры.

В [17] соответствие модели N и журнала событий L определяется следующим образом:

$$fitness = \frac{1}{2}\left(1 - \frac{\sum_{\sigma \in L} \cdot m_\sigma}{\sum_{\sigma \in L} \cdot c_\sigma}\right) + \frac{1}{2}\left(1 - \frac{\sum_{\sigma \in L} \cdot r_\sigma}{\sum_{\sigma \in L} \cdot p_\sigma}\right),$$

где m_σ – количество пропавших маркеров, r_σ – количество лишних маркеров, c_σ – количество потребленных маркеров, p_σ – количество произведенных маркеров, σ – конкретная трасса.

Существуют и другие, более совершенные способы оценки соответствия модели и журнала событий. В [3–5] предложена методика расчета соответствия на основе использования так называемых *выравниваний*. Составляются таблицы соответствий трасс в логе и исполнений в модели, а затем решается оптимизационная задача для выбора наилучших выравниваний. Несоответствиям в выравниваниях присваиваются стоимости, которые затем суммируются в итоговую оценку. Описанный метод довольно сложно устроен и очень требователен к вычислительным ресурсам, но дает более качественную и реалистичную оценку соответствия. Подробное описанием метода приведено в [3–5]. В данной работе использовались оба способа оценки. Результаты, приводимые в статье, получены, главным образом, с использованием второй методики. Благодаря применению выравниваний алгоритм, основанный на этой методике, выдает более точную оценку [3].

Простота модели может определяться различными способами. Содержательно эта метрика определяет простоту восприятия модели для произвольного эксперта. Простой можно считать такую модель, которая имеет хорошее соответствие с заданным журналом событий и при этом содержит минимальное количество узлов (позиций и переходов), а также дуг, их связывающих. Соответственно, сложной считается запутанная модель, содержащая большое количество узлов и дуг. Конкретные значения метрики, соответствующие понятиям «простой» или «сложный», определяются исходя из мнения эксперта и конкретных условий решаемой задачи. В данной работе в качестве метрик простоты использовалась оценка размера и запутанности модели по количеству узлов и дуг.

Использование только критериев простоты и соответствия не всегда позволяет получать модели, обладающие требуемыми характеристиками. Можно построить предельно простую модель, которая будет допускать практически любое поведение, в том числе и записанное в заданном журнале событий. Для исключения таких случаев вводятся критерии точности и обобщенности. *Точная* по отношению к некоторому журналу модель допускает поведение, записанное в журнале, но не допускает иного поведения. Соответственно, неточная модель допускает любое поведение. Соблюдение баланса точности модели – важный вопрос при анализе и конструировании моделей процессов. В данной работе использовалась техника оценки точности, предложенная в [3, 6].

Слишком точная модель не имеет смысла, так как является, по сути, другим форматом записи журнала событий (не допускает обобщения). Моделирование и анализ процессов имеют целью построение моделей, которые не только умеют воспроизводить уже известное поведение, но пригодны и для получения предположений относительно неизвестного, но возможного поведения. Критерий *обобщенности* модели показывает, насколько обширные варианты поведения она допускает. Модель, допускающая только поведение, записанное в исходном журнале событий,

называют переобученной, и ценность ее не велика. Модель, допускающую сколь угодно различное поведение наравне с записанным в журнале, называют недоученной. Методика расчета обобщенности, используемая в данной работе, предложена в [3, 6].

Соблюдение баланса и выработка равновесных критериев качества модели – одна из открытых и важных задач в области извлечения процессов. В данной работе для оценки качества моделей используются метрики *соответствия*, *точности* и *простоты* модели. Построение хороших метрик обобщенности – не до конца решенная задача. Существующие метрики имеют свои недостатки и требуют объективного оценивания. Например, результаты вычисления обобщенности по основной существующей методике почти всегда очень высоки. Таким образом, можно говорить о том, что метрика недостаточно хорошо разделяет модели на классы по обобщенности. За подробностями можно обратиться к [7].

4. Классификация шумов в журналах событий

При решении практических задач с использованием подходов извлечения процессов исследователь имеет в качестве исходных данных журнал событий, в котором содержится поведение системы за некоторый промежуток времени. В некоторых случаях может быть доступна также модель процесса, синтезированная на основе журнала или сконструированная вручную. Задачей, как правило, является построение модели реального поведения системы и выявление узких мест. Иногда неизвестно, есть ли вообще узкие места и неэффективные составляющие в информационной системе. Построение модели реального поведения и ее анализ могут позволить определить, стоит ли проводить дальнейшее исследование системы. Содержащиеся в журналах событий помехи могут существенно затруднять получение корректных моделей.

В этом разделе приводится классификация различных типов помех и искажений, которые могут появиться в журналах событий информационных систем. Основные причины появления помех в журналах событий можно разделить на три большие группы: (1) технические и системные сбои, (2) неправильная трактовка событий, (3) ошибочная запись временных меток. Характер помех, возникающих в журнале событий, отличен для каждой группы.

Системные сбои приводят к следующим типам помех: пропущенные события в журнале (переход в модели сработал, что означает выполнение операции процесса, тогда как запись в журнал событий не была произведена), лишние события в журнале (в результате сбоя в журнале появляется запись, тогда как в реальности соответствующая операция не производилась), случайные искажения журнала событий. Любой журнал событий является определенным образом форматированной текстовой информацией, хранящейся в цифровом виде, в базах данных или в аналоговом формате (в виде распечаток, записей). Очевидно, что системный сбой может привести к повреждению записей, внести случайные помехи, сбить форматирование и т.д. В результате сбоя полноценное чтение журнала событий становится невозможным. Чаще всего в таких случаях повреждается только часть журнала, а формат хранения журнала позволяет осуществить его чтение, отфильтровав поврежденные разделы.

Как уже говорилось, в данной работе рассматриваются журналы в формате

XES. В силу своей XML-природы этот формат позволяет исправлять журналы событий путем отфильтровывания испорченных трасс. Более подробно об исправлении журналов событий можно узнать из [16]. Очевидно, что нет смысла рассматривать сильно испорченные журналы событий. В этом случае крайне сложно отличить правильно записанное поведение системы от сбойного, так как испорченная часть может существенно превосходить правильную.

Помимо чисто технических сбоев, к проблемам могут приводить и другие причины. Источником большого количества проблем в журналах событий является неправильная трактовка процессов или исполняемых операций. Под неправильной трактовкой здесь понимается прежде всего ошибочная запись идентификатора события или дополнительной информации (когда, например, вместо идентификатора одного из событий (действий) в журнал записывается идентификатор другого события (действия), которое также присутствует в процессе). Такие ошибки могут возникать по различным причинам. К ним приводит неправильная настройка программного или аппаратного обеспечения информационной системы. Ошибки в коде программного обеспечения информационной системы также могут вызывать запись ложных событий в журнал. Больше всего подобных ошибок возникает в случае, если информация о выполненном действии или событии вносится человеком (пользователем информационной системы). Людям свойственно ошибаться. Даже когда речь идет об ответственных операциях, нельзя считать вероятность ввода ошибочных данных нулевой.

Другим видом ошибочных записей в журналах событий являются записи с помехами во временных метках. Как уже было отмечено выше, каждая запись журнала должна содержать временную метку, в которой содержится привязка события или действия к дате и времени. Для построения моделей важна последовательность, в которой происходят события. Многие алгоритмы извлечения моделей учитывают только подобные зависимости (в других алгоритмах данная информация играет ключевую роль). Таким образом, ошибки данного типа являются одними из самых существенных в вопросах анализа работы информационной системы по журналам событий.

Помехи во временных метках могут носить как случайный, так и периодический характер. В качестве примера можно привести тот факт, что люди редко выставляют точное время исполнения ими того или иного действия (или длительности промежутка). Люди мыслят скорее категориями «я делал это примерно 15 минут», чем «данную работу я выполнил за 13 минут 29 секунд». Размер и величина такого округления отличаются и зависят от культуры, характера и профессии человека. Если на опасном производстве или в аэропорту будет записано время с точностью до десятых секунды, то в поликлинике или государственном учреждении сотрудник, скорее всего, довольно сильно округлит время выполнения задачи. Подобные периодические помехи могут давать довольно странные результаты в ходе анализа журналов событий. Нередко пользователи информационных систем заполняют отчетные поля задним числом, что также приводит к округлениям и появлению периодического шума.

В журналах событий реальных информационных систем встречаются и другие характерные помехи, вызванные ошибочным вводом временных меток. Выбирается необходимый день, но месяц или год остается выставленным по умолчанию. Оши-

бочно вводится время в системах с 12-часовым форматом задания времени. Многие из подобных ошибок устраняются путем перепроектирования интерфейса пользователя. Например, замена полей ввода на 24-часовой формат уменьшает количество ошибок при вводе времени.

Наконец, к проблемам с временными метками приводят различные рассинхронизации узлов информационной системы. Часы могут быть настроены неправильно на некоторых компьютерах или сетевых серверах, время установлено неверно. Ошибки в коде программного обеспечения также могут приводить к неправильно выставленным временным меткам.

5. Исследование журналов событий с шумами

В этом разделе приводятся результаты исследования влияния различных типов шума в журналах событий на изменение свойств синтезируемой на их основе модели.

Экспериментальная установка. Для тестирования использовался генератор журналов событий, разработанный в лаборатории ПОИС [20, 21]. Генератор журналов событий *Gena* разработан в виде плагина для среды ProM 6.3 Framework [8, 13], которая предназначена для работы с алгоритмами извлечения и анализа процессов (Process mining). Среда ProM содержит большое количество расширений и позволяет дополнительно разрабатывать пользовательские. Одним из таких расширений и является генератор Gena.

Подход к генерации журнала событий, реализованный в программе Gena, содержит три важнейшие составляющие: (1) генерация одиночного журнала, (2) генерация набора журналов событий и (3) генерация дополнительных искажений в журнале событий. Технически процесс генерации журнала событий выглядит следующим образом.

1. Во все позиции, перечисленные во множестве начальной разметки, помещаются маркеры разметки. Маркеры при этом имеют специальный идентификатор трассы. Таким образом, одновременно в системе могут находиться маркеры, соответствующие различным трассам (различным экземплярам процесса).

2. Из всех переходов выбираются активные, которые могут сработать в заданной разметке.

3. Из множества активных переходов выбирается один случайным образом, либо в соответствии с заданными пользователем предпочтениями. Если из одной позиции исходят дуги в несколько переходов, то есть возможно разветвление управляющего потока, пользователь плагина имеет возможность задать приоритетную ветвь путем установки предпочтений для соответствующего перехода.

4. Выбранный переход срабатывает, а в журнал событий добавляется соответствующая запись, содержащая имя произведенного действия и временную метку. Gena также позволяет записывать и данные об использовавшихся процессом ресурсах. Но этот функционал выходит за рамки данной работы. Генератор допускает два режима работы: непосредственная запись временной метки при срабатывании или запись двух событий, соответствующих началу и окончанию действия. Если выбран второй режим, во время начала срабатывания перехода в журнал записывается событие начала действия, а затем происходит продолжение генерации журнала. Через заданный пользователем для конкретного действия временной промежуток его ис-

полнение считается оконченным. В журнал добавляется запись, соответствующая окончанию действия, и временная метка.

5. В случае, если выбран режим генерации журнала с помехами, во время срабатывания перехода выполняется дополнительная работа для определения, является ли срабатывание данного перехода абсолютно корректным или необходимо смоделировать помеху. Исходя из заданных пользователем настроек и предпочтений, а также на основании выпавших случайных чисел, Gena решает, есть ли необходимость пропустить срабатывание перехода (не записывать информацию о срабатывании в журнал), повторить срабатывание, заменить имя на другое, исказить временную метку или имя ресурса. По умолчанию, в режиме помех срабатывания любых переходов могут быть равновероятно искажены. Вероятность искажения при срабатывании конкретного перехода определяется на основании заданного пользователем общего уровня отсечки (от 0 до 100).

6. Во все позиции, в которые исходят дуги из сработавшего перехода, добавляется необходимое количество маркеров с соответствующими идентификаторами трасс.

7. Из входных для сработавшего перехода позиций удаляется соответствующее количество маркеров.

8. Подобные шаги повторяются, пока не будет достигнута заданная пользователем финальная разметка.

9. Если достичь финальной разметки не получается за заданное пользователем число шагов (например, для модели, показанной на рисунке 1, из-за ошибки системы может оказаться так, что процесс лечения в цикле повторяется бесконечно), считается, что трасса не завершилась успешно. Программа Gena позволяет как оставлять такие трассы в журнале, так и удалять их, оставляя только успешно завершенные. Надо иметь в виду, что в случае больших моделей с циклами количество шагов, необходимых для завершения исполнения процесса, может оказаться очень существенным, что сказывается на времени генерации журнала.

10. Новые трассы начинаются либо в случайный момент времени по ходу генерации, либо при попадании в ситуацию, когда финальная разметка достигнута, а новых активных переходов найти не удается. Для начала новой трассы в позиции, соответствующие начальной разметке, добавляется необходимое количество маркеров, содержащих идентификатор новой трассы.

Пользователь программы задает количество трасс в генерируемом журнале событий, а также количество журналов. После того, как один журнал с заданным количеством трасс будет сгенерирован, система автоматически перезапускается и начинается генерация следующего журнала. Таким образом, генерация нескольких журналов происходит последовательно. При этом временные метки генерации могут задаваться искусственно так, будто бы все журналы содержат поведение, исполненное в одно время. Это бывает полезно для сравнительного тестирования алгоритмов.

Описание эксперимента. В качестве исходных данных использовались три тестовые модели: T (рисунок 2), S (рисунок 2) и M (рисунок 3).

С каждой моделью проводился следующий эксперимент:

1. Первым шагом была генерация журнала событий без шума, соответствующего идеально правильному поведению информационной системы. Для простоты

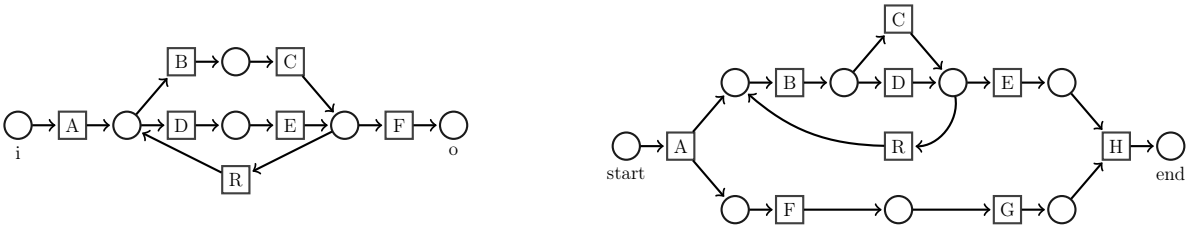

Рис. 2. Модели T (7 переходов, 6 позиций) и S (9 переходов, 9 позиций)

в ходе эксперимента длительности действий процесса задавались таким образом, чтобы одно исполнение процесса примерно попадало в одни сутки. Для генерации использовались настройки по умолчанию, процесс не использовал ресурсы или дополнительные характеристики модели.

Для этого начального журнала событий измерялись характеристики его соответствия модели. Эти характеристики в дальнейшем использовались как эталонные и сравнивались с характеристиками соответствия моделей и журналов событий, полученных путем генерации с добавлением искусственного шума.

2. С помощью программы Gena генерировались журналы событий с дополнительными помехами двух уровней (3% и 10% срабатываний переходов с шумом). Помехи использовались следующего характера: (1) случайная перестановка имен событий (во время срабатывания перехода в журнал событий записывается имя другого случайно выбранного перехода из модели); (2) ложная запись одного из трех специальных случайно выбираемых событий (имена событий: *No1*, *No2*, *No3*); (3) комбинация характера помех из первого и второго пунктов, то есть перестановка имен внутри модели и добавление новых событий; (4) помехи при записи временных меток: случайный сдвиг события по времени и периодическая помеха с округлением времени до 5 и 15 минут (имитируются системные часы с грубой шкалой, округление персоналом). Таким образом, для каждой из тестовых моделей генерировался набор тестовых журналов событий, соответствующих помехам разного характера.

3. По каждому из полученных журналов событий строились четыре модели с помощью четырех плагинов среды ProM, а именно: (1) Alpha Miner; (2) ILP Miner; (3) Heuristics Miner; (4) Inductive Miner. Каждый из этих плагинов является реализацией одного из основных алгоритмов извлечения модели процесса и имеет свою специфику применения.

Alpha Miner реализует альфа-алгоритм, который является фактически первым и простейшим алгоритмом извлечения модели процесса [1,2]. В данной работе не приводится подробное описание алгоритмов извлечения модели, так как подобное описание само по себе оказалось бы слишком объемным. Важно отметить, что альфа-алгоритм не является достаточно эффективным, а также не обладает механизмами, позволяющими обрабатывать журналы событий с шумом. Плагин *Alpha Miner* не содержит каких-либо настроек по причине простоты алгоритма, который он реализует. Следует отметить, что семейство альфа-алгоритмов в наибольшей степени заточено на максимизацию критерия простоты модели [7]. Более подробное описание алгоритма содержится в [2].

Более сложным является алгоритм, использующий линейное программирование при извлечении модели. Не вдаваясь в подробности, отметим, что во многих алго-

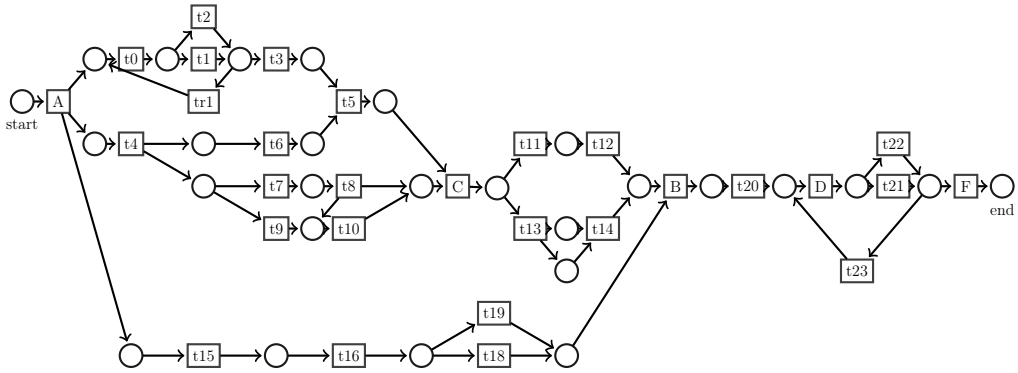

Рис. 3. Модель M (29 переходов, 27 позиций)

ритмах сеть Петри синтезируется из журнала событий с промежуточным звеном в виде системы переходов. Для минимизации количества состояний в такой системе и решается задача линейного программирования. Реализует данный алгоритм плагин под названием *ILP Miner* [22]. Данный плагин имеет некоторые настройки, относящиеся к выбору решателя и ограничениям по производительности. В ходе эксперимента использовались настройки по умолчанию. Следует отметить, что данный плагин работает очень медленно в случае больших моделей (журналов, содержащих большое количество различных классов событий). В некоторых случаях в ходе эксперимента извлечь модель с помощью *ILP Miner* не удалось. Более подробно алгоритм описан в [22].

Heuristics Miner реализует эвристический алгоритм извлечения модели [23]. Отличием данного алгоритма является то, что при построении модели учитываются частотные характеристики событий в журнале. Эвристическая модель, извлекаемая с помощью алгоритма, содержит информацию о том, какие узлы и дуги модели чаще задействуются при исполнении заданного журнала событий. Таким образом, возможным становится выделение основных и второстепенных частей в модели. Эвристический алгоритм справляется с шумом и способен выдавать модель, отражающую только основное поведение, из записанного в журнале событий. Детали и редкие особенности исполнения процесса отбрасываются алгоритмом. Более подробно алгоритм и свойства эвристических моделей описаны в [23]. Важно отметить, что плагин *Heuristics Miner* в качестве результата выдает эвристическую модель, в отличие от других плагинов, используемых в данной работе. К счастью, совместно с плагином извлечения модели поставляется плагин преобразования эвристической модели в сеть Петри. В ходе эксперимента использовались настройки плагина по умолчанию.

Наиболее современным алгоритмом является алгоритм индуктивного извлечения модели, реализованный в виде плагина *Inductive Miner* [11]. Этот алгоритм также рассматривает журнал событий с вероятностной стороны, то есть может справляться с помехами. Главным его достоинством является тот факт, что модели, получаемые в результате, всегда корректны и исполнимы. Тогда как другие алгоритмы не гарантируют этого факта. Более того, как описывается в [11], алгоритм содержит специальные средства для работы с неполными и испорченными

журналами событий. В ходе эксперимента использовались настройки плагина по умолчанию.

4. Полученные с помощью различных плагинов для каждого из журналов модели оценивались по критериям соответствия.

Результаты эксперимента. Для исходных моделей, показанных на рисунках 2 (T, 7 переходов, 6 позиций, 14 дуг), 2 (S, 9 переходов, 9 позиций, 20 дуг) и 3 (M, 29 переходов, 27 позиций, 67 дуг), были сгенерированы журналы событий без помех, соответствующие идеально правильному поведению, а также с помехами.

Модель	Журнал	Шум, %	Настройки генерации
T	eT0	0	20 трасс, паузы 900-1200 сек, исполнение операции 6200-8200 сек
T	eT3-1	3	искусственные события
T	eT3-2	3	перестановка порядка действий A, C, R
T	eT3-3	3	перестановка действий + искусственные события
T	eT3-4	3	шум во временных метках с макс. отклонением в 14400 сек
T	eT5-1	10	искусственные события
T	eT5-2	10	перестановка порядка действий A, C, R
T	eT5-3	10	перестановка действий + искусственные события
T	eT5-4	10	шум во временных метках с макс. отклонением в 14400 сек
S	eS0	0	20 трасс, паузы 900-1200 сек, исполнение операции 6200-8200 сек
S	eS3-1 – eS3-4	3	аналогично журналам eT3-1 – eT3-4
S	eS5-1 – eS5-4	10	аналогично журналам eT5-1 – eT5-4
M	eM0	0	20 трасс, паузы 900-1200 сек, исполнение операции 2600-4600 сек
M	eM3-1 – eM3-4	3	аналогично журналам eT3-1 – eT3-4
M	eM5-1 – eM5-4	10	аналогично журналам eT5-1 – eT5-4

Таблица 2. Сгенерированные журналы событий.

Таблица 2 содержит перечисление экспериментальных журналов событий, созданных в ходе опыта, а также настройки их генерации. Конкретное количество событий в каждом из журналов было различно и варьировалось от 176 событий в журнале без шума для самой малой модели до 561 для большой модели. Журналы колоссальных размеров сознательно не использовались в данной работе. Цель состояла не в том, чтобы оценить производительность алгоритмов, а в том, чтобы определить их способности по обработке журналов с шумом.

Для каждого из журналов событий из таблицы 2 были построены модели с помощью четырех алгоритмов извлечения модели: Alpha, Heuristics, ILP, Inductive. Привести изображения всех моделей в данной работе не представляется возмож-

ным ввиду ограничений объема. В таблицу 3 вынесены результаты проверки соответствия по разным критериям между исходным журналом событий без шума и моделями, построенными в ходе эксперимента. Методики измерения соответствия, точности и обобщенности описаны в [3–6]. В качестве метрики сложности модели использовались количества узлов и дуг модели.

Журнал	Алгоритм	Соответствие	Точность	Обобщенность	Узлов	Дуг
eT0	Alpha	1,0	0,86	0,99	13	14
eT0	Heuristics	0,83	0,82	0,99	29	30
eT0	ILP	0,89	0,82	0,99	12	13
eT0	Inductive	1,0	0,86	0,99	15	16
eT3-1	Alpha	0,31	1,0	0,96	23	35
eT3-1	Heuristics	0,46	0,34	0,87	>40	>40
eT3-1	ILP	1,0	–	–	>50	>50
eT3-1	Inductive	1,0	0,77	0,99	20	22
eT3-2	Alpha	0,11	0,15	0,0	>20	>20
eT3-2	Heuristics	0,69	0,88	0,95	20	20
eT3-2	ILP	1,0	0,60	0,86	11	23
eT3-2	Inductive	1,0	0,19	0,99	22	28
eT3-3	Alpha	0	0	0	>30	>30
eT3-3	Heuristics	0,1	–	–	30	30
eT3-3	ILP	0,97	0,43	0,97	15	>30
eT3-3	Inductive	1,0	0,86	0,99	15	16
eT3-4	Alpha	1,0	0,86	0,99	13	14
eT3-4	Heuristics	0,69	0,88	0,95	20	20
eT3-4	ILP	1,0	0,86	0,99	12	13
eT3-4	Inductive	1,0	0,86	0,99	15	16

Таблица 3. Соответствие исходного журнала событий для модели T моделям, извлеченным из журналов с различными характеристиками.

Алгоритмы Heuristics и ILP выдают довольно сложные модели, добавляя большое относительное количество дополнительных элементов (например, для журнала без шума эвристический алгоритм увеличивает размер модели вдвое). Более того, эти алгоритмы не дают гарантии, что модель получится исполнимой. В то же время алгоритмы ILP и Inductive гарантируют получение модели с идеальным соответствием. Что, правда, негативно отражается на точности, обобщенности и размере модели.

В некоторых случаях характеристики модели посчитать в разумное время не удалось. Например, подобное часто случается с моделями, построенными с помощью алгоритма ILP, которые оказываются очень сложны структурно. Как было указано выше, алгоритм Alpha совершенно не способен справляться с шумом в модели, что подтверждается данными даже для самых простых моделей. Для журналов eT3-2 – eT3-4, eS3-2 – eS3-4, eM3-2 – eM3-4 этот алгоритм выдает более или менее бессмысленные модели с крайне низким соответствием.

Алгоритм Heuristics дает неплохие результаты для зашумленных журналов, вне

зависимости от размера модели. Алгоритм Inductive справляется с шумом, удерживает соответствие модели журналу на уровне 1, обобщенность также сохраняется на довольно высоком уровне. Проблемой этого алгоритма является уменьшение точности с увеличением количества шума в модели.

Сильной стороной алгоритма Alpha является тот факт, что он не обращает внимания на временные метки, руководствуясь только последовательностью записи событий в журнале. Благодаря этому шум во временных метках совершенно не оказывает влияния на его работу. Из результатов также видно, что с перестановкой действий алгоритмы справляются лучше, чем с новыми действиями в журнале.

В данной работе приведена только часть экспериментальных данных по причине ограниченности места. Закономерности, прослеживаемые в приводимых данных, прослеживаются и для всех остальных моделей.

В таблице 4 можно увидеть результаты проверки соответствия по разным критериям между исходной моделью и различными журналами событий с шумом и без. Из таблицы видно, как наличие шума в журнале событий влияет на соответствие его модели. Интересно, что на точность больше всего влияет изменение порядка событий при срабатывании, тогда как на соответствие влияют помехи во временных метках. Для журнала eM5-4 обобщенность модели выросла с добавлением шума в журнал. Такой эффект вполне ожидаем. Поведение с шумом, записанное в журнале событий, значительно отличается от поведения модели. В то же время, модель допускает намного более разнообразное поведение по сравнению с конкретными исполнениями процесса, содержащимися в журнале событий.

Журнал	Соответствие	Точность	Обобщенность
eM0	1,0	0,62	0,85
eM3-1	0,92	0,61	0,83
eM3-2	0,93	0,60	0,78
eM3-3	0,92	0,61	0,83
eM3-4	0,85	0,58	0,78
eM5-1	0,82	0,61	0,84
eM5-2	0,79	0,61	0,83
eM5-3	0,84	0,60	0,78
eM5-4	0,77	0,61	0,89

Таблица 4. Соответствие исходной модели журналам событий для модели M.

6. Заключение

Произведенные эксперименты показывают, что шум различных типов изменяет результирующую извлекаемую модель. Тип применяемого подхода имеет значение, но шум оказывает влияние для любого применяемого алгоритма. В зависимости от типа шума влияние может быть различным по характеру. При наличии шума во временных метках событий в журнале итоговые модели могут получаться отличными от полученных из соответствующих журналов событий без шума. В то же время,

многие алгоритмы справляются с шумом в виде перестановок событий в журнале, когда корректность временных меток не нарушается.

Для больших по размеру логов, содержащих каждый вариант исполнения процесса в нескольких экземплярах, удаление некорректных трасс не влияет на вид итоговой модели. В случае, если в журнале событий представлен очень малый объем поведения процесса, алгоритмы извлечения становятся неустойчивыми. Небольшое изменение в журнале приводит к сильным изменениям в итоговой модели.

Разработчики информационных систем должны быть очень внимательны к корректности записи данных о функционировании процесса, особенно временных меток в случае, если предполагается дальнейшее отслеживание функционирования системы или улучшение параметров процесса. В области извлечения и анализа процессов разработаны многие алгоритмы, способные обрабатывать журналы с шумом (Heuristics, Genetic, Flexible, Inductive). Тем не менее, работа с журналами событий, содержащими шум, часто заключается в простой фильтрации с удалением трасс, содержащих шум, или игнорировании событий с некорректными записями [1]. Алгоритмы, успешные в работе с шумом, как правило, выдают в качестве результата модели не в виде классических сетей Петри (см., например, [23]). Преобразование в сеть Петри возможно почти для всех типов моделей, однако увеличивает количество потенциальных ошибок, а также сильно усложняет модель.

Некоторые из видов шума (например, шум во временных метках) очень коварны и совершенно не обрабатываются. Алгоритм выдает, казалось бы, корректную модель, совершенно отличную от модели для аналогичного журнала без шума. Это означает, что, во-первых, исследователи должны быть внимательны при использовании методик извлечения процессов с журналами событий, содержащими шум. Во-вторых, необходима разработка более совершенных алгоритмов извлечения моделей процессов для обработки журналов с шумом. В частности, хорошие перспективы имеет разработка алгоритмов, учитывающих при построении модели дополнительную информацию, привязанную к событиям: данные, ресурсы, роли исполнителей и другую. Подобные алгоритмы будут более устойчивы к помехам во временных метках.

Список литературы

1. *Van der Aalst W. M. P.* Process mining: discovery, conformance and enhancement of business processes. Springer, 2011.

2. *Van der Aalst W.M.P., Weijters A.J.M.M., Maruster L.* Workflow Mining: Discovering Process Models from Event Logs // IEEE Transactions on Knowledge and Data Engineering, 2004. Vol. 16(9). P. 1128–1142.

3. *Van der Aalst W.M.P., Adriansyah A., Van Dongen B.F.* Replaying history on process models for conformance checking and performance analysis // Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. Vol. 2(2). Wiley Online Library. 2012. P. 182–192.

4. *Adriansyah A., Van Dongen B.F., Van der Aalst W.M.P.* Conformance checking using cost-based fitness analysis // 15th IEEE International Conference on Enterprise Distributed Object Computing Conference (EDOC). 2011. P. 55–64.

5. *Adriansyah A., Van Dongen B.F., Van der Aalst W.M.P.* Towards robust conformance checking // Business Process Management Workshops. Springer. 2011. P. 122–133.

6. *Adriansyah A., Munoz-Gama J., Carmona J., Van Dongen B.F., Van der Aalst W.M.P.* Alignment Based Precision Checking // Business Process Management Workshops. Springer. 2012. P. 137–149.

7. *Buijs J.C.A.M., Van Dongen B.F., Van der Aalst W.M.P.* On the Role of Fitness, Precision, Generalization and Simplicity in Process Discovery // 20th International Conference on Cooperative Information Systems (CoopIS 2012). Ser. lncs. 2012.

8. *Van Dongen B. F., Van der Aalst W. M. P., Günther C. W., Rozinat A., Verbeek E., Weijters T.* ProM: the process mining toolkit // Business Process Management Demonstration Track (BPMDemos2009). Ser. CEUR Workshop Proceedings, A. K. A. d. Medeiros and B. Weber, Eds. 2009. Vol. 489. P. 1–4.

9. *Kalenkova A. A., Lomazova I. A.* Discovery of Cancellation Regions within Process Mining Techniques // Proceedings of the 22nd International Workshop on Concurrency, Specification and Programming. Warsaw, Poland, 2013. P. 232–244.

10. *Kalenkova A. A., Lomazova I. A., Van der Aalst W. M. P.* Process Model Discovery: A Method Based on Transition System Decomposition // Application and Theory of Petri Nets and Concurrency, LNCS 8489. Springer, 2014. P. 71–90.

11. *Leemans S. J. J., Fahland D., Van der Aalst W. M. P.* Discovering Block-Structured Process Models from Incomplete Event Logs. Tech. Rep. BPM-14-05. Eindhoven University of Technology. March 2014.

12. *Munoz-Gama J., Carmona J., Van der Aalst W. M. P.* Conformance Checking in the Large: Partitioning and Topology // International Conference on Business Process Management (BPM 2013), LNCS 8094. Springer-Verlag, Berlin, 2013. P. 130–145.

13. *Verbeek H. M. W., Buijs J. C. A. M., Van Dongen B. F., Van der Aalst W. M. P.* Prom 6: The process mining toolkit // Proceedings of BPM Demonstration Track. 2010. Vol. 615. P. 34—39.

14. *Verbeek H. M. W., Buijs J. C. A. M., Van Dongen B. F., Van der Aalst W. M. P.* XES, XESame, and ProM 6 // Information Systems Evolution, Lecture Notes in Business Information Processing. 2011. Vol. 72. P. 60–75. DOI : 10.1007/978-3-642-17722-4_5

15. http://www.xes-standard.org/xesstandarddefinition

16. *Rogge-Solti A., Mans R. S., Van der Aalst W. M. P., Weske M.* Repairing Event Logs Using Timed Process Models // OTM 2013 Workshops, LNCS 8186. 2013. P. 705–708.

17. *Rozinat A.* Process Mining: Conformance and Extension. PhD Thesis, Eindhoven University of Technology. 2010.

18. *Rubin V. A., Lomazova I. A., Van der Aalst W. M. P.* Agile Development with Software Process Mining // Proceedings of the 2014 International Conference on Software and System Process (ICSSP 2014). Nanjing, China. ACM, 2014. P. 70–74.

19. *Rubin V. A., Mitsyuk A. A., Lomazova I. A., Van der Aalst W. M. P.* Process Mining Can Be Applied to Software Too! // Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM 2014). Torino, Italy. ACM, 2014. P. 57:1–57:8.

20. *Shugurov I., Mitsyuk A. A.* Generation of a Set of Event Logs with Noise // Proceedings of the 8th Spring/Summer Young Researchers' Colloquium on Software Engineering (SYRCoSE 2014). 2014. P. 88–95.

21. http://pais.hse.ru/research/projects/gena

22. *Van der Werf J. M. E. M. et al.* Process discovery using integer linear programming // Applications and Theory of Petri Nets. Springer Berlin Heidelberg, 2008. P. 368–387.

23. *Weijters A., Van der Aalst W. M. P., De Medeiros A. K. A.* Process mining with the heuristics miner-algorithm // Technische Universiteit Eindhoven, Tech. Rep. WP. 2006. Vol. 166. P. 1–34.

On Process Model Synthesis Based on Event Logs with Noise

Mitsyuk A. A., Shugurov I. S.

National Research University Higher School of Economics,
International laboratory of process-aware information systems,
Kochnovskiy Proezd, 3, Moscow, 125319, Russia

Keywords: process mining, Petri net, event log, event log generation, ProM

Process mining is a new emerging discipline related to process management, formal process models, and data mining. One of the main tasks of process mining is the model synthesis (discovery) based on event logs. A wide range of algorithms for process model discovery, analysis, and enhancement is developed. The real-life event logs often contain noise of different types. In this paper we describe the main causes of noise in the event logs and study the effect of noise on the performance of process discovery algorithms. The experimental results of application of the main process discovery algorithms to artificial event logs with noise are provided. Specially generated event logs with noise of different types were processed using the four basic discovery techniques. Although modern algorithms can cope with some types of noise, in most cases, their use does not lead to obtaining a satisfactory result. Thus, there is a need for more sophisticated algorithms to deal with noise of different types.

Сведения об авторах:
Мицюк Алексей Александрович,
Национальный Исследовательский Университет Высшая Школа Экономики,
Лаборатория ПОИС, аналитик
Шугуров Иван Сергеевич,
Национальный Исследовательский Университет Высшая Школа Экономики,
Лаборатория ПОИС, стажер-исследователь